

Diss. ETH No. 26900

MACHINE LEARNING ON MOLECULAR
NETWORKS TO DECIPHER THE GENETICS
UNDERLYING COMPLEX TRAITS

A dissertation submitted to attain the degree of
DOCTOR OF SCIENCES OF ETH ZURICH
(Dr. sc. ETH Zurich)

presented by

ANJA CATHRIN GUMPINGER
M.Sc., Technical University Munich
born on 24.01.1989
citizen of Germany

accepted on the recommendation of

Prof. Dr. Karsten Borgwardt
Prof. Dr. Niko Beerenwinkel
Prof. Dr. Kasper Lage

2020

Anja Cathrin Gumpinger

Machine learning on molecular networks to decipher the genetics underlying complex traits,

© 2020.

Für Wolfi.

*I am convinced that the crux of the problem of learning is recognizing relationships and
being able to use them.*

Christopher Strachey in a letter to Alan Turing, 1954.

Abstract

The past decades have been coined by technological advancements at remarkable rates, leading to the generation and collection of unprecedented amounts of data. This is especially the case in the field of genetic research: novel technologies enable a faster and cheaper collection of data at large scales, sparking the hope that the holy grail of personalised medicine is within reach. However, getting there is conditioned on our capability to make sense of this ever-growing wealth of data. While data obtained from different experiments can differ widely, many genetic data sets are governed by an underlying *graph structure*, and leveraging this graph structure holds great potential to better understand complex molecular processes underlying complex traits or diseases. This dissertation describes the combination of molecular networks with genetic data in two different contexts, that is network-guided association studies, and network-guided discovery of cancer driver genes.

The first part of the thesis evolves around network-guided association studies which extend classical genome-wide association studies (GWASs) with molecular networks. In a traditional GWASs individual point-mutations in the human genome are tested for their association with a trait of interest. The loci discovered in those studies often fail to fully explain the observed trait — a phenomenon called *missing heritability*. Different hypothesis that try to explain missing heritability exist. One of them, the existence of non-linear interactions between genetic loci, is at the heart of our contributions. We focus on the non-linear interaction model of genetic heterogeneity, which assumes that multiple loci might influence the phenotype in a similar direction. Hence, our goal is to find interactions between genetic loci that are associated to a phenotype of interest under a model of genetic heterogeneity. This search is complicated by the vast number of potential interactions, posing a statistical challenge in the form of an immense multiple hypothesis testing problem, and a computational challenge, as a combinatorial number of interactions has to be explored. To address these challenges, we leverage two concepts. Firstly, we integrate molecular networks that describe interactions between genes in the form of a graph. Instead of testing all possible interactions between genotyped variants, we only test interactions that are supported by a molecular network. This reduces the search space significantly. Secondly, we exploit concepts from the field of significant pattern mining, a recently emerging field of machine learning. Significant pattern mining concepts have proven useful in addressing statistical and computational challenges that arise when searching for interactions in large-scale genetic analyses.

We developed two different approaches, **tNeAT** and **SiNIMin** to enable network-guided association studies. Extensive studies on simulated and real-world data sets indicate that genetic heterogeneity takes place at small scales, which means interactions within a single gene, or between pairs of genes, but rarely at the level of multiple genes. Especially with

our proposed SiNIMin approach, we were able to identify novel genetic interactions in the plant *Arabidopsis thaliana*, and in a cohort of migraine patients.

The second part of this thesis evolves around the integration of molecular networks with genetic mutation scores to discover genes that are, upon mutation, implicated in cancer. While there exists a rich body of literature on this topic, most existing methods approach this task from a purely unsupervised perspective, ignoring the knowledge of well-established cancer genes. We propose a new node embedding scheme that creates a feature representation for each gene based on the molecular network in a two-step procedure: first, each gene is described by the distribution of features in its neighbourhood via moments of the distribution, and in the second step, those moments are propagated through the network. We refer to this two-step embedding procedure as moment propagation (MoPro) embeddings, and use them as input to binary classifiers to predict the cancer status of a gene. To enable this classification, we use a set of well-established cancer driver genes as the true positive class label. This creates two major challenges: (i) a highly imbalanced data set, as the number of known cancer drives is relatively small compared to the body of all genes, and (ii) a lack of a high-quality negative class, which we address with a sophisticated cross-validation scheme. Our proposed approach led to the discovery of a set of fourteen cancer driver genes that show strong links to cancer, and that constitute promising candidates for further biological evaluation. Their detection showcases the potential of combining network-derived features with supervised machine learning for the discovery of genes implicated in cancer.

Our results illustrate the usefulness of integrating genetic analysis with molecular networks to enhance our understanding of the genetics underlying complex traits. Molecular networks that capture complex cellular processes enable a holistic view of trait-causing mechanisms at various scales. Adopting this holistic view of genetics is, and will remain, a promising line of research.

Zusammenfassung

Die vergangenen Jahrzehnte waren geprägt von technologischem Fortschritt, der dazu führte, dass immer grössere Mengen an Daten in zunehmend kürzerer Zeit erhoben werden konnten. Dies betrifft vor allem die Genetik: die stete Entwicklung neuer und die Verbesserung bestehender Technologien beschleunigt die Erhebung grosser Datensätze. Das Ziel der personalisierten Medizin scheint in greifbare Nähe zu rücken. Es zu erreichen setzt jedoch voraus, dass wir systematisch Wissen aus der stets wachsenden Menge an Daten generieren können. Netzwerke, die vielen dieser genetischen Daten zugrunde liegen, können uns dabei helfen. Sie beschreiben typischerweise Interaktionen zwischen verschiedenen genetischen Einheiten, zum Beispiel die Interaktion zweier Gene. Die Einbindung solcher Netzwerkstrukturen in Analysen erlaubt es, ein umfassendes Bild komplexer, molekularer Prozesse zu zeichnen, die für die Ausbildung komplexer Merkmale oder Krankheitsbilder bei Menschen verantwortlich sind. In der vorliegenden Dissertation haben wir uns zum Ziel gesetzt, Methoden zur Kombination genetischer Daten mit molekularen Netzwerken, im Kontext klassischer Assoziationsstudien und Krebsstudien, zu entwickeln.

Im ersten Teil dieser Dissertation widmen wir uns einer Einführung in biologische Netzwerke, bevor wir im zweiten Teil die Erweiterung klassischer genomweiter Assoziationsstudien um molekulare Netzwerke beschreiben. In einer klassischen Assoziationsstudie wird der Einfluss einzelner Punktmutationen in der DNA auf einen Phänotypen durch statistische Tests untersucht. Die Punktmutationen, die in solchen Studien gefunden werden, können häufig nur einen Teil der genetischen Heritabilität, also Vererbbarkeit, erklären. Eine Hypothese, die dieses Phänomen zu erklären versucht, ist, dass sich ein Teil der genetischen Heritabilität durch nicht-lineare Interaktionen mehrerer genetischer Varianten begründen lässt. Ein solches Modell bildet das Kernstück unserer Arbeit: die genetische Heterogenität. Sie besagt, dass ein phänotypisches Merkmal durch mehrere unterschiedliche genetische Varianten auf eine ähnliche Art und Weise beeinflusst werden kann. Unser Ziel ist es, Interaktionen zwischen mehreren Punktmutationen zu finden, die unter eine statistisch signifikante Assoziation mit einem phänotypischen Merkmal aufweisen. Den kompletten Raum aller möglichen Interaktionen zu erfassen ist jedoch aus statistischer und rechnerischer Sicht aufgrund der Vielzahl an Tests herausfordernd. Um die Suche dennoch zu ermöglichen, binden wir molekulare Netzwerke in den Suchprozess ein. Anstatt alle möglichen Interaktionen zu betrachten, beschränken wir uns auf die Varianten, die auch im Netzwerk miteinander verbunden sind. Zusätzlich formulieren wir unser Problem aus dem Blickwinkel der signifikanten Mustererkennung, einem neuartigen Feld im Bereich des Maschinellen Lernens, dessen Methoden und Konzepte sich besonders für grosse genetische Analysen bewährt haben.

Wir haben zwei neue Methoden entwickelt, **tNeAT** und **SiNIMin**. Beide kombinieren Netzwerke auf unterschiedliche Art und Weise mit genetischen Assoziationsstudien. Aus-

fürliche Simulationsstudien, sowie die Anwendung auf mehreren Datensätzen deuten an, dass genetische Heterogenität vor allem in einem kleinen Massstab eine wichtige Rolle zu spielen scheint. So haben wir hauptsächlich Interaktionen innerhalb einzelner Gene, oder zwischen Paaren von Genen nachweisen können. Besonders SiNIMin fand bisher unbekannt Interaktionen im Modellorganismus *Arabidopsis thaliana*, sowie in einer Studie über Migränepatienten.

Im dritten Teil dieser Dissertation beschreiben wir unseren Beitrag zur netzwerkbasierten Identifikation von Genen, die im mutierten Zustand Krebs zur Folge haben können. Viele unterschiedliche Methoden haben diese Fragestellung untersucht, die meisten jedoch aus einer unüberwachten Perspektive, in der das Wissen über etablierte Krebsgene nicht Teil der Methode ist. Wir haben einen neuartigen Ansatz entwickelt, um Deskriptoren für Gene aus einem Netzwerk zu errechnen. Dazu wird zunächst ein Wert für jedes Gen errechnet, der die Mutationslast des Gens beschreibt. Anschliessend wird jedes Gen durch diese Werte seiner Nachbarn im Netzwerk beschrieben. Die daraus resultierenden Deskriptoren können nun zusammen mit Algorithmen für überwachtes maschinelles Lernen zur Klassifikation von Krebsgenen eingesetzt werden. Dies erfordert Beispiele aus einer positiven und negativen Klasse, die wir aus den etablierten Krebsgenen ableiten. Daraus resultieren zwei grosse Herausforderungen: (i) die positive Klasse enthält nur einen Bruchteil der Gene, während für die meisten Gene keine positive Klasse vorhanden ist, und (ii) es gibt keine eindeutige negative Klasse, da jedes Gen, das noch nicht als Krebsgen klassifiziert wurde, potentiell zur Entwicklung von Krebs beitragen kann. Um die beiden obengenannten Punkte zu adressieren, entwickelten wir ein durchdachtes Schema zur Kreuzvalidierung. Unsere neuartigen Deskriptoren haben es uns ermöglicht, in Kombination mit dem überwachten Ansatz und unserer Kreuzvalidierung, vierzehn Gene zu identifizieren, deren Verbindung zu Krebs durch eine Literaturrecherche bestätigt werden konnte. Sie stellen damit vielversprechende Kandidaten für eine biologische Validierung dar. Dies verdeutlicht das Potential eines kombinierten Ansatzes aus überwachtem Lernen mit netzwerkbasierten Deskriptoren von Genen, um neue Krebsgene zu finden.

Durch die Ergebnisse der vorliegenden Dissertation kommen wir zu dem Schluss, dass die Entwicklung von Methoden, die genetische Daten in Einklang mit molekularen Netzwerken analysieren, wertvolle Beiträge zu unserem Verständnis komplexer phänotypischer Merkmale und deren genetischen Ursachen leisten können. Dieser gesamtheitliche Ansatz, der nicht nur Gene in Isolation, sondern auch deren Wechselwirkungen betrachtet, ist und bleibt eine vielversprechende Forschungsrichtung.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Prof. Dr. Karsten Borgwardt for giving me the opportunity to conduct my doctoral studies in his group, for his constant support, supervision, and most of all his excellent scientific input and guidance. The past years that I had the honour to spend in this group were an incredible learning experience.

I would like to thank Prof. Dr. Niko Beerenwinkel and Prof. Dr. Kasper Lage for being part of my committee, and Prof. Dr. Sven Panke for chairing the doctoral examination.

Next, I would like to thank all my co-authors and collaborators who significantly contributed to the projects described in this thesis. A special thanks to Dr. Bastian Rieck, a true expert on graphs and their implications in machine learning. Thanks for the many invaluable discussions (scientific and non-scientific), for helping with coding-related issues, and for the support during writing papers. I am very glad to have had the opportunity to co-author a paper with him. Thanks to Dr. Damian Roqueiro who has been a source of great support, knowledge and patience, and who significantly shaped my understanding of the intersections between genetics and machine learning. I also would like to thank Prof. Dr. Dominik Grimm, who co-authored two of my publications, for his expertise and guidance. He contributed significantly into designing experiments in the model organism *Arabidopsis thaliana*, and was always offering a helping hand. Thanks to Dr. Heiko Horn, who taught me a lot about machine learning for cancer genetics, for many fruitful discussions that led to a joint publication. Thanks to our collaborators from the bioprocess laboratory at D-BSSE, Dr. Markus Jeschek and Simon Höllner, as well as Prof. Dr. Kobi Benenson who heads the synthetic biology group. It was a great pleasure to work with you. And a huge thanks to Dr. Bastian Rieck, Dr. Catherine Jutzeler, Dr. Ulrike Menzel, and Simon Höllner for your time and effort in proof-reading this thesis. Thank you, Cindy Malnasi, the first person I ever spoke to at the D-BSSE, for your constant help and assistance with any administrative issues that would come up.

Furthermore, I would like to thank all past and current members of the MLCB group whom I had the pleasure to work with: Thanks Dr. Dean Bodenham, Christian Bock, Dr. Sarah Bruennigk, Bowen Fan, Dr. Lukas Folkman, Elisabetta Ghisu, Udo Gieraths, Thomas Gumbsch, Dr. Xiao He, Dr. Katharina Heinrich, Dr. Felix Hensel, Max Horn, Dr. Catherine Jutzeler, Dr. Juliane Klatt, Birgit Knapp, Dr. Felipe Llinares-López, Dr. Michael Moor, Giulia Muzio, Leslie O'Bray, Laetitia Papaxanthos, Dr. Bastian Rieck, Dr. Damian Roqueiro, Matteo Togninalli, Caroline Weis, Menno Witteveen, and Dr. Daisuke Yoneoka for endless fruitful and inspiring scientific discussions, coffee breaks, and after-work beers. You made this experience a great one.

Last, but by no means least, I would like to extend my gratitude to my family and friends. My parents, Irene and Christoph, none of this would have been possible without your

constant love and support. I cannot even start to explain how important your contribution is. Thanks to my brother, Sebastian, for always believing in me. Simon, my partner and best friend, thanks for your love and patience, for always standing by me, having my back and supporting me. And of course, thanks to Franzi, Adon and Emil, Lami and Schildo.

Contents

I	Introduction to biological networks	1
1	The age of network biology	2
1.1	Biological networks and their application in biomedical research	3
1.2	Discovering gene-interactions with molecular networks	6
1.3	Contributions of the thesis	7
II	Network-guided genetic association studies	11
2	Introduction to genetic association studies	13
2.1	Deciphering the genetics underlying complex traits	13
2.2	Concepts and methods	14
2.3	An overview of network based genetic analysis	21
2.4	New directions in network based genetic analyses	27
3	Introduction to significant pattern mining	29
3.1	Significant pattern mining	29
3.2	Notation and problem statement	30
3.3	Association testing and the multiple comparison problem	33
3.4	Enabling significant pattern mining with Tarone’s testability criterion	41
3.5	Accounting for pattern dependence: extension to permutation testing	46
3.6	Accounting for categorical covariates	49
3.7	Significant pattern mining in biology and medicine	53
4	Network-guided testing of local neighbourhoods	55
4.1	Finding significant neighbourhoods in networks	56
4.2	The tNeAT method	62
4.3	The tNeAT-WY method	64
4.4	Implementation details	67
4.5	Simulation study	68
4.6	Application to 20 <i>Arabidopsis thaliana</i> phenotypes	74
4.7	Summary and discussion	80
5	Network-guided testing of gene-pairs	82
5.1	Finding significant segment interactions	83
5.2	The SiNiMin method	89
5.3	The SiNiMin-WY method	92
5.4	Implementation details	94
5.5	Simulation study	96
5.6	Application to 20 <i>Arabidopsis thaliana</i> phenotypes	104

Contents

5.7	Study on low-frequency variants in migraine	108
5.8	Summary and discussion	113
6	Discussion and outlook	116
III	Network-based identification of cancer driver genes	122
7	Introduction to computational cancer genetics	124
7.1	Cancer as a result of genetic aberrations	124
7.2	Data availability	126
7.3	Detection of cancer driver genes from somatic mutations	128
8	Network-based classification of cancer driver genes	130
8.1	Network-based moment propagation of mutation scores	132
8.2	Prediction of cancer driver genes for a TCGA pan-cancer study	138
9	Discussion and outlook	153
IV	Discussion and outlook – Learning from structured data	159
10	Learning from structured data	161
10.1	Time-structured data	161
10.2	Summary and Discussion	164
10.3	Trends and challenges in learning from graph-structured data	166
10.4	Closing remarks	168
V	Appendix	170
A	Data-simulation for significant pattern mining approaches	171
B	Software packages	174
	Acronyms	176
	Glossary	177
	List of Figures	179
	List of Tables	180
	Bibliography	181

Part I.

Introduction to biological networks

1. The age of network biology

The technological advancements over the past decades enabled the collection of unprecedented amounts of data, ushering in the so-called *age of Big Data* [1]. The availability of more and larger data sets is continuously transforming many scientific disciplines towards a more data-driven discovery and decision making, impacting social and computer sciences, economics, transportation and telecommunication, biology, medicine and health care [2–8]. The Big Data revolution presumably has the potential to impact many aspects of our society, by improving the forecasts of environmental, political or financial crises, allowing to predict an individual’s risk to develop a disease, enhancing personalised treatments of cancer, or making every-day life easier by increasing customer satisfaction [9–15].

While the data obtained from different experiments can differ widely, many data sets are governed by an underlying *graph structure*. From a mathematical point of view, a graph represents a set of elements (*nodes* or *vertices*) and a set of, potentially directed, relationships between those elements (*edges*). In practise, this graph structure constitutes a different view of the data that potentially encodes useful prior knowledge and upon integration with other data sources, can serve as a guide during data analysis or for the interpretation of results. A prominent example of graph-structured data is the representation of individuals in a social network, where each individual is represented as a node in the network, and two nodes are connected by an edge if the corresponding individuals know each other. This network-representation of social interactions can help in various tasks, such as personalised advertisement: each individual, i.e. node, in the network could be represented by a set of features, such as age, gender, favourite music, films and books. In order to suggest a novel book to an individual, the graph structure can be leveraged by using the assumption that a person might prefer a book that a friend, i.e. a connected node, enjoyed.

Especially in biology graph structures are omnipresent. As opposed to analysing biological entities such as genes or proteins in isolation, graphs give a holistic view of complete systems on different scales, ranging from the structural representations of molecules and proteins to large scale networks that represent relationships and mechanisms in human cells, or even ecological networks that illustrate relationships in whole ecosystems. There exist countless examples where graphs naturally represent biological objects and relationships between their components. Proteins, for instance, can be represented as graphs, where nodes correspond to amino acids, and edges correspond to physical interactions between them [16]; Another option is to represent proteins as graphs where nodes correspond to secondary structure elements, and nodes are connected by physical proximity [17]; the cell can be represented as a graph, where nodes correspond to molecules in the cell, and interactions represent thermodynamics of binding between them [18]; the human genome can be viewed from a graph perspective using so-called protein-protein interaction networks, where nodes correspond to gene products (proteins) and an edge between genes

1. The age of network biology

indicates an interaction between them [19].

Those biological networks originate from various sources, including systematic experimental screens, literature curation or computational inference [20]. With the ever-accelerating technological progress in those areas, the graph structures themselves are subject to the Big Data revolution. The last decade has seen a steep increase in both, the number of available molecular networks, as well as the size of those networks with respect to the number of nodes and edges contained in them [20]: for example, in January 2010, the BioGRID database [21] that describes protein-protein interactions, contained protein interactions for 17 organisms, including 27'695 genetic interactions between 8'754 unique proteins in homo sapiens [22]. Ten years later, in February 2020, BioGRID contained information on physical and genetic interactions for 70 organisms, with 417'446 unique interactions between 24'730 human proteins [23], describing a 16-fold increase in the number of interactions, and a 3-fold increase in the number of known proteins.

The tremendous increase in the dimensionality of data coupled with the ever-increasing availability of graph-structured data exceeds the statistical and computational capabilities of standard analyses that worked well on small- to mid-sized data sets. Hence, parallel to and in line with the technological developments, computational and statistical methods are required to fully exploit the knowledge contained in those large-scale graph-structured data sets.

1.1. Biological networks and their application in biomedical research

Biological networks are diverse [20]: there exists a vast amount of different network types that describe distinct biological elements and processes and that cover a wide range of interaction types on different scales (for in-depth review, see Liu et al. [24]). Which network to use strongly depends on the research question and focus, as well as the data at hand. Different research goals ask for different network types, and integrating distinct network types for the same problem might emphasise distinct aspects of the same problem. In this section, we give an overview over a variety of such network types that are commonly used in biomedical research, and highlight some of their areas of application.

Protein-protein interaction (PPI) networks represent physical contacts between proteins in a cell. The nature of those contacts can be *stable*, as is the case for the formation of protein complexes, or *transient*, e.g. the short contact between proteins during carrying or modification of proteins [19]. PPIs can be derived from different experiment types, such as yeast two-hybrid assays to measure interactions in cells, or affinity-purification-mass spectrometry to measure the composition of protein complexes [24]. Additionally, edges can be inferred by orthology mapping from other organisms [25], and there exist PPI networks containing interactions that were predicted using computational tools [26]. Some PPI networks contain edge weights that indicate the confidence of an edge [25]. Commonly, interactions that have been validated experimentally obtain high edge weights, while edges that are inferred computationally or from orthology are provided with lower weights. PPI networks are

1. *The age of network biology*

inherently noisy which implies that our knowledge about PPI is incomplete, as well as that interactions that are present in the network correspond to false-positives [27]. PPI networks have been used widely to increase the power of genetic association studies [28–30], and were especially successful in the identification of novel genes associated to cancer development and progression [31, 32].

Metabolic pathways describe reactions in cells that modulate cell growth, reproduction or response to environmental changes [27]. The nodes in the networks correspond to enzymes, proteins, macromolecules, lipids, nucleic acids and carbohydrates, and the links indicate biochemical interactions between them [24]. Other than PPI networks, metabolic pathways are commonly directed, and the edges describe a cascade of chemical processes that subsequently take place in the cell. A comprehensive resource describing the most common pathways is the KEGG pathway database [33]. Metabolic pathways have enabled the interpretation of biological mechanisms underlying many diseases, such as Parkinson’s disease [34] and age-related macular degeneration [35].

Isoform-isoform networks are related to PPI networks, but do not describe interactions between individual proteins, but their isoforms. Protein-isoforms constitute different variants of a single gene, that can be formed e.g. through alternative splicing, alternative transcription, translation, 3’-end formation, or post-transcriptional modifications [36]. It has been shown that isoforms originating from the same gene might have different functions, or even opposite effects in certain processes [37]. This variety of function is not represented in PPI networks, where isoforms of the same protein are grouped into the same node. Isoform-isoform networks account for this variety of descendants of the same gene: each node corresponds to an isoform and edges to interactions between those [38]. It is assumed that the approximately 20’000 human proteins give rise to approximately 100’000 distinct isoform transcripts [39]. Isoform-isoform networks capture those interactions between the genes’ isoform transcripts, and hence deliver more detail than PPI networks. For example in autism, the generation of isoform-isoform networks helped to improve understanding the mechanisms underlying the disease [40].

Genetic interaction networks represent functional relationships (edges) between the genes (nodes). The networks are phenotype-specific, and describe the response to a non-linear perturbation to the connected genes in the cell. An interaction implies that the cell will show a phenotypic effect, such as cell death, upon simultaneous perturbation of both genes, but not upon perturbation of one of the genes [41]. Finding these genetic interactions is its own field of research, as the genetic interaction networks represent dynamics of perturbations involved in the phenotype. Genetic interactions are commonly inferred from RNA inference or CRISPR-Cas9 assays, and provide promising targets especially for drug development [42].

Gene-regulatory network are biological networks that are derived from gene expression data. Various types of biological entities, such as genes, proteins or RNAs constitute the nodes, and edges indicate regulatory relationships between them. Gene regulatory networks are highly phenotype- and tissue-specific [43], and there exist different

1. The age of network biology

computational approaches to assemble networks from gene expression data, ranging from highly biologically motivated to graph-theory driven ones [44].

Brain networks correspond to a graph-based representation of the brain, where nodes correspond to different brain regions. The brain regions can be identified using structural or functional magnet resonance imaging. The edges in the network can either describe physical connections inferred from the image data [45] or functional correlations between the brain regions derived from signal series analyses [46].

Drug target networks are bipartite graphs that describe interactions between drugs and targets (such as genes or proteins). The latter represent the two groups of nodes in the graph, and an edge between a drug and a target indicates that the drug binds the target. Those networks intrinsically contain both therapeutic indications as well as adverse drug effects, and potentially contain valuable information for the identification of novel drug targets, as well as for drug repurposing [47].

Disease networks (diseasomes) can be categorized into two groups: (i) homogeneous networks, in which nodes correspond to diseases, and nodes are linked based on similarities between the corresponding diseases; (ii) heterogeneous networks, that can mathematically be represented as bipartite graphs, in which one node set corresponds to the diseases, and the other node set corresponds to, e.g. disease genes or symptoms. In those heterogeneous networks, nodes from the disease set are connected to nodes from the second set, if the disease is associated with the specific gene or symptom [48].

Biological knowledge graphs are an emerging object of biomedical research. Knowledge graphs are a graph-based representation of knowledge bases that emphasise relations between objects [49]. Within one graph, nodes can correspond to a variety of different biological entities, and edges to relationships between them. As opposed to PPI networks, the edges in knowledge graphs preserve the semantics of different types of associations, such as phosphorylation, inhibition, or activation [50]. A main task in those knowledge graphs is link prediction [49], which refers to the inference of novel edges in the graph. This can for instance be achieved by leveraging vectorial latent-space representations of nodes, so called node-embeddings, in the knowledge graph. Successful examples of knowledge graph applications in biomedical research include the prediction of adverse drug effects [51] or the prediction of drug targets [52, 53].

Biological networks have become a major mode of analysis in various disciplines in computational biology [54]. Especially *molecular networks*, such as protein-protein interaction networks, regulatory networks or metabolic pathways, have been extensively used in biomedical applications [20]. This focus on molecular networks brought some interesting properties regarding their architecture and topology to light. As opposed to random networks that assume a uniform distribution of edges across the network, molecular networks are often found to be *scale-free*. This implies that their degree distribution follows a power law distribution [55], where most nodes only have few links, while few nodes have a large number of links. The latter are often referred to as *hubs* in the network [56]. The existence of hub nodes promotes the second important property, which is the so called *small world*

1. The age of network biology

effect: most nodes in the network can be connected with a path that only contains few edges. In protein-protein interaction networks those hub nodes exhibit a *disassortative* behaviour, which means that hub nodes tend to be connected to nodes with only few neighbours, rather than other hub nodes.

Molecular networks offer a compelling strategy to represent the complexity of the genome. Consequently, those networks were extensively leveraged in studies that aim to unravel genetic causes underlying complex traits and diseases [20] – traits and diseases that are presumably co-determined by a multitude of genetic and environmental factors. Along those lines, an important task is the identification of genes and interactions among genes that jointly form the basis of complex traits and diseases.

1.2. Discovering gene-interactions underlying complex traits with molecular networks

The field devoted to the discovery of novel genes that are implicated in complex traits has flourished in the last decade. Be it via gene-expression analysis or the analysis of genomic data in genome-wide association studies, discovering genes that presumably cause the development of complex traits is at the forefront of science. To achieve the goal of ‘deciphering the genetics underlying complex diseases’, more and more analyses adopt a network-based perspective of genetics and the genome. This network-based perspective is encoded in molecular networks. Integrating molecular networks with genetic data is mainly motivated by the combination of two observations: (i) many complex traits cannot be explained by individual genes alone, and it is rather hypothesised that they are the consequence of a complex interplay between multiple genetic variants [56, 57], and (ii) Goh & Choi [58] observed that genes implicated in the same cellular processes or same diseases tend to cluster in biological networks (*guilt by association*). Taken together, this sparked the idea that genes that contribute to the same trait share more interactions with each other than with randomly chosen genes in the network. Hence, genes that are in close proximity to each other in molecular networks build interesting candidates for joint genetic analyses. From a biological viewpoint, this is corroborated by the fact that cells are capable of rewiring processes in the presence of perturbations to maintain homeostasis [59]. Consequently, such perturbations have to be present at multiple spots simultaneously in order to lead to the manifestation of a (disease) phenotype.

Including molecular networks into analysis pipelines of genetic data is promising for two reasons: (i) they help to interpret and understand results of genetic associations by elucidating the biological mechanisms the genetic variants are involved in, and (ii) they might help to guide the discovery of novel interactions between genes that confer an individual’s susceptibility to a complex trait where the individual genes might show only weak or even no association to the trait. This interaction-based view of genetics is a deviation from the paradigm of classical genetic association studies where commonly proteins and genes are analysed in isolation. However, this deviation constitutes a presumably decisive step, as it remains difficult to understand disease-causing mechanisms and develop therapeutic strategies without this holistic view of genetic mechanisms [57], especially in the case of complex diseases.

1. The age of network biology

This interaction-perspective of complex traits sparked the development of a plethora of computational methods and approaches that exploit network and interaction information between (genetic) entities, by making use of the existing interaction and network information in various ways. They can broadly be categorized into two different groups, namely those methods that leverage network information as a *post-processing* step to interpret results of genetic analyses, and those approaches that *integrate* network information into the knowledge generation process. Prominent examples of the first group are *gene-set enrichment analyses* [60], that associate, for example, individual genes to a trait, and subsequently test whether the detected genes are overrepresented in pre-defined gene sets, such as metabolic pathways. Furthermore, methods that aid the visualisation of results from genetic analysis by means of molecular networks pertain to the first group [61, 62]. The second class of methods incorporate the network information directly into the process of understanding complex traits, and use the network as an input data type. Those approaches mostly rely on gene-based summary statistics, such as association scores to the trait of interest, that are superimposed on the nodes in the network, followed by an exploration of the network. The result commonly are sub-modules of nodes that constitute interesting candidate genes for downstream analysis. Widely-adopted methods rely on a greedy exploration of the network [63, 64]. A graph-theoretical approach that recently gained a lot of attention is *network propagation* [65]. Methods based on this framework rely on the stepwise propagation of node features along shortest paths through the network. During the propagation, each node updates its feature according to the features in its local neighbourhood, such that nodes lying on shortest paths between high-scoring nodes will eventually end up with higher scored features themselves. Those nodes will be prioritized in follow-up analyses. Network propagation has been successfully applied to different problems, such as the prediction of protein function and the characterization of disease genes, and we refer the interested reader to the review by Cowen et al. [65].

In summary, the last decade has seen a steep increase in the availability of data describing biological networks in general, and of molecular networks in particular. They contribute to a holistic understanding of biological processes on various scales, and represent knowledge that has been aggregated from various sources and experiments. The networks themselves are a subject of extensive research, existing networks are continuously refined and extended, and novel networks are inferred using experimental data or computational inference. In combination with genetic data, those networks have shown to be instrumental for the interpretation of genetic analysis, as well as for the discovery of novel trait-associated genes and gene-sets. This thesis is dedicated to the development of novel methods to integrate molecular networks with genetic analyses in two different disciplines, namely genetic association studies, and the discovery of novel cancer driver genes.

1.3. Contributions of the thesis

The topic of this dissertation is the development of methods for the identification of novel genes associated to complex traits with the help of molecular networks. In Part I, we gave an extensive introduction to biological networks and their biomedical implications in general, and to the application of molecular networks for the discovery of trait-associated

1. The age of network biology

genes in particular. Part II and Part III introduce the main contributions of the thesis, that is the implication of molecular networks in genetic association studies, and in the discovery of cancer driver genes, respectively. Due to the diversity of those two topics, an outlook specific to each one can be found at the end of Part II and Part III. We conclude this thesis with a broader discussion of current trends and challenges in the field of graph-structured learning in Part IV.

Part II: Network-guided genetic association studies

Genetic association studies aim at discovering genes or other types of genetic variants that are associated to complex traits by analysing the DNA-sequence of individuals. While this led to the discovery of many genotype to phenotype associations, complex traits most often cannot be sufficiently explained by additive effects of individual genetic variants, and this phenomenon is referred to as the *missing heritability*. A common hypothesis to explain missing heritability is that classical association studies neglect non-linear interaction effects between multiple genetic variants that might confer an individual's susceptibility to the trait. However, addressing interaction effects between genetic variants constitutes a challenging undertaking. The sheer number of possible interactions creates an immense statistical burden in form of a multiple hypothesis testing problem, as well as a computational burden due to the enumeration of the vast search space. To this end, we propose two different methods to address those challenges and enable the search for non-linear interactions in the genome by combining genomic data with molecular networks. Furthermore, we leverage concepts from the field of significant pattern mining to alleviate the implicit statistical and computational limitations. Those contributions are explicated in Part II of the thesis at hand.

This second part starts with Chapter 2 which constitutes an introduction to classical genetic association studies. This includes a detailed description of the genetic data that is used throughout this part of the thesis, along with standard methods and approaches for their analysis. It furthermore discusses the limitations of those methods, and how biological networks represent an promising candidate data representation to resolve them. This chapter is based on, but not restricted to, the author's book chapter

Gumpinger, A.C., Roqueiro, D., Grimm, D.G., Borgwardt, K.M. *Methods and tools in genome-wide association studies* in *Computational Cell Biology* (2018), 93–136

As detailed above, to address the complex challenges imposed by the analysis of interactions between genetic variants, we turn to concepts and tools from the field of significant pattern mining. Hence, the subsequent Chapter 3 gives a self-contained introduction to the field of *significant pattern mining*. It introduces how concepts and techniques pertaining to significant pattern mining can be used to alleviate the statistical and computational challenges in interaction mining, and how they can be integrated with genetic association analyses. Importantly, this chapter is purely based on prior art.

Our first contribution is presented in Chapter 4. It is based on unpublished work

1. The age of network biology

Gumpinger, A.C., Llinares-López, F., Roqueiro, D., Borgwardt, K.M. *Network-guided association mapping of local neighbourhoods* (2019).

and describes a method called **tNeAT** (thresholded Neighbourhood Aggregation with Testing) to find sets of genes that form neighbourhoods in a biological network, and that are, upon combination, statistically significantly associated to a case-control phenotype of interest under model of genetic heterogeneity. It does so by exploiting concepts from significant pattern mining to handle the vast search space efficiently, and to correct for the resulting multiple hypothesis burden. The output of the method are local neighbourhoods in a gene-network, a fact that facilitates easier interpretation in downstream analyses. We designed a comprehensive simulation study, accounting for various network configurations, to measure performance and evaluate limitations of our method. Additionally, we apply **tNeAT** to a study of twenty phenotypes in the plant *Arabidopsis thaliana*.

Our second contribution to the realm of network-guided association studies is described in Chapter 5, which is based on the paper

Gumpinger, A.C., Rieck, B., Grimm, G.D., International Headache Genetics Consortium, Borgwardt, K.M. *Network-guided search for genetic heterogeneity between gene pairs*. In press at *OUP Bioinformatics* (2020).

In this paper, we describe a novel method called **SiNIMin** to find pairs of genes that (i) interact within a network, and (ii) are associated to a case-control phenotype under a model of genetic heterogeneity. Similar to the method presented in the previous chapter, it is based on principles of significant pattern mining to address computational and statistical challenges. In contrast to **tNeAT**, the way the network is integrated to guide the search for interactions varies strongly. In order to evaluate the performance of our method, we conducted a variety of simulated studies to illustrate the successful control of the type-I error, as well as its superiority to comparison partners with respect to statistical power and computational efficiency. Application of the method to 20 *Arabidopsis thaliana* phenotypes and a study of low-frequency variants for migraine showed the method's great potential to uncover novel gene interactions driving the development of phenotypes.

We end this second part of the thesis with an extensive summary of our contributions to network-guided association studies, followed by an outlook into promising future directions in Chapter 6.

Part III: Network-guided prediction of cancer driver genes

Part III of this thesis addresses a different topic. While still evolving around the integration of molecular networks with genetic data, the focus is shifted to cancer studies. To be more precise, the central project described in this part aims at the discovery of novel cancer driver genes by combining somatic mutation scores with molecular networks.

We start this part with an introduction to computational cancer genetics in Chapter 7, that elucidates the implication of genetic aberrations on the development and progression of cancer. Furthermore, available data resources are discussed, followed by an overview over established concepts and methods for the discovery of genes and mutations that presumable transform healthy cells into malignant tumours.

1. The age of network biology

Following this introduction, Chapter 8 presents our contribution

Gumpinger, A.C., Lage, K., Horn, H., Borgwardt, K.M., *Prediction of cancer driver genes through network based moment propagation of mutation scores*. In press at *OUP Bioinformatics* (2020).

In this project, we developed a novel node embedding approach for the supervised prediction of cancer driver genes based on mutation scores and molecular networks. As opposed to most existing methods, we approach the problem of cancer gene identification from a supervised, rather than an unsupervised perspective, by leveraging the knowledge about well-established cancer driver genes. We propose a novel representation of each node (gene) in a biological network, that is based on descriptors of its local neighbourhoods in the network, followed by a network propagation, a strategy we named *moment propagation embeddings*, short **MoPro** embeddings. Together with the set of well-established cancer driver genes, the **MoPro** embeddings can be subjected to a binary classifier. We developed a sophisticated cross-validation procedure to account for the class-imbalance and lack of a high-quality negative class, two challenges inherent to our problem definition. We apply our newly designed **MoPro** embeddings to a pan-cancer analysis of data from *The Cancer Genome Atlas* (TCGA). We show that the supervised formulation of the problem combined with **MoPro** embeddings achieves better performance than unsupervised approaches, and that the *knowledge contamination* inherent to protein-protein interaction networks is successfully addressed. We furthermore identified fourteen genes that were predicted to be cancer drivers, and that could not have been detected with the comparison partners. An extensive literature search indicated that those gene exhibited promising links to cancer, deeming them ideal candidates for further biological validation.

We end the chapter on cancer gene prediction with a summary of our proposed approach and findings, and present exciting future directions for the network-guided classification of cancer genes in Chapter 9.

Part IV: Discussion and outlook – learning from structured data

The final Part IV of this thesis is devoted to the discussion of recent advances in ‘learning from (graph-) structured data’. We start this part with a short diversion into another type of structured data, namely time-structure. In this context, we briefly outline a contribution of the author to a synthetic biology project, that entailed experimental design for time-resolved data, namely

Höllerer, S., Papaxanthos, L, **Gumpinger, A.C.**, Fischer, K., Beisel, C., Borgwardt, K, Benenson, Y., and Jeschek, M.. *Large-scale DNA-based phenotypic recording and deep learning enable highly accurate sequence-function mapping*. In press at *Nature communications* (2020).

We continue to summarise and discuss the contributions in this thesis, and conclude this dissertation with a discussion of trends and challenges in learning from graph-structured data.

Part II.

Network-guided genetic association studies

Context and overview

This second part of the thesis is devoted to network-guided association studies for complex traits. The first part introduced the main theme and the frame of the thesis, that is the implication of molecular networks in the discovery of genetic causes underlying complex traits. As such, it constituted a comprehensive overview over different types of biological networks, their implications in computational biology in general, and in genetic association studies in particular. In this chapter, we will focus on the latter, i.e. their implication in genetic association studies of complex traits. This is achieved by an integration of concepts from the realm of genome-wide association studies with biological networks, while formulating the task of finding network-guided associations as a significant pattern mining problem. This reformulation will prove itself useful, as significant pattern mining provides a flexible toolbox to address the statistical and computational challenges arising in network guided association studies.

To this end, we start this part of the thesis with an introduction to genetic association studies, that covers the most important ideas and methods (Chapter 2). We devote the last two sections of this introduction to the combination of genetic association studies with molecular networks. We continue with a comprehensive summary of the field of significant pattern mining in Chapter 3, and present concepts that will be at the very heart of our contributions. Both chapters present the background that builds the foundation of this part of the thesis, and both exclusively describe prior art. Our main contributions to the field of network-guided association studies can be found in Chapters 4 and 5. We end this part of the thesis with a summary and outlook into future directions of research in Chapter 6.

2. Introduction to genetic association studies

2.1. Deciphering the genetics underlying complex traits

Disentangling the genetic underpinnings of human diseases is among the big goals of biomedical research. It holds the promise of personalised medicine: estimation of an individual's susceptibility to disease long before the outbreak of the disease, enabling prevention, early interventions as well as personalised treatment [66]. The first successes in this field were achieved for monogenic diseases, that is diseases caused by mutations in single genes, by conducting *linkage analysis* on large-scale pedigrees [67]. Prominent examples of those efforts are the identification of loci associated to Huntingtons disease [68] and cystic fibrosis [69]. Despite those successes, genetic causes underlying *complex diseases* that presumably are co-determined by a multitude of genetic factors [70], such as type II diabetes, migraine, chronic obstructive pulmonary disorder (COPD), or migraine could not be detected with linkage studies. Only the mapping of the human genome [71], and community efforts such as the HapMap Project [72] or the 1000 Genomes Project [73] brought the field forward, enabling the stable inference of genetic causes underlying common diseases and ushering in the era of genome wide association studies (GWASs) [e.g. 74–76]. Ever since, GWASs have been instrumental in linking diverse disease phenotypes to genotypes [77–80].

Over the last decades, the advent of novel sequencing technologies combined with the decreasing cost of genomic sequencing [81] has led to an ever-increasing wealth of genomic data [82], both in terms of the number of sequenced variants, as well as sequenced individuals, substantially outpacing Moore's law [81]. Large biobank [e.g. 83–85] efforts aim at making genetic data available to researchers all over the world to enhance our understanding of the contribution of genetics to disease development and progression. While larger data sets that include more samples and more sequenced variants presumably contain novel insights, they also induce inherent statistical and computational challenges that have to be addressed, requiring the development of new methods to efficiently analyse those large-scale data sets.

This chapter gives a self-contained summary of the field of genome-wide association studies. The first part introduces classical (univariate) GWASs, and discusses the data types and data representations, followed by an overview over the standard methods to analyse them. A discussion of the limitations motivates the second part, which is the integration of GWASs using network information, and the most prevalent approaches to address this. The section ends with an outlook of what we consider important challenges to address next in the field of network-guided association studies.

2.2. Concepts and methods

Genome-wide association studies aim at deciphering the genetics underlying complex traits, that is traits that are presumably caused by a multitude of factors, as opposed to monogenic phenotypes. An important hypothesis underlying GWAS is the **common disease/common variant** one: diseases that are common in a population are supposedly caused by variants that are common in the population as well, and each of those variants contributes to the overall risk to develop the phenotype [86]. As opposed to rare variants, those common variants have lower effect sizes, and it is assumed that the joint effect of those common variants gives rise to complex traits [87]. While many GWASs discovered genetic variants that follow the common disease/common variant hypothesis, this does not imply that *all* common diseases are exclusively explained by common genetic variants [87]. However, the design of GWASs is laid out to discover common variation in the genome, in that the study object are genetic loci of common variation in the population.

This section provides an introduction to concepts and methods in GWASs, as well as their limitations. It continues to give an overview over novel approaches that combine GWASs with biological networks, in order to overcome some of the limitations inherent to GWASs, and ends with a motivational outlook for further analysis.

2.2.1. Genomic data

Genomic data describes the entirety of DNA pertaining to an organism, and is organised in only four letters: A, T, G, and C, corresponding to the four nucleic acids Adenine, Thymine, Guanine, and Cytosine. The haplotype human genome comprises more than three billion base pairs, and according to the 100 genomes project [73], each individual deviates from the reference genome at approximately 4.1 to 5.0 million base pairs. Interestingly, most genetic diversity is constrained to specific loci of common variation in the genome, so-called **single nucleotide polymorphisms**, short **SNPs**. There exist approximately 84.7 million SNPs in humans across all populations [73], making up $\sim 3.0\%$ of the human genome. They are the objects under study in a genome-wide association study. SNPs can be obtained for an individual either via next-generation sequencing (**NGS**, [88, 89]) or chip-based microarrays [reviewed in 90]. While NGS technologies enable sequencing of whole exomes or genomes, chip-based microarrays build upon the principle of **linkage disequilibrium (LD)** and **tag SNPs**: Linkage disequilibrium describes non-random correlations between SNPs throughout the genome. It arises due to different factors including recombination, genetic linkage, recombination, mutation, genetic drift, non-random mating, and population structure [90]. LD has important implications when capturing genetic variations in populations via genotyping arrays. Those arrays do not survey the complete entirety of all 84.7 million SNPs, but rely on choosing and genotyping a representative from each LD region, a so-called tag SNP. In case a tag SNP is found to be associated with the phenotype, it is assumed to be in high LD with the presumably causal SNP. With older populations having undergone more cycles of recombination, they exhibit shorter LD patterns than younger populations, necessitating the need for different tag SNPs and genotyping arrays. Identifying patterns of highly correlated SNPs throughout the human genomes for different ethnicities was one of the self-proclaimed goals of the International

2. Introduction to genetic association studies

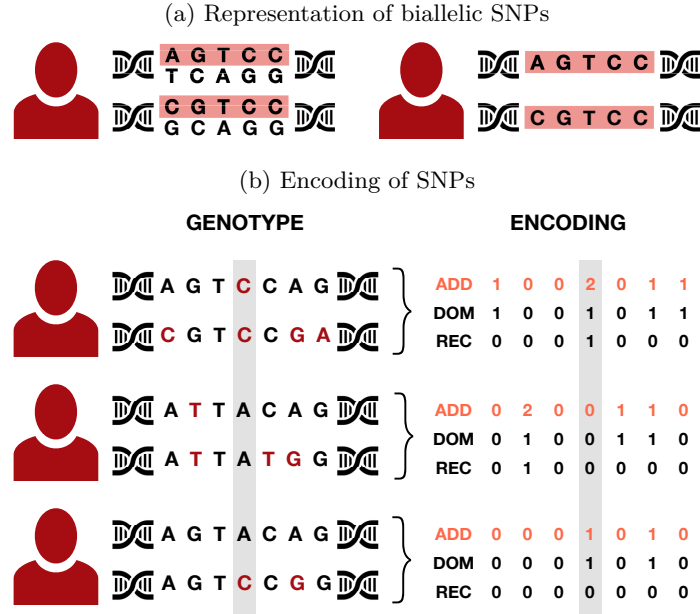


FIGURE 2.1.: Representation of single nucleotide polymorphisms (SNPs) in a genome-wide association study. (a) Each individual has two copies of each chromosome, resulting in biallelic SNPs. Since each nucleic acid pairs with its complement (A – T, C – G), only the allele on one DNA strand is considered. (b) The genotypes for three individuals. The minor alleles at each locus are highlighted in red. The right tables show the additive (ADD), dominant (DOM) and recessive (REC) encoding of SNPs with respect to the three individuals. (This figure is adapted from Figure 1 in Gumpinger et al. [92]).

HapMap Project [91].

This thesis focusses on diploid organisms, i.e. organisms that have two copies of each chromosome. As a consequence, each SNP can be represented by two (possibly different) alleles, corresponding to the bases on both copies of the chromosome (see Figure 2.1a). The allele that occurs more frequently in a reference population is referred to as the **major allele**, and the less frequent one inversely as the **minor allele**. The frequency of the minor allele is denoted as the minor allele frequency (**MAF**), and SNPs with minor allele frequencies below 1.0% are called rare variants. Rare variants are commonly removed in classical GWASs, as standard methods are underpowered to detect their effects, unless sample sizes are very large or the penetrance of the variants is very high.

In order to conduct a GWAS the data has to be represented in numerical form, as opposed to the letters A, T, G, and C. There exist different schemes for encoding the SNPs, and each of them emulates different prior assumptions about the underlying modes of inheritance. The most commonly used schemes for this encoding are (i) additive, (ii) recessive and (iii) dominant. In the additive case, an individual's SNP is represented as the number of minor alleles, resulting in values $\{0, 1, 2\}$ for each SNP. In the recessive encoding, a SNP is represented as 1 if both of its alleles are minor alleles, otherwise it is represented as 0. In the dominant encoding, a SNP is represented as 1 if at least one minor allele is present,

2. Introduction to genetic association studies

otherwise it is represented as 0 (see Figure 2.1b). Which of the encodings is chosen depends on the specific problem at hand, and should be considered when interpreting GWAS results [92].

2.2.2. Classical genome-wide association studies

The goal in classical genome-wide association studies is to find genetic variants that are associated to a phenotype or trait of interest. This phenotype can have various forms, with the most common ones being dichotomous/binary and quantitative/continuous. Dichotomous phenotypes arise mostly in so-called case-control studies, where individuals belong to one of two classes, for instance *affected by a disease (case)* or *healthy (control)*. Quantitative phenotypes, on the other hand, are measurable and live on a continuous scale. Prominent examples are height, blood pressure or weight. The nature of the phenotype in a genetic association study is a decisive factor for the type of method used in the study, as will be discussed in Section 2.2.2.3.

During the GWAS, the genomic data from multiple individuals and their corresponding phenotypes are pooled. A score of dependence between each genetic variant (SNP) and the phenotype is derived, and a p -value can be computed. It represents the probability of obtaining a more extreme score than the observed one under the null-hypothesis of no association. In general, the p -value of such a statistical test is considered to be significant in case it falls below a pre-defined significance level α , where commonly $\alpha = 0.01$ or $\alpha = 0.05$.

2.2.2.1. Multiple hypothesis testing in GWASs

In GWASs it is common to test hundreds of thousands to millions of SNPs for their association to a phenotype [87]. This implies that the same number of statistical tests have to be conducted, resulting in a **multiple hypothesis testing problem** that, if not properly accounted for, leads to the discovery of large numbers of false-positive, or spurious, associations. Especially if the SNPs detected in a GWAS should be evaluated in follow-up studies, avoiding large numbers of false positives is of utmost importance. For instance, if a significance threshold $\alpha = 0.05$ is chosen, the simultaneous testing of $t = 10'000$ hypothesis leads to $\alpha \times t = 500$ associations by random chance alone. A prevalent way to restrict the number of false positives is by controlling the so-called **family-wise error rate (FWER)**. It is defined as the probability to observe at least one false-positive association:

$$\text{FWER} = \mathbb{P}(\text{number of false positives} \geq 1) \quad (2.1)$$

If no correction for multiple hypothesis testing is applied, the FWER increases rapidly for low numbers of tests, indicating the presence of false-positive associations. This is illustrated in Figure 2.2. It is common practise in statistical hypothesis testing to control the FWER at the significance threshold α , that is to ensure that $\text{FWER} \leq \alpha$. A prominent way to achieve this is by applying the so-called **Bonferroni correction** [93]. It corresponds to testing every hypothesis at the Bonferroni threshold $\delta_{\text{Bonf}} = \alpha/t$, where t corresponds to the number of simultaneous tests. While the Bonferroni correction provably guarantees FWER control, it is known to be stringent for large numbers of tests: with an increasing

2. Introduction to genetic association studies

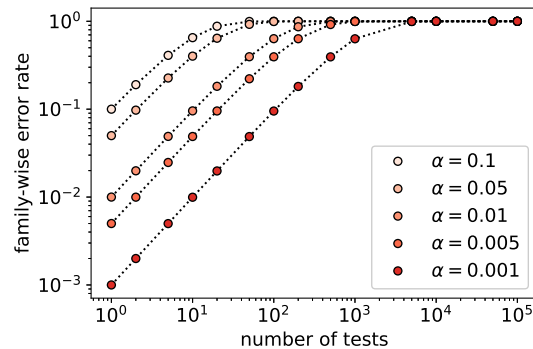


FIGURE 2.2.: Family-wise error rates for varying numbers of tests and varying values of the significance level α

correction factor t , the p -value a hypothesis must reach in order to be considered significant, decreases. Eventually, avoiding false positives is at the expense of potentially missing weak associations. An alternative way to control the number of false positives is to restrict the **false discovery rate (FDR)**. It is defined as the expected value of the false-discovery proportion, which in turn describes the fraction of false associations among all detected associations. Different methods exist to control the FDR [94–97], with the most widely used one being the Benjamini-Hochberg procedure [94].

Whether the FWER or the FDR should be controlled in a GWAS depends on the choice of design. If the FWER is controlled at level $\alpha = 0.05$, there is at most a 5.0% chance that one or more of the significant associations are false positives. If by contrast the FDR is controlled at level $\alpha = 0.05$, on average 5.0% of the significant associations might be false positives. Hence, for both methods there is a trade-off between the number of false positives and false negatives one is willing to accept.

2.2.2.2. Confounding factors

Another cause for false-positive associations is the presence of confounding factors, such as age, gender, population structure, or cryptic relatedness between individuals [98–100]. A confounder corresponds to a variable that is, for a given data set, correlated with the phenotype of interest, violating the assumption of independently distributed phenotypes across samples underlying many GWAS methods [100]. As a result, genetic variants that are associated to the confounder, not the phenotype, might become significant in a GWAS [99], as illustrated in Figure 2.3. Cryptic relatedness, for instance, describes the presence of hidden family structure in the data. This results in some individuals to share similar genetic variants, and probably also more similar phenotypes. If not accounted for, those genetic variants might bias the results of a genetic association study. Correction for confounding variables, if available, is indispensable to avoid spurious associations. Many models allow including covariates into the hypothesis testing, and especially mixed linear model have proven instrumental to address the problem of population structure in data.

2. Introduction to genetic association studies













SAMPLES		GENOTYPES										ORIGIN	
		1	0	0	0	0	1	0	0	2	1		
CASE	1	0	0	0	0	1	0	0	2	1			
	0	0	1	2	2	0	2	0	0	0			
	0	0	0	0	0	0	2	1	0	1			
	0	0	0	0	0	0	2	1	0	2			
	0	1	0	0	1	0	2	0	0	0			
	0	1	0	0	1	0	2	0	0	0			
CONTROL	0	1	0	0	1	0	0	1	2	0			
	0	0	1	0	1	0	2	0	0	1			
	0	0	1	1	1	0	0	0	0	0			
	0	0	1	0	0	0	0	0	0	0			
	0	0	1	0	0	0	0	0	0	0			
	0	0	1	0	0	0	0	0	0	0			

FIGURE 2.3.: Example of a genome-wide association study with a dichotomous case-control phenotype. Each row in the data set corresponds to one sample that is either classified to be a case, or a healthy control with respect to a disease. Each column corresponds to one genotyped SNP using an additive encoding. Additionally, the country of origin of each individual is given as a covariate, and each sample comes from one of two countries, also indicated by the colours in the data table. In a GWAS, each SNP is tested for its association to the phenotype. Two SNPs that are associated to the case-control phenotype are highlighted with a grey box. The right SNP constitutes a spurious association: it is actually associated to the covariate. The association to the phenotype vanishes if additionally a correction for the covariate takes place.

2.2.2.3. Statistical association testing in GWASs

There exist a multitude of statistical tests in GWASs, and the choice of test depends on the type of trait. Quantitative traits, such as height or body-mass-index, are mostly studied in linear model frameworks. Such frameworks understand the phenotype as a linear combination between the SNP and, if available, covariates such as age, gender or clinical covariates, and commonly include a residual noise term, modelling uncertainty in the data [92].

Especially the class of linear mixed models (LMMs) has proven to successfully correct for cryptic relatedness and population structure in GWASs [101–104]. Those models assume that the phenotype follows a continuous probability distribution (commonly a Gaussian), and explain the phenotype as a linear combination of fixed and random effects. For most applications, the offset, genotype and the covariates are modelled as fixed effects, while the relatedness between samples is modelled as a random effect. This emphasises the underlying assumption that the phenotypic variance is an additive combination of the genetic variance between individuals and normally-distributed residual noise. Mathematically, these models can be described as

$$\mathbf{y} \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma_g^2\mathbf{K} + \sigma_\epsilon^2\mathbf{I}) \quad (2.2)$$

where \mathbf{y} is the vector that holds the phenotypes for n samples, and $\mathcal{N}(\mu, \sigma)$ denotes the

2. Introduction to genetic association studies

TABLE 2.1.: Example of a contingency table to test the association between a SNP (represented by the occurrence of minor alleles) and a case-control trait. The columns indicate the number of minor alleles, the rows indicate the trait, i.e. case or control, n denotes the total number of samples in the dataset.

	number of MA = 0	number of MA ≥ 1	row total
case	a	$n_1 - a$	n_1
control	$z - a$	$n - n_1 - (z - a)$	$n - n_1$
column total	z	$n - z$	n

normal distribution with mean μ and standard deviation σ . Hence, from this formulation it becomes clear that the phenotype is assumed to follow a normal distribution. In the equation, the explanatory variables, i.e. the model bias, the (optional) covariates, and most importantly the genotype, are stored in the $n \times d$ -dimensional matrix \mathbf{X} . The variables stored in \mathbf{X} are also referred to as the *fixed effects*. The covariance of the normal distribution consists of two components, the $n \times n$ -dimensional matrix \mathbf{K} that stores genetic similarities between the n samples, and a residual variance component, included by the $n \times n$ -dimensional identity matrix \mathbf{I} . The d -dimensional vector $\boldsymbol{\beta}$ describes the weights of the fixed effects, and $\sigma_g^2, \sigma_\epsilon^2 \in \mathbb{R}$ describe the magnitude of the genetic and residual variance, respectively. Importantly, the LMM is highly flexible, in that other variance components can be added into the model [105]. The statistical association can be tested by conducting a likelihood ratio test, in which the likelihood of the model *including* the genotype is compared to the likelihood of the model *excluding* the genotype.

In the case of dichotomous/binary traits, the assumption of a normally distributed phenotype is violated. In order to still leverage the benefits of the LMM frameworks, instead of modelling the phenotype, the logarithm of the odds is modelled [e.g. 106], i.e. Equation 2.2 becomes

$$\log \left(\frac{\text{P}(y = 1)}{1 - \text{P}(y = 1)} \right) \sim \mathcal{N}(\mathbf{X}\boldsymbol{\beta}, \sigma_g^2 \mathbf{K} + \sigma_\epsilon^2 \mathbf{I}). \quad (2.3)$$

Alternatively, common methods applied for binary traits are two-sample tests that are based on contingency tables, such as Fisher’s exact test, Pearson’s χ^2 test [107] or the Cochran-Mantel-Haenszel test [108, 109], with the latter allowing the inclusion of categorical covariates. Those methods aim at testing whether the distributions of a SNP’s alleles are independent between the case and control class. While they are less flexible than linear mixed models, they have the advantage of being computationally efficient, as p -values can be computed in closed form from the contingency tables (see Table 2.1 for example).

2.2.3. Limitations

The importance of the contribution of GWASs to our current understanding of genetics underlying complex traits cannot be overstated: of February 2020, the GWAS-catalogue [110], a comprehensive resource of GWAS results, contained 175’870 associations from 4’439 pub-

2. Introduction to genetic association studies

lications spanning a multitude of phenotypes and traits. Despite those successes, GWASs still suffer from certain limitations, and we refer to Tam et al. [111] for a comprehensive review of benefits and limitations of GWASs. We address some of the most pressing ones in the following.

Identification of causal variants One major challenge in GWASs is the inference of causal relationships between genetic variants and a phenotype [111]. Linkage disequilibrium hinders the interpretation of GWAS results, as a SNP that is found to be significantly associated to a phenotype is not necessarily the causal variant, but might be linked to the causal variant — if there exists any causal relationship at all. Furthermore, most tag SNPs in genotyping arrays ($\sim 88.0\%$, [112]) lie in intronic or intergenic regions of the genome, resulting in many significantly associated SNPs that cannot directly be mapped to a gene. The interpretation of those associations, let alone the inference of causality, is inherently difficult and requires additional steps, such as re-sequencing, functional analyses or *in vitro* and *in vivo* verifications [112].

Another important concern in GWASs is that the genetic variants detected in those studies often only explain a small part of the phenotypic heritability, a phenomenon called the **missing heritability** [113]. Different hypotheses that aim at explaining missing heritability in GWASs exist: (i) the impact of rare variants [114, 115], (ii) the multiple hypothesis testing burden [111], and (iii) the existence of non-linear interactions between genetic loci [116].

Rare variants One potential explanation for the missing heritability is that rare variants of large effects substantially contribute to an individuals' risk to develop a disease [115]. A genetic variant is considered to be a **rare variant** if it is observed in less than γ percent of the samples, with γ commonly $\leq 1.0\%$. Those infrequent genetic variants cannot be detected with standard statistical tests used in GWASs given their low prevalence in the population [117]. In order to resolve the rare-variant challenge, a common approach is to combine rare variants based on their position in the genome, using different collapsing strategies [118], or specifically designed multi-marker tests [119].

Multiple comparisons problem A second potential cause explaining the missing heritability is the burden of controlling the number of false positives in statistical testing (see Section 2.2.2.1). GWASs often contain hundreds of thousands to millions of SNPs, and are hence affected by a severe multiple comparisons problem. Due to the large number of simultaneous tests in a GWAS, most of the tested variants do not reach the stringent significance threshold imposed e.g. by the Bonferroni correction [111]. A large part of the missing heritability might lie within variants that are only moderately associated to the phenotype.

Interactions between genetic loci A third potential cause explaining missing heritability is that GWASs analyse genetic variants in isolation, and do not consider non-linear interaction effects between multiple loci [116]. This hypothesis is at the core of **epistasis** research, a type of genetic analysis that is focussing on interactions between pairwise

2. Introduction to genetic association studies

genetic loci, such as single variants or genes [120–122]. The main idea behind epistasis research is that the effect of one genetic locus might be masked by another locus, resulting in a non-linear effect between the two. While addressing the challenge of incorporating interactions between genetic loci seems promising, it complicates the multiple comparisons problem described in the previous paragraph: by testing pair-wise interactions, the number of statistical tests corresponds to the square of the genetic variants to be tested, such that the chances that any interaction reaches the significance threshold is further reduced. To circumvent this, other directions of research approach the interaction search by exploiting biological prior knowledge in the form of molecular networks.

The last aspect, that is the analysis of interactions between genetic loci, has recently gained a lot of attention, and finding association signals beyond single variant analysis has become a research paradigm [123]. However, the space of potential interactions between genetic variants increases exponentially with the variants: the number of possible interactions between d SNPs equals 2^d , posing a severe computational and statistical challenge. Given the ever-increasing number of biological, and in particular molecular networks, novel methods emerged that are based on an integration of genetic data with the biological prior knowledge encoded in those networks [54]. The hypothesis underlying those methods is that multiple genetic variants interact and thereby lead to the manifestation of a phenotype. Those interactions do not take place between random variants, but follow the complex blueprint of cellular organisation that is captured in molecular networks. This deems some interactions more likely than others, and by restricting the search for interactions to those that are supported by the network, the search space of potential interactions is reduced drastically. This integration of biological networks transforms the problem of exploring all possible sets of interacting genetic variants to the problem of exploring sets of variants that form connected modules in a network. As opposed to evaluating all possible interactions, this approach gives rise to interpretable and meaningful variant sets. The approach to guide interaction analyses by molecular networks is further corroborated by the observation that genes implicated in the same phenotypes tend to cluster within the network [58]. Over the last decades, a variety of different approaches have been developed that are based on the integration of molecular networks with genetic data. The next section gives an overview over the main methodical concepts underlying those analyses.

2.3. An overview of network based genetic analysis

2.3.1. From SNPs to genes to networks

Since molecular networks are based on interactions between genes, combining them with SNP data requires a mapping of SNPs to genes. Commonly, SNPs are mapped to genes by proximity, i.e. genes are represented by those SNPs that overlap with them. Different approaches exist for this mapping, such as mapping all SNPs to a gene that lie within its coding region (exon), or mapping all SNPs that lie either in the exons or introns of a gene. Furthermore, it is common to also map SNPs to genes that lie within a certain number of bases up- and downstream of the genes, sometimes referred to as a window around the gene [125–127]. This window can be symmetrical or asymmetrical, and accounts for mutations that might affect the binding of regulatory elements, such as transcription

2. Introduction to genetic association studies

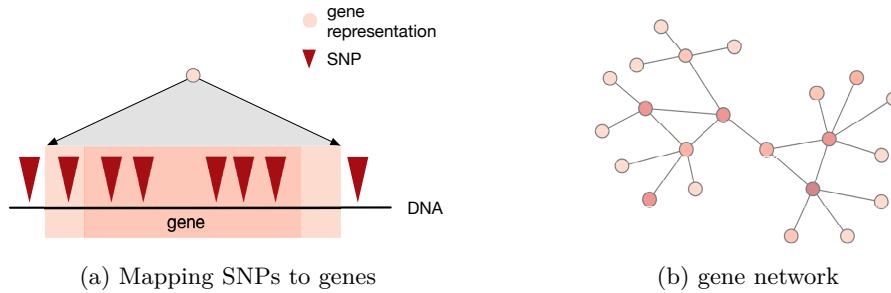


FIGURE 2.4.: Integration of genetic data with biological networks. (This figure is adapted from Figure 1 in Gumpinger et al. [124])

factors (see Figure 2.4a). It is important to be aware that an increased window around genes might result in mapping a single SNP to multiple genes, which might lead to an inflation of the results [128].

The mapping of SNPs to genes is followed by the derivation of a scalar representation of the genes. In the context of genome-wide association studies, this is mostly done in form of a p -value of association to the phenotype of interest, or a test statistic. Different methods exist, how a gene-wise p -value can be computed, and they can be mainly grouped into two classes [129]: (i) two-step approaches, that first compute univariate, SNP-wise p -values and combine them in a second step to a gene-based statistic, or (ii) one-step approaches, where gene p -values are computed by jointly analysing all SNPs in the gene.

The two-step methods start by computing a test statistic and p -value of association for every individual SNP mapped to the gene. The second step consists of the generation of a gene-based test statistic and/or p -value from the SNP-based ones. Different schemes to achieve this exist, ranging from very simple aggregations, such as picking the most significant p -value within a gene as a representative [127, 129], or using the sum or average of χ^2 statistics of the SNPs overlapping with the gene [126, 127]. Alternatively, all SNP p -values in a gene can be combined, for instance with Fisher’s method [130]. Some of those statistics are confounded by the size of the gene, as well as the LD structure (for example when picking the lowest p -value within a gene, the probability to observe low p -values increases with increasing gene sizes). It is important to correct for this confounding in order to obtain unbiased gene representations [127].

The one-step approaches do not test each SNP individually, but consider all SNPs mapping to a single gene as a set, and test them jointly [104, 131, 132]. Those so-called set-tests have been applied successfully to identify genes associated to various diseases, and especially for the analysis of rare variants, set-tests show increased power when close-by rare variants are aggregated compared to testing rare variants in isolation [132]. There exist different regimes of how variants can be tested jointly. One way is to represent all SNPs that are mapped to a gene in a data matrix and test them jointly. For example the FastLMM-set method [104] achieves this by extending the linear mixed model in Equation 2.2, i.e. the design matrix \mathbf{X} contains all SNPs that map to the gene. Another option is to first collapse all SNPs in a gene into an n -dimensional vector, and regress this representation onto the phenotype. This is the mode of analysis in so-called burden tests [133–135].

2. Introduction to genetic association studies

After this step, each gene can be represented by a scalar value, that is either a p -value or a test statistic. We refer to the representation of a gene in the following very generally as a *gene score*. The last step is the mapping of the gene-scores to the network. This is done by superimposing the gene scores to the network of interest. This results in a gene-gene network, where each gene is represented by a score that measures the relevance of a gene in the phenotype of interest (see Figure 2.4b).

Those network representations build the basis of many network-based approaches to GWAS. There exist various ways how they can be used to discover novel disease-associated genes, ranging from greedy approaches, to graph regularisation approaches, to network propagation. We start by introducing some notation used throughout this section, and continue to present major ideas and concepts at the core of network guided genetic analysis methods.

Notation

We describe a network, or graph, as $\mathcal{G} = (V, E, \mathbf{W})$, where V denotes the set of d vertices, and E the set of m edges between them, and write (v, u) to denote an edge between two vertices $u, v \in V$. Hence, the edge set E can be written as $E = \{(u, v) \mid u, v \in V\}$. The matrix \mathbf{W} is the, potentially weighted, $d \times d$ adjacency matrix. The entries contain the edge weights, that for instance could describe the confidence of the edge. Without loss of generality, we assume the weights to be normalised to the range between 0 and 1, with 0 denoting absence of an edge, and 1 indicating high confidence of the edge. We denote this edge weight as $\omega(u, v) \in [0, 1]$, $u, v \in V$. In the absence of weighted edges, the matrix \mathbf{W} corresponds to the binary adjacency matrix of the network. We furthermore assume that every node $v \in V$ has a scalar feature $x_v \in \mathbb{R}$ assigned to it, which is also referred to as the score of the node.

2.3.2. Finding subgraphs enriched in low p -value genes

One group of methods that incorporate networks with genomic data are focussed on finding subgraphs that are enriched with low p -value genes, also denoted as modules. This is related to the maximum-weight connected subgraph (MWCS) problem, which is known to be NP-hard [136]. Different strategies to solve the problem have been introduced [63, 137–139], for instance based on a greedy exploration of the search space [63] or simulated annealing [137]. While greedy approaches take the locally optimal decision at each step [140], simulated annealing based approaches allow non-optimal decisions with certain probabilities. The results found with both strategies might only result in local optima that deviate from the global optimum, but have proven very useful for the exploration of genetic networks.

A prominent greedy approach to explore networks is **dmGWAS** [63]. It aims at finding subgraphs in the network that are connected and enriched in low p -value genes. For this purpose, **dmGWAS** applies a greedy strategy that is based on iterating through all d genes in the network, and greedily growing subgraphs at each gene until a stopping criterion is met. In total, d potential subgraphs are evaluated, hence between 0 and d potential subgraphs can be detected. Due to the greedy nature of the exploration, detected subgraphs commonly exhibit high similarity, as subgraphs rooted at close-by

2. Introduction to genetic association studies

vertices are often overlapping. A further development, termed edge weighted dmGWAS (`ew_dmGWAS`) [64] enabled the incorporation of edge weights into the approach, that e.g. could be inferred from gene expression data.

The `jActiveModules` method [137], implemented as a Cytoscape [141] plug-in, finds the maximum-weighted connected subgraph via simulated annealing [142]. It uses simulated annealing to determine active versus inactive genes in a network, and the final reported subnetwork is the largest connected component consisting of active nodes in a network. To achieve this, each node is assigned to the active/inactive class with probability 50.0%, followed by toggling the state of random nodes. In case this leads to a higher-scoring subnetwork, the toggled state is kept, in case the score for the subnetwork decreased, the score is kept with a probability depending on the annealing temperature and the score-delta. While initially developed for gene expression data [137], `jActiveModules` can readily be applied to any study in which genes can be represented by p -values.

2.3.3. Network propagation methods

Another concept that has proven itself valuable for the discovery of disease genes are methods based on graph diffusion, also known as network propagation [65]. Prominent methods, such as `HotNet` [143, 144], `HotNet2` [145] and `hierarchical HotNet` [146] are based on network propagation. While they were initially developed to discover previously unknown cancer genes, those methods were successfully applied to other types of data as well, including genome-wide association studies [20, 147]. Network propagation methods represent each gene in the network with a score (see Section 2.3.1), and propagate those scores through the network. During propagation, each vertex’s score is updated based on the scores in its local neighbourhood. Conceptually, this can be best explained by interpreting the node score as an amount of fluid, heat or information, that is assigned to the node. In each propagation step, nodes pass their heat to, and receive a certain amount of heat from their neighbours. This is repeated, until a steady state of the vertex scores is reached, corresponding to the final state of the network.

Mathematically, this can be described as follows: if the score of vertex v at propagation step t is denoted as $x_v^{(t)}$, the score of v is updated at every propagation step as

$$x_v^{(t+1)} = \sum_{e(u,v) \in E} \omega(u,v)x_u^{(t)}. \quad (2.4)$$

This can equivalently be written in matrix notation as

$$x^{(t)} = \mathbf{W}^{(t)}x^0, \quad (2.5)$$

where $x^{(0)}$ is the initial state of the network [65], and \mathbf{W} is the $d \times d$ -dimensional (weighted) adjacency matrix. Another concept related to network propagation are random walks with restart (`RWR`), mathematically described as

$$x^{(t+1)} = \alpha x^{(0)} + (1 - \alpha)\mathbf{W}x^{(t)}, \quad (2.6)$$

where $\alpha \in [0, 1]$. That is, at every time step, the state of the vertex will return to its

2. Introduction to genetic association studies

original value with probability α . For connected networks, and weighted adjacency matrix \mathbf{W} with eigenvalues smaller than or equal to 1, the steady state distribution of node weights converges to

$$\bar{x} = \alpha(\mathbf{I} - (1 - \alpha)\mathbf{W})^{-1}x^{(0)}. \quad (2.7)$$

A common way to use the steady state distribution is as a criterion to rank genes [65], and prioritise top-ranked genes in follow up analysis.

A recently developed method called **uKIN**, short for **using Knowledge In Networks**, [148] incorporates prior knowledge in the form of known disease genes with novel information, e.g. from genome-wide association studies, by implementing a guided random walk with restart. It has shown to improve the classification of cancer driver genes for multiple cancer types, and has been applied to GWASs of age-related macular degeneration, amyotrophic lateral sclerosis and epilepsy.

Other approaches, such as **hierarchical HotNet** [146] use random walks with restart in the network, and construct a similarity matrix between nodes from the resulting steady-state distribution. This similarity matrix serves as the input to a hierarchical clustering of nodes, giving rise to ‘interesting’ subgraphs. Different permutation schemes are applied to test the significance of the *size* of the observed largest cluster, resulting in an empirical p -value. Ruffalo et al. [149] suggest the separate propagation of mutation and gene-expression indicators through a network using random walks with restart, and subsequently generate features from the resulting steady state distribution to predict the disease-causing status of genes.

2.3.4. Using networks as regularisers

Linear models have been used extensively to model the additive effect of genetic markers, such as SNPs or genes, on a phenotype of interest. Mathematically, those models can be expressed as

$$\begin{aligned} \mathbf{y} &= \beta_0 + \mathbf{x}_1\beta_1 + \dots + \mathbf{x}_d\beta_d \\ &= \mathbf{X}\boldsymbol{\beta} \end{aligned} \quad (2.8)$$

for an n -dimensional phenotype vector \mathbf{y} and d genetic features, denoted with \mathbf{x}_i , $i = 1, \dots, d$. Importantly, those models do not use a summary-statistic representation of the genes in the network as described in Section 2.3.1, but are n -dimensional vectors describing the feature value for each of the samples. For ease of notation, we use the matrix notation, that is an $n \times (d + 1)$ -dimensional design matrix $\mathbf{X} = [\mathbf{1}, \mathbf{x}_1, \dots, \mathbf{x}_d]$, and $\boldsymbol{\beta} = [\beta_0, \dots, \beta_d]^T \in \mathbb{R}^{d+1}$. In order to determine the best coefficients β_i for $i = 0, \dots, d$, a loss function

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_f \quad (2.9)$$

that measures the model error is minimised with respect to a norm $\|\cdot\|_f$. Those models can be extended to incorporate molecular networks by including regularisation terms into the loss function.

The **group Lasso** [150] is a basic method that can be used to include network information,

2. Introduction to genetic association studies

by grouping genetic markers according to subnetworks in a molecular network. In order to achieve this, the **group Lasso** extends the loss in Equation 2.9 with a regularisation term that jointly selects grouped variables. Therefore, variables have to be grouped together prior to the analysis into J groups, and the loss-function becomes [151]:

$$\mathcal{L}_{\text{group}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_f + \lambda \sum_{j=0}^J \|\boldsymbol{\beta}_{[j]}\|. \quad (2.10)$$

where $\boldsymbol{\beta}_{[j]}$ are the β values belonging to the markers in the j -th group. However, the **group Lasso** assumes non-overlapping groups, an assumption that is often violated. A variation of this group Lasso is the so-called **overlapping group Lasso** [152, 153], that selects features that belong to the union of few feature groups. This enables the incorporation of network knowledge by defining those groups as node-pairs (i.e. edges) in the network, giving rise to the so-called **graph Lasso**. However, the combinatorial explosion of possible subnetworks in large molecular networks hinders the exhaustive exploration of all subnetworks.

Li & Li [154] developed a network-constrained regularisation that extends the loss function $\mathcal{L}(\boldsymbol{\beta})$ in Equation 2.9 with two regularisation terms:

$$\mathcal{L}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_f + \lambda_1 \|\boldsymbol{\beta}\|_1 + \lambda_2 \boldsymbol{\beta}^T \mathbf{L} \boldsymbol{\beta}. \quad (2.11)$$

The L_1 regularisation term induces a sparse solution with parameter $\lambda_1 \in \mathbb{R}$, while the second regularisation term gives a smooth solution of the coefficients over the network by inclusion of the graph Laplacian \mathbf{L} with parameter $\lambda_2 \in \mathbb{R}$. The graph Laplacian is defined as

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}, \quad (2.12)$$

where the matrix \mathbf{D} is a diagonal matrix containing the degrees of the nodes, and \mathbf{W} is the weighted adjacency matrix. The loss function in Equation 2.11 leads to neighbouring nodes in the network having similar coefficients if they are correlated [see Theorem 1 in 154]. While the method was originally applied to analyse gene expression data, it can be readily extended to discrete SNP data, where each feature in the design matrix \mathbf{X} can for instance represent a gene, and count the number of minor alleles in that gene.

A slightly different, yet related, approach was suggested by Azencott et al. [155]. They developed a method called **SConES**, short for **Selecting Connected Explanatory SNPs**. As the name suggests, the method was originally developed for SNP-SNP networks, rather than molecular networks where nodes correspond to genes. While the method is agnostic to the type of underlying data, it is able to deal with large-scale SNP-SNP networks that might contain millions of nodes. Given a SNP-SNP network, that for instance can describe connections between neighbouring SNPs, SNPs within genes, or SNPs within interacting genes, the method aims at identifying a minimal set of SNPs that is associated to the phenotype, sparse (with respect to the set of all SNPs), and contains SNPs that are connected in the underlying network. This can be formulated as an optimisation problem, and Azencott *et al.* show that this problem can be solved efficiently by interpreting it as

a minimum cut problem.

2.4. New directions in network based genetic analyses

The aforementioned list of methods is far from complete, but should give an idea of existing directions and approaches of how networks can be integrated with genetic analyses. Without any doubt, those methods have shown successes and led to the discovery of novel disease associated genes and molecular mechanisms. Two interesting aspects presumably have high potential to further advance the field of network-guided association studies. The first one concerns the mode of interaction between genetic entities, the second one the representation of data in molecular networks. As outlined in the remainder of this Part II of the thesis, addressing those two challenges poses important statistical and computational challenges. Interestingly, a seemingly unrelated field, that is the field of significant pattern mining, holds the tools and concepts necessary to deem those analyses feasible.

Genetic heterogeneity as an alternative mode of interaction

There exist different hypothesis how genetic entities, such as SNPs or genes, interact with each other. For instance linear models mostly adopt an additive perspective, i.e. the effect of an interaction is the sum of the effects of its individual participants. However, there exists another mode of interaction, namely **genetic heterogeneity**. It refers to an interaction model in which different genetic variants affect a phenotype in a similar way [156]. On a molecular level, this is corroborated by the ability of cells to rewire processes to maintain homeostasis in the presence of perturbation, such that non-linear interaction effects of perturbations cannot be detected by looking at the additive effects of individual variants [57]. Non-linear interactions in the form of genetic heterogeneity are assumed to play an important role in different diseases and phenotypes, with cancer being a prominent example [157]. In the field of complex disease research, genetic heterogeneity presumably underlies diseases caused by multiple rare-variants [156]. Developing models that analyse the effect of genetic heterogeneity promise to enable new insights into genetic interactions underlying complex traits.

From summary statistics to SNP-based representations of genes

Most approaches for network guided association studies rely on summary statistics to represent genes in networks, as described in Section 2.3.1. This representation corresponds to a compression of all information that is available for a single gene into one summary statistic. Especially for large genes that are represented with hundreds of SNPs, this potentially leads to an unwanted loss of information. Assuming that there exists a small subunit of the gene that has a high effect on the phenotype, this effect might remain undiscovered, as the majority of SNPs in the gene mask the effect. We believe that representing a gene as a set of SNPs rather than a summary statistic holds potential to increase the power of network guided association studies.

The role of significant pattern mining in the discovery of biomarkers

Recent advances in the field of significant pattern mining lead to the development of statistical methods that address the computational and statistical challenges associated to the testing of large numbers of interactions in binary transaction databases [158–163]. While this may seem like an unrelated topic at first, different biological problems, such as the search for genetic interactions between consecutive SNPs in the DNA [164, 165], can be reformulated to fit the framework of significant pattern mining, and allowing to leverage sophisticated methods and algorithms from this field of research. This enabled analysis that were previously not possible due to statistical and/or computational limitations, and resulted in an increase in the power to discover novel genetic biomarkers [166]. The next section introduces the theoretical foundations of significant pattern mining, and presents how those concepts can be integrated into the analysis of genetic interactions.

3. Introduction to significant pattern mining

3.1. Significant pattern mining

Significant pattern mining, sometimes also referred to as discriminative itemset mining, is a rather new approach in machine learning research, that combines concepts from the fields of *association rule mining* and *frequent pattern mining* principles with statistical association testing [166]. While the goal in frequent pattern mining and association rule mining is to detect patterns in data that follow certain rules or frequency constraints, significant pattern mining assumes the existence of two or more classes, and the goal is to find patterns that occur significantly more often in one class as opposed to the other classes.

The field of pattern mining originated from the task to analyse shopping baskets [167]: In shopping basket analysis, the data describes shopping baskets and the items they contain. The goal of frequent pattern mining is to find patterns, for example sets of items, that are *frequently* bought together. For instance, if the items {onions, leek} are contained in more than 50.0% of shopping baskets, it is considered a frequent itemset at any user-defined threshold $\leq 50.0\%$. Association rule mining evolves around detecting rules of the form $a \Rightarrow b$, for example 20.0% of individuals that bought onions and leek also bought carrots, in which case the rule is described as {onions, leek} \Rightarrow {carrots}. As for frequent itemset mining, the threshold at which the association rule is considered to be of interest (in the example 20.0%), is user-defined.

In contrast to frequent pattern mining and association rule mining, significant pattern mining assumes the existence of an additional binary class label of the data, such as whether the shopping basket belongs to a vegetarian or a meat-eater. The goal is to identify patterns that occur significantly more often in one class versus the other class. For example for the vegetarian versus meat-eater example, one probably finds the set {chicken, onion, leek} more often in the class of meat-eaters than in the class of vegetarians. The degree to which this more frequent occurrence is present in the data set is quantified using statistical testing.

This chapter is devoted to an introduction to significant pattern mining, and exclusively summarises prior art. The chapter is based on the doctoral dissertation by Llinares López [168], and the review by Llinares-López & Borgwardt [166] which offers a comprehensive review of the field. We start by introducing the notation used throughout this chapter, give a formal definition of the problem, and explain the main challenges in significant pattern mining. In the remainder, we explain the concepts and ideas underlying the solutions to those challenges. We end with an overview of how significant pattern mining have previously been exploited to enhance analyses in the field of biomedical research.

3.2. Notation and problem statement

Consider an index-set $\mathcal{I} = \{1, \dots, n\}$ indexing n samples, and an index-set $\mathcal{J} = \{1, \dots, d\}$ indexing d discrete items, also referred to as features. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ be a data set that contains n samples. Each sample i consists of a **transaction** $x_i \subseteq \mathcal{J}$, i.e. a finite subset of \mathcal{J} , and a binary class label $y_i \in \{0, 1\}$. For example, $x_i = \{3, 5, 7\}$ indicates that the 3rd, 5th and 7th feature are present for the i^{th} sample. Such a data set is in the pattern mining literature often referred to as a **transaction database**. Alternatively, the data can also be represented as a binary matrix $\mathbf{X} \in \{0, 1\}^{n \times d}$ that describes the presence or absence of d features for each of n samples, such that $\mathbf{X}_{i,j} = 1$ indicates that feature j is present for sample i , where $j \in \mathcal{J}$, $i \in \mathcal{I}$. The class assignment can be stored in an n -dimensional binary label vector $\mathbf{y} \in \{0, 1\}^n$. Every discrete substructure \mathcal{S} of the index-set \mathcal{J} is called a **pattern**, $\mathcal{S} \subseteq \mathcal{J}$. By definition, patterns exhibit sub- and super-pattern relationships. If $\mathcal{S} \subseteq \mathcal{S}'$, we refer to \mathcal{S} as a sub-pattern of \mathcal{S}' and inversely to \mathcal{S}' as a super-pattern of \mathcal{S} . We furthermore define a **pattern indicator function** $g_{\mathcal{S}} : x_i \rightarrow \{0, 1\}$ that indicates whether a pattern \mathcal{S} is present in sample i .

The goal of significant pattern mining is to find patterns \mathcal{S} that occur significantly more often in one of the two classes. More formally, for a fixed pattern \mathcal{S} we can evaluate the pattern indicator function $g_{\mathcal{S}}$ for every sample, resulting in an n -dimensional binary vector that indicates the presence or absence of the pattern. We denote the vector that contains the evaluation of the function $g_{\mathcal{S}}$ for all samples $i \in \mathcal{I}$ as

$$\mathbf{g}_{\mathcal{S}} = [g_{\mathcal{S}}(x_1), \dots, g_{\mathcal{S}}(x_n)] \in \{0, 1\}^n. \quad (3.1)$$

The sum over this vector, i.e. the number of occurrences of the pattern, is referred to as its **support**. Together with the label vector \mathbf{y} , a contingency table can be derived, which counts the number of patterns in both classes (see Figure 3.1b). This contingency table can be analysed with a discrete statistical test, such as Fisher’s exact test, to obtain a p -value of association between the labels and the occurrence of the pattern. Such a test is useful in evaluating whether the support of the pattern is distributed equally across both classes in the dataset.

On the pattern indicator function $g_{\mathcal{S}}$

The choice of the function $g_{\mathcal{S}}$ depends on the problem, and there exist different options. For instance, in classical frequent pattern mining, a pattern is considered to be present in a transaction if it is fully contained in the transaction. We evaluate this with the *minimum* pattern indicator function $g_{\mathcal{S}}^{\min}(\cdot)$ that takes the following form:

$$g_{\mathcal{S}}^{\min}(x_i) = \begin{cases} 1 & \text{if and only if } \mathcal{S} \subseteq x_i \\ 0 & \text{else.} \end{cases} \quad (3.2)$$

The *minimum* in the name is inspired by the matrix notation of the problem, where evaluating the function corresponds to taking the minimum over features in the pattern. Another choice for the pattern indicator function that has been successfully applied in the literature [164, 165] is the *maximum* pattern indicator function $g_{\mathcal{S}}^{\max}$, that measures

3. Introduction to significant pattern mining

whether there is any overlap between a pattern \mathcal{S} and the transaction x_i , that is

$$g_{\mathcal{S}}^{\max}(x_i) = \begin{cases} 1 & \text{if and only if } \mathcal{S} \cap x_i \neq \emptyset \\ 0 & \text{else.} \end{cases} \quad (3.3)$$

Analogously to the minimum indicator function, the *maximum* in the name is inspired by the matrix notation of the problem, where the function can be evaluated by taking the maximum over the features in the pattern. An example of the maximum pattern indicator function is illustrated in Figure 3.1a.

Alternatively, an approach that relies on choosing thresholds could be used, for example

$$g_{\mathcal{S}}^{\text{thres}}(x_i) = \begin{cases} 1 & \text{if and only if } \frac{|\mathcal{S} \cap x_i|}{|\mathcal{S}|} \geq \tau \\ 0 & \text{else,} \end{cases} \quad (3.4)$$

where τ is a user-defined threshold. However, this reliance on a user-defined threshold can be challenging, as choosing a suitable threshold τ apriori is not a trivial task.

Remark 3.2.1 (On the use of pattern indicator functions). *While the minimum pattern indicator is the most widely-used function in classical frequent pattern mining, the methods developed in this thesis use the maximum pattern indicator $g_{\mathcal{S}}^{\max}$. This choice is motivated biologically, and will be explained in further detail in the chapters describing the contribution of this thesis. For this reason, the remainder of this chapter is written to support the maximum encoding. This implies that some descriptions of the methods that were originally used with the minimum indicator had to be reformulated, and hence do not exactly follow the description in the original publications. It is important to note that all methods only require minor reformulations to be used with the minimum pattern indicator.*

Remark 3.2.2 (Notation). *For notational simplicity, we omit the \max and write $g_{\mathcal{S}}$ to mean $g_{\mathcal{S}}^{\max}$, unless explicitly stated otherwise.*

Challenges in significant pattern mining

The space of all possible patterns is called the **search space** or **hypothesis space**, denoted by \mathcal{H} . Its size depends on the type of patterns to be analysed. The most general instance of significant pattern mining is so-called significant itemset mining, where all possible patterns are explored. Hence, the search space consists of all possible subsets of the feature-index set \mathcal{J} , denoted with the power set $\mathcal{P}(\mathcal{J})$ of the feature-indices. The number of possible itemsets scales exponentially with the number of features: there are a total of $2^{|\mathcal{J}|}$ such combinations. Already small data sets with only a few hundred features thus give rise to an enormous search space. To give an example [taken from 166], for a data set with $d = 266$ features, the search space in itemset mining consists of approximately 10^{80} higher-order feature combinations, corresponding to the number of electrons estimated to be in the observable universe. Hence, an exhaustive exploration of the search space poses two challenges:

1. a *statistical* challenge, caused by the excessive number of patterns that have to be tested, imposing a severe multiple hypothesis testing burden in order to avoid the

3. Introduction to significant pattern mining

(a) Data set and pattern representation

		FEATURES										\mathbf{g}_S^{\max}	\mathbf{y}
		1	0	0	0	0	1	0	0	1	1		
SAMPLES	CASE	1	0	0	0	0	1	0	0	1	1	1	1
		0	0	1	1	1	0	1	0	0	0	1	1
		0	0	0	0	0	0	1	1	0	1	0	1
		0	0	1	0	0	0	1	1	0	1	1	1
		0	1	1	0	1	0	1	0	0	0	1	1
		0	1	0	0	1	1	1	0	0	0	1	1
SAMPLES	CONTROL	0	1	0	0	1	0	0	1	1	0	1	0
		0	0	0	0	1	0	1	0	0	1	0	0
		0	0	0	1	1	0	0	0	0	0	0	0
		0	0	0	0	0	0	0	0	0	0	0	0
		0	1	0	0	0	0	0	0	0	0	0	0
		0	0	0	0	0	0	0	0	0	0	0	0

(b) Contingency table

	$\mathbf{g}_S = 1$	$\mathbf{g}_S = 0$	total row
$\mathbf{y} = 1$	a_S	$n_1 - a_S$	n_1
$\mathbf{y} = 0$	$z_S - a_S$	$n - n_1 - (z_S - a_S)$	$n - n_1$
column total	z_S	$n - z_S$	n

FIGURE 3.1.: Example of significant itemset mining. (a) Data set for $n = 12$ samples and $d = 10$ features. The itemset to be tested contains the third, the sixth and the ninth item. The pattern is considered to be present, if at least one of the features is 1 for a sample, corresponding to the maximum pattern indicator function \mathbf{g}_S^{\max} in Equation 3.3. (b) Contingency table derived from comparing the label vector \mathbf{y} with the maximum pattern indicator function \mathbf{g}_S^{\max} . The counts according to (a) are $n = 12$, $n_1 = 6$, $z_S = 6$ and $a_S = 5$.

discovery or large numbers of false positive associations, and

2. a *computational* challenge, as all patterns pertaining to the vast search space have to be enumerated.

In order to make significant pattern mining feasible, both challenges have to be addressed. Decades of research went into resolving the second challenge, and led to the development of efficient data mining algorithms that are based on sophisticated data representations and pruning schemes [e.g. 169–178]. Only recently, novel approaches emerged that targeted the statistical challenge, which is of utmost importance when restricting the number of false positive associations has a high priority [158–163].

3.3. Association testing and the multiple comparison problem

As described in the previous section, significant pattern mining is based on a statistical test of association between a pattern \mathcal{S} and the binary class label. We will adopt the vector-based representation of the data here, such that the task of statistical association testing becomes a task of testing the association between two binary vectors, one of them being the label vector \mathbf{y} , that assigns each sample to one of two classes, and the other one the pattern indicator $\mathbf{g}_{\mathcal{S}}$, as illustrated in Figure 3.1. It is important to note that this representation of a pattern is agnostic to the underlying pattern type, i.e. whether the pattern is an itemset or any other predefined sub-structure of the feature index-set \mathcal{J} does not affect the representation in a contingency table.

Those vectors are assumed to be draws from the distribution of two unobserved random variables, Y and $G_{\mathcal{S}}$, and the two variables are said to be statistically associated, if they are not independent from each other, i.e. if their joint probability distributions $\mathbb{P}(Y, G_{\mathcal{S}})$ does not factorise as $\mathbb{P}(Y)\mathbb{P}(G_{\mathcal{S}})$. In an ideal scenario, this criterion would be checked.

In practice however, the probability distributions underlying the data are unknown, such that independence cannot be assessed. Instead, the binary vectors \mathbf{y} and $\mathbf{g}_{\mathcal{S}}$ are assumed to be independent, identical draws from the underlying data-generating distributions, and the joint probability distribution can be approximated by counting the four configurations of the random variables, that is $(\mathbf{y}_i = 1, \mathbf{g}_{\mathcal{S},i} = 1)$, $(\mathbf{y}_i = 1, \mathbf{g}_{\mathcal{S},i} = 0)$, $(\mathbf{y}_i = 0, \mathbf{g}_{\mathcal{S},i} = 1)$ and $(\mathbf{y}_i = 0, \mathbf{g}_{\mathcal{S},i} = 0)$. Those are the counts represented in the 2×2 contingency table (see Figure 3.1b), such that all those approximations can be read from the table as

$$\begin{aligned} \mathbb{P}(\mathbf{y}_i = 1, \mathbf{g}_{\mathcal{S},i} = 1) &= \frac{a_{\mathcal{S}}}{n} \\ \mathbb{P}(\mathbf{y}_i = 1, \mathbf{g}_{\mathcal{S},i} = 0) &= \frac{n_1 - a_{\mathcal{S}}}{n} \\ \mathbb{P}(\mathbf{y}_i = 0, \mathbf{g}_{\mathcal{S},i} = 1) &= \frac{z_{\mathcal{S}} - a_{\mathcal{S}}}{n} \\ \mathbb{P}(\mathbf{y}_i = 0, \mathbf{g}_{\mathcal{S},i} = 0) &= \frac{n - n_1 - (z_{\mathcal{S}} - a_{\mathcal{S}})}{n} \end{aligned} \tag{3.5}$$

Those probabilities sum up to one, and the marginal probabilities, i.e. $\mathbb{P}(\mathbf{y} = 1)$ and $\mathbb{P}(\mathbf{g}_{\mathcal{S}} = 1)$ can be estimated as n_1/n and $z_{\mathcal{S}}/n$, respectively.

Given the counts n , n_1 and $z_{\mathcal{S}}$, the contingency table only has one degree of freedom, i.e. fixing one count in the table determines all other counts. Without loss of generality, we chose the count $a_{\mathcal{S}}$. The classical Fisher's exact test and Pearson's χ^2 test both treat the table margins $z_{\mathcal{S}}$, n and n_1 as fixed quantities, focussing on the cell count $a_{\mathcal{S}}$ as the only random quantity, and thus the central object in the test statistic. Both tests derive a test statistic that depends on the random variable $A_{\mathcal{S}}$, assuming that the cell count $a_{\mathcal{S}}$ is a realisation of $A_{\mathcal{S}}$, and establish its distribution under the null hypothesis of independence between the random variables Y and $G_{\mathcal{S}}$. We denote this test statistic as $T(A_{\mathcal{S}})$. It can be shown [see 166, for proof] that under the null hypothesis \mathbb{H}_0 of independence between Y and $G_{\mathcal{S}}$, the conditional probability of the cell count $a_{\mathcal{S}}$ follows

3. Introduction to significant pattern mining

a hypergeometric distribution, that is

$$\begin{aligned} \mathbb{P}(A_{\mathcal{S}} = a_{\mathcal{S}} | X_{\mathcal{S}} = z_{\mathcal{S}}, N_1 = n_1, \mathbb{H}_0) &= \text{hypergeom}(a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}) \\ &= \frac{\binom{n_1}{a_{\mathcal{S}}} \binom{n-n_1}{z_{\mathcal{S}}-a_{\mathcal{S}}}}{\binom{n}{z_{\mathcal{S}}}}. \end{aligned} \quad (3.6)$$

In the following, we will explain the test statistics and resulting null-distributions for Pearson's χ^2 test and Fisher's exact test, following the description in the review by Llinares-López & Borgwardt [166].

3.3.1. Pearson's χ^2 test

The test statistic in Pearson's χ^2 test [107] is based on a z-score like transformation of the cell count $a_{\mathcal{S}}$, i.e.

$$Z_{\text{Pearson}} = \frac{a_{\mathcal{S}} - \mathbb{E}[a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}]}{\sqrt{\sigma[a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}]}} \quad (3.7)$$

where $\mathbb{E}[a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}]$ corresponds to the expected value of $a_{\mathcal{S}}$, and $\sigma[a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}]$ to the standard deviation of $a_{\mathcal{S}}$ under the null distribution. As stated above, this null distribution corresponds to a hypergeometric distribution, such that [see, e.g. 179]

$$Z_{\text{Pearson}} = \frac{a_{\mathcal{S}} - z_{\mathcal{S}} \frac{n_1}{n}}{\sqrt{\frac{z_{\mathcal{S}}(n-z_{\mathcal{S}})(n-n_1)}{n^2(n-1)}} n_1}. \quad (3.8)$$

The central limit theorem implies that for large sample sizes n , and independently and identically distributed samples, the statistic Z_{Pearson} converges to a standard normal distribution under the null hypothesis. The test statistic used in Pearson's χ^2 test corresponds to the square of the test statistic Z_{Pearson} and as such follows a χ^2 distribution with one degree of freedom, i.e.

$$T_{\text{Pearson}} = Z_{\text{Pearson}}^2 \sim \chi_1^2. \quad (3.9)$$

To obtain a p -value from the Pearson test, the probability of the observed value of the test statistic $T_{\text{Pearson}}^{\text{obs}}$ under the null-distribution is computed, that is:

$$p_{\mathcal{S}, \text{Pearson}}(a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}}) = F_{\chi_1^2}(T_{\text{Pearson}}^{\text{obs}}(a_{\mathcal{S}} | n, n_1, z_{\mathcal{S}})), \quad (3.10)$$

where $F_{\chi_1^2}(\cdot)$ indicates the survival function of the χ^2 distribution with one degree of freedom. Intuitively, the test statistic T_{Pearson} will take larger values when the observed table count $a_{\mathcal{S}}$ deviates from the expected table count, given the margins. This hints towards a dependence between the random variables, and hence the existence of an association. For those cases, the p -value will take low values, and indicate that the null-hypothesis of independence can be rejected.

It is important to note that Pearson's χ^2 test relies on the central limit theorem, and on that account on an *approximation* of the distribution of the test statistic. This approxima-

3. Introduction to significant pattern mining

tion is valid for large sample sizes, and identically and independently distributed samples. In cases where only few samples are present, or the independence between samples is violated, the approximation might be inaccurate and lead to false conclusions with respect to the rejection of the null hypothesis.

3.3.2. Fisher's exact test

Fisher's exact test [180] is, similarly to Pearson's χ^2 test, applied to assess the independence of categorical random variables stored in contingency tables. As opposed to Pearson's test, it does not rely on the approximation of the null distribution, but its exact computation, making it the method of choice in the presence of only few samples.

The test statistic used for Fisher's exact test corresponds to the probability under the hypergeometric distribution itself (see Equation 3.6), i.e.

$$T_{\text{Fisher}}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) = \text{hypergeom}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) = \frac{\binom{n_1}{a_{\mathcal{S}}} \binom{n-n_1}{z_{\mathcal{S}}-a_{\mathcal{S}}}}{\binom{n}{z_{\mathcal{S}}}}. \quad (3.11)$$

Hence, the probability of a configuration of the contingency table is given as

$$\mathbb{P}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) = T_{\text{Fisher}}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) \quad (3.12)$$

To obtain a two-sided p -value of association, Fisher's exact test adds the probabilities of all configurations that are less probable than the observed one, given the margins $z_{\mathcal{S}}$, n_1 and n . More formally [following the notation in 166], for an observed table count $a_{\mathcal{S}}$, we denote the set $\mathcal{A}(a_{\mathcal{S}}) = \{a'_{\mathcal{S}} | \text{hypergeom}(a'_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) \leq \text{hypergeom}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}})\}$, and write the p -value as

$$p_{\text{Fisher}}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) = \sum_{a_{\mathcal{S}} \in \mathcal{A}(a_{\mathcal{S}})} \text{hypergeom}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}). \quad (3.13)$$

3.3.3. Correction for multiple hypothesis testing in significant pattern mining

As we have seen in the introduction to this chapter, one major challenge in pattern mining is the statistical burden induced by the exploration of large numbers of patterns [181]. Some of the concepts in this chapter have been introduced previously in Section 2.2.2.1 that covered the problem of multiple hypothesis testing in the context of genome-wide association studies. In order to make this introduction self-contained, we repeat those concepts here.

To assess the association of a pattern \mathcal{S} with the label, a contingency table as in Figure 3.1b is created, and a p -value $p_{\mathcal{S}}$ of association is computed, as described in the preceding section. Based on the evidence in the data in the form of this p -value, one decides whether the null hypothesis \mathbb{H}_0 of independence between \mathbf{y} and $\mathbf{g}_{\mathcal{S}}$ should be rejected. To achieve this, a significance level α is set, commonly to a value of 0.05 or 0.01, and the null hypothesis is rejected for a *single* pattern \mathcal{S} if its p -value falls below this significance level,

3. Introduction to significant pattern mining

i.e.

$$\text{if } p_{S,\cdot}(a_S|n, n_1, z_S) \leq \alpha \text{ then reject } \mathbb{H}_0, \quad (3.14)$$

where $p_{S,\cdot}$ indicates the p -value obtained either with Fisher's exact test or Pearson's χ^2 test. If this is the case, the pattern is said to be statistically significantly associated to the label.

However, when testing large numbers of patterns simultaneously, as is the case in significant pattern mining, this potentially induces a large number of false positives. p -values are by definition uniformly distributed under the null hypothesis \mathbb{H}_0 , implying that each test for which \mathbb{H}_0 is true has a $\alpha \times 100.0\%$ chance to be incorrectly reported as significant. This incorrect rejection of the null hypothesis is also referred to as committing a **type-I error**. Hence, when testing a single pattern, there is a $\alpha \times 100.0\%$ probability to commit a type-I error. If multiple tests are performed simultaneously, and each one of them is tested at significance level α , the probability to observe one or more false positive associations increases far above α , giving rise to a severe **multiple hypothesis testing problem**, and potentially leading to a large amount of false positive associations. The concept of the number of false positive associations is formalised in the **family-wise error rate (FWER)**, that is the probability of committing at least one type-I error:

$$\text{FWER} = \mathbb{P}(\text{number of false positives} \geq 1). \quad (3.15)$$

A widely-used approach to reduce the number of type-I errors evolves around control of the FWER, that is the FWER should meet the pre-defined significance level α . This is achieved by finding a significance threshold $\delta \leq \alpha$ at which each individual hypothesis is tested, such that the chance of observing a false positive association stays below $\alpha \times 100.0\%$. Importantly, the number of false positive associations depends on this significance threshold δ : When testing all hypothesis is the hypothesis space \mathcal{H} , the total number of tests corresponds to $|\mathcal{H}|$. For those $|\mathcal{H}|$ tests, and a significance threshold δ , the FWER can be estimated using the following equation:

$$\begin{aligned} \text{FWER}(\delta) &= \mathbb{P}(\text{number of false positives} \geq 1 \mid \delta) \\ &= 1 - \mathbb{P}(\text{number of false positives} = 0 \mid \delta) \\ &= 1 - \prod_{i=1}^{|\mathcal{H}|} \mathbb{P}(p_i \geq \delta) \\ &= 1 - \prod_{i=1}^{|\mathcal{H}|} (1 - \mathbb{P}(p_i \leq \delta)) \\ &= 1 - (1 - \delta)^{|\mathcal{H}|} \end{aligned} \quad (3.16)$$

The last step follows due to the p -values being uniformly distributed. The FWER for different values of δ is illustrated in Figure 2.2 (where δ is replaced by α) for varying numbers of tests. It can be seen that for large numbers of tests the probability to observe one or more false positive associations increases, and correction for MHT is indispensable.

3. Introduction to significant pattern mining

Formally, the goal of FWER control is to find a $\delta \in (0, 1)$, such that

$$\text{FWER}(\delta) \leq \alpha \quad (3.17)$$

From Equations 3.16 and 3.17, it follows that the FWER is controlled for $|\mathcal{H}| = 1$ by setting $\delta = \alpha$. However, as soon as $|\mathcal{H}|$ exceeds 1, the threshold δ has to be adapted.

A widely-used approach to guarantee control of the FWER is the Bonferroni correction [182], that defines the significance threshold as $\delta = \alpha/|\mathcal{H}|$. The Bonferroni correction guarantees control of the FWER, which can be shown using Bool's inequality: assuming that \mathcal{H}_0 is the set of all hypothesis for which the null hypothesis is true, and p_i indicates the p -value of the i -th hypothesis, the FWER at threshold $\delta = \alpha/|\mathcal{H}|$ can be estimated as

$$\begin{aligned} \text{FWER}(\delta) &= \mathbb{P} \left\{ \bigcup_{i=1}^{|\mathcal{H}_0|} (p_i \leq \delta) \right\} \leq \sum_{i=1}^{|\mathcal{H}_0|} \mathbb{P}(p_i \leq \delta) \\ &\stackrel{(*)}{=} |\mathcal{H}_0| \delta = |\mathcal{H}_0| \frac{\alpha}{|\mathcal{H}|} \leq |\mathcal{H}| \frac{\alpha}{|\mathcal{H}|} = \alpha, \end{aligned} \quad (3.18)$$

where $(*)$ follows from the fact that p -values are uniformly distributed under the null hypothesis. Alternatively, when inserting the threshold $\delta = \alpha/|\mathcal{H}|$ into Equation 3.16, one can show that

$$\lim_{|\mathcal{H}| \rightarrow \infty} 1 - \left(1 - \frac{\alpha}{|\mathcal{H}|}\right)^{|\mathcal{H}|} = 1 - e^{-\alpha} < \alpha \quad \forall \alpha \in \mathbb{R}_+. \quad (3.19)$$

In an ideal scenario, one is interested in finding the largest possible threshold δ^* that still guarantees control of the FWER, i.e. one is interested in finding

$$\delta^* = \max \{ \delta \mid \text{FWER}(\delta) \leq \alpha \} \quad (3.20)$$

such that δ^* achieves highest possible statistical power, i.e. minimises the **type-II error**. While Bonferroni's method is widely used to control the FWER, it is known to be very conservative, that is $\delta_{\text{Bonf}} \ll \delta^*$. The Bonferroni correction controls the type-I error at the expense of the type-II error: one accepts missing true associations to avoid false associations. In significant pattern mining, the number of simultaneous hypothesis tests is often so large such that none of the patterns reach the very stringent significance threshold imposed by the Bonferroni correction, especially when sample sizes are low. This stresses the need for different approaches to address the problem of control of the FWER.

3.3.4. Tarone's procedure: an improved Bonferroni correction for discrete test statistics

An important contribution to the field of type-I error control using the FWER was made by Tarone [183], hereafter referred to as **Tarone's procedure**. Tarone observed that for discrete tests a significance threshold δ_{Tar} can be found that satisfies

$$\delta_{\text{Bonf}} \ll \delta_{\text{Tar}} \leq \delta^* \quad (3.21)$$

3. Introduction to significant pattern mining

that still guarantees control of the type-I error, but decreases the type-II error compared to the standard Bonferroni correction. Tarone's threshold is computed as $\delta_{\text{Tar}} = \alpha/|\mathcal{H}_{\text{test}}|$, where $\mathcal{H}_{\text{test}}$ is the subset of testable hypotheses $\mathcal{H}_{\text{test}} \subset \mathcal{H}$, and generally $|\mathcal{H}_{\text{test}}| \ll |\mathcal{H}|$. The following sections explain how this set of testable hypothesis is defined. It is based on the concept of the minimum p -value of a discrete test.

3.3.4.1. The concept of the minimum p -value

Assume a 2×2 contingency table is used to assess the independence between a pattern and a label, based on the binary label vector \mathbf{y} and the indicator function $\mathbf{g}_{\mathcal{S}}$ of the pattern \mathcal{S} for n independent and identically distributed (i.i.d.) samples, as illustrated in Figure 3.1b. This contingency table is uniquely defined by the table margins n_1 , n and $z_{\mathcal{S}}$, as well as the table count $a_{\mathcal{S}}$.

Given a pattern \mathcal{S} , we can compute the table margins. Given only those margins, the table has exactly one degree of freedom, i.e. by setting one of the table counts, all other table counts are defined. We chose without loss of generality the table count $a_{\mathcal{S}}$, i.e. the number of samples in the positive class that contain the pattern \mathcal{S} . Due to the discreteness of the test and the fixed margins, there exists only a finite number of possible $a_{\mathcal{S}}$ values for the contingency table. We denote those values as the set $\mathcal{A}_{\mathcal{S}} = \{a_{\mathcal{S}} \mid a_{\mathcal{S}}^{\min} \leq a_{\mathcal{S}} \leq a_{\mathcal{S}}^{\max}, a_{\mathcal{S}} \in \mathbb{N}\}$, where

$$\begin{aligned} a_{\mathcal{S}}^{\max} &= \min(z_{\mathcal{S}}, n_1) \\ a_{\mathcal{S}}^{\min} &= \max(0, n_1 + z_{\mathcal{S}} - n), \end{aligned} \quad (3.22)$$

and the number of possible configurations of the table corresponds to the size of this set, i.e. $|\mathcal{A}_{\mathcal{S}}| = a_{\mathcal{S}}^{\max} - a_{\mathcal{S}}^{\min} + 1$.

For each of the table configurations induced by the set $\mathcal{A}_{\mathcal{S}}$ a p -value can be computed, and the minimum p -value of a pattern corresponds to the smallest p -value among those. Formally, we define the minimum p -value as

$$\Phi(n_1, n, z_{\mathcal{S}}) = \min_p \{p_{\mathcal{S},\cdot}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) \mid a_{\mathcal{S}} \in \mathcal{A}_{\mathcal{S}}\}, \quad (3.23)$$

where $p_{\mathcal{S},\cdot}$ corresponds to p -value, e.g. obtained from a χ^2 test or Fisher's exact test for pattern \mathcal{S} . Since we assume the label vector for a data set to be fixed, the table margins n and n_1 are constants, and we write the minimum p -value as a function of the support $z_{\mathcal{S}}$ only, i.e. $\Phi(z_{\mathcal{S}})$. The minimum p -value $\Phi(z_{\mathcal{S}})$ corresponds to a lower bound of the observed p -value, i.e.

$$p_{\mathcal{S},\cdot}(a_{\mathcal{S}}|n, n_1, z_{\mathcal{S}}) \geq \Phi(z_{\mathcal{S}}). \quad (3.24)$$

As such, the minimum p -value indicates the strongest possible association for a pattern \mathcal{S} . For Pearson's χ^2 test and Fisher's exact test, the minimum p -values can be computed in closed form, as described in the following.

Remark 3.3.1 (Minimum p -value computation does not require labels). *Importantly, for pattern \mathcal{S} , the table margin n_1 is a function of the label vector \mathbf{y} , i.e. $n_1 = f(\mathbf{y})$, and $\mathbf{g}_{\mathcal{S}}$*

3. Introduction to significant pattern mining

is a function of the pattern indicator \mathbf{g}_S , i.e. $z_S = f(\mathbf{g}_S)$, where $f(\cdot)$ returns the sum over the input vector. Hence, computing the table margins does not require any combination of the pattern indicator and the labels, and to derive the minimum p -value of a pattern, this information is not used.

The minimum p -value for Pearson's χ^2 test

For Pearson's χ^2 test, a minimum p -value can be computed in closed form [see Proposition 8.3 in 166]. We define $n_a = \min(n_1, n - n_1)$ and $n_b = \max(n_1, n - n_1)$. For Pearson's χ^2 test, the minimum p -value is given as

$$\Phi_{\text{Pearson}}(z_S) = \begin{cases} F_{\chi^2_1} \left((n-1) \frac{n_b}{n_a} \frac{z_S}{n-z_S} \right) & \text{if } 0 \leq z_S \leq n_a \\ F_{\chi^2_1} \left((n-1) \frac{n_a}{n_b} \frac{n-z_S}{z_S} \right) & \text{if } n_a \leq z_S \leq \frac{n}{2} \\ F_{\chi^2_1} \left((n-1) \frac{n_a}{n_b} \frac{z_S}{n-z_S} \right) & \text{if } \frac{n}{2} \leq z_S \leq n_b \\ F_{\chi^2_1} \left((n-1) \frac{n_b}{n_a} \frac{n-z_S}{z_S} \right) & \text{if } n_b \leq z_S \leq n \end{cases} \quad (3.25)$$

where $F_{\chi^2_1}$ corresponds to the survival function of the χ^2 distribution with one degree of freedom. A proof of the minimum p -value for Pearson's χ^2 test can be found in Llinares-López & Borgwardt [166].

The minimum p -value for Fisher's exact test

For Fisher's exact test, a minimum p -value can be computed in closed form [see Proposition 8.4. in 166]. We define $n_a = \min(n_1, n - n_1)$ and $n_b = \max(n_1, n - n_1)$. For Fisher's exact test, the minimum p -value is given as

$$\Phi_{\text{Fisher}}(z_S) = \begin{cases} \binom{n_a}{z_S} / \binom{n}{z_S} & \text{if } 0 \leq z_S \leq n_a \\ \binom{n_b}{n-z_S} / \binom{n}{z_S} & \text{if } n_a \leq z_S \leq \frac{n}{2} \\ \binom{n_b}{z_S} / \binom{n}{z_S} & \text{if } \frac{n}{2} \leq z_S \leq n_b \\ \binom{n_a}{n-z_S} / \binom{n}{z_S} & \text{if } n_b \leq z_S \leq n \end{cases} \quad (3.26)$$

A proof of the minimum p -value function for Fisher's exact test can be found in Llinares-López & Borgwardt [166].

Figure 3.2 illustrates the minimum p -values for Pearson's χ^2 test and Fisher's exact test for $n = 100$ samples. In the case of balanced data, i.e. $n_1 = \frac{n}{2}$, the resulting minimum p -value functions $\Phi(\cdot)$ exhibit a V-like shape with one global minimum at n_1 (see Figure 3.2a). In the case of an unbalanced data set, the minimum p -value function exhibits a W-like shape with two minima, reached at n_1 and $n - n_1$ (see Figure 3.2b). For both scenarios, the minimum p -value functions are symmetric around $\frac{n}{2}$.

3.3.4.2. Testability and the improved Bonferroni correction

Tarone used the minimum p -value to define an improved Bonferroni correction [183] by introducing the concept of **testability**. Let us for now assume a fixed significance threshold

3. Introduction to significant pattern mining

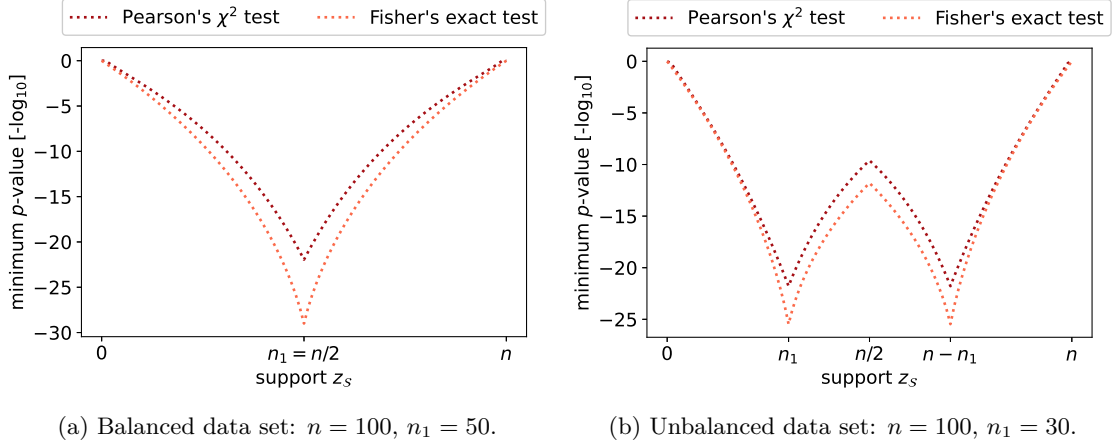


FIGURE 3.2.: Minimum p -values for Pearson's χ^2 test (dark red) and Fisher's exact test (light red) for (a) a balanced data set where $n_1 = \frac{n}{2}$, and (b) an unbalanced data set where $n_1 < \frac{n}{2}$. (This figure is adapted from Figure 8.4 in Llinares-López & Borgwardt [166].)

δ , and denote with $\mathcal{H}_{\text{test}}(\delta)$ the set of all patterns that have a minimum p -value that fall below δ , i.e.

$$\mathcal{H}_{\text{test}}(\delta) = \{\mathcal{S} \mid \Phi.(z_{\mathcal{S}}) \leq \delta\}, \quad (3.27)$$

where Φ indicates a minimum p -value obtained from any discrete test. This is called the set of *testable hypotheses at threshold δ* , and a pattern $\mathcal{S} \in \mathcal{H}_{\text{test}}(\delta)$ is called *testable at threshold δ* . This is illustrated in Figure 3.3 by means of the minimum p -value function for Fisher's exact test (Pearson's χ^2 test follows analogously).

This number of testable hypotheses $|\mathcal{H}_{\text{test}}(\delta)|$ can be used to derive an improved Bonferroni threshold, by replacing the correction factor $|\mathcal{H}|$ with $|\mathcal{H}_{\text{test}}(\delta)|$, i.e. Tarone's threshold is defined as $\delta_{\text{Tar}} = \alpha/|\mathcal{H}_{\text{test}}(\delta)|$. Since $|\mathcal{H}_{\text{test}}(\delta)| \leq |\mathcal{H}|$ is trivially fulfilled, the Bonferroni threshold δ_{Bonf} constitutes a lower bound of the resulting Tarone threshold δ_{Tar} . Furthermore, using Bool's inequality, one can show that $\delta |\mathcal{H}_{\text{test}}(\delta)|$ constitutes an upper bound to the family-wise error rate at threshold δ :

$$\begin{aligned} \text{FWER}(\delta) &= \mathbb{P} \left\{ \bigcup_{\mathcal{S} \in \mathcal{H}_0} (p_{\mathcal{S}} \leq \delta) \right\} \stackrel{(1)}{=} \mathbb{P} \left\{ \bigcup_{\mathcal{S} \in \mathcal{H}_{\text{test}}(\delta)} (p_{\mathcal{S}} \leq \delta) \right\} \\ &\leq \sum_{\mathcal{S} \in \mathcal{H}_{\text{test}}(\delta)} \mathbb{P}(p_{\mathcal{S}} \leq \delta) \leq |\mathcal{H}_{\text{test}}(\delta)| \delta \end{aligned} \quad (3.28)$$

where (1) follows from the fact that a pattern that is not testable at threshold δ , i.e. $\mathcal{S} \in \mathcal{H}_0/\mathcal{H}_{\text{test}}(\delta)$ does not have a p -value below δ by definition, and is subsequently not contained in the union [166]. The intuition behind Tarone's threshold is simple: a pattern that is not testable at an adjusted significance threshold δ can by definition of testability never become significant, hence it cannot become a false positive association and as such does not contribute to the FWER.

3. Introduction to significant pattern mining

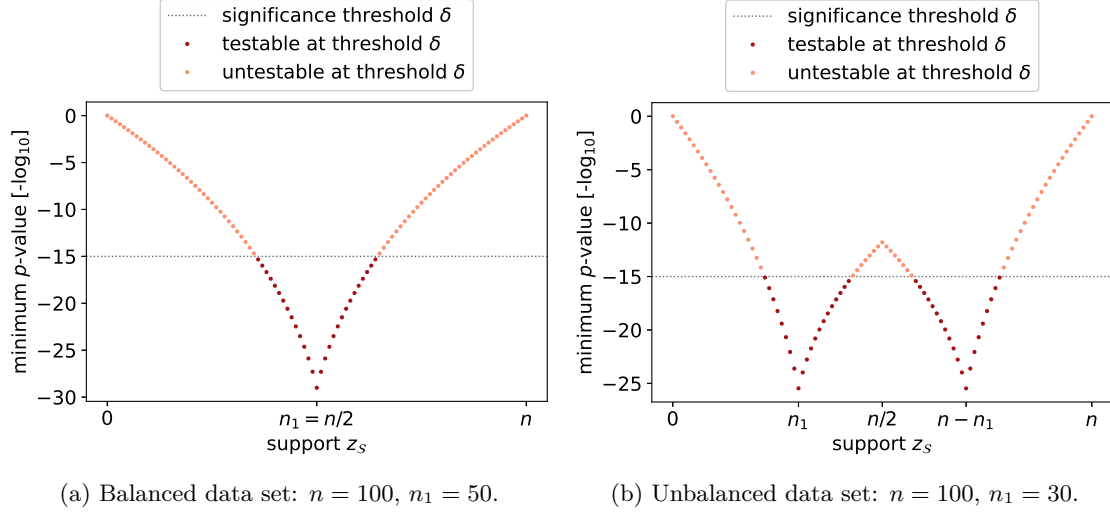


FIGURE 3.3.: Illustration of testability at threshold δ for (a) balanced data set, and (b) unbalanced data set with minimum p -values computed from Fisher’s exact test. The grey horizontal line indicates the significance threshold. All hypotheses with minimum p -value below δ are testable at threshold δ (dark red), and other hypotheses are untestable (light red). (This figure is adapted from Figure 8.5 in Llinares-López & Borgwardt [166])

Two major challenges have to be considered when applying Tarone’s procedure:

1. the choice of the significance threshold δ , and
2. the enumeration of the set $|\mathcal{H}_{\text{test}}(\delta)|$.

The first challenge can be addressed by using the approximation to the FWER in Equation 3.28. That is, assuming we can evaluate $|\mathcal{H}_{\text{test}}(\delta)|$, δ is chosen such that

$$\delta^* = \max_{\delta} \{ \delta \mid \text{FWER}(\delta) \leq \alpha \} = \max_{\delta} \{ \delta \mid |\mathcal{H}_{\text{test}}(\delta)| \delta \leq \alpha \}. \quad (3.29)$$

This threshold guarantees control of the FWER while enabling maximal statistical power. The second challenge, i.e. the enumeration of the testable sets, naïvely requires the computation of the minimum p -values of all patterns in the hypothesis space \mathcal{H} . Due to the excessive size of the \mathcal{H} for most problems in significant pattern mining, this naïve enumeration is infeasible. Fortunately, the minimum p -values exhibit favourable monotonicity properties that can be used to efficiently prune the hypothesis space \mathcal{H} , and lighten the burden of an exhaustive enumeration [158, 159, 161, 164].

3.4. Enabling significant pattern mining with Tarone’s testability criterion

In the previous section we identified the multiple hypothesis testing problem as one of the major impairments in significant pattern mining. The Tarone procedure provides an improvement over the classical Bonferroni correction, however if applied in a naïve way, it

3. Introduction to significant pattern mining

requires the enumeration of all patterns \mathcal{S} in the hypothesis space \mathcal{H} and the evaluation of their minimum p -values. For most significant pattern mining problems, the search space \mathcal{H} is excessive in size, thus an enumeration is infeasible.

This limitation inhibited the usage of Tarone’s procedure until recently: Terada et al. [158] showed that discrete tests, such as Pearson’s χ^2 test and Fisher’s exact test, exhibit certain properties that enable an efficient pruning of the search space \mathcal{H} , and thereby reduce the computational burden of enumerating all hypothesis substantially. This contribution laid the cornerstone for further development and improvement of algorithms that incorporate Tarone’s correction into significant pattern mining [159–161, 163]. In this section, we present those contributions for the task of significant *pattern* mining, without focussing on a specific type of pattern, such that a pattern can be defined as any discrete substructure of the features in the dataset.

Enabling Tarone’s procedure in significant pattern mining mainly relies on two important monotonicity properties that can be exploited to efficiently prune the search space. We refer to the first one as the Apriori property as it was exploited in the Apriori algorithm for association rule mining [169]. The second type of monotonicity is the piecewise monotonicity of the minimum p -value, which will be explained later.

3.4.1. The Apriori property

The **Apriori algorithm** is an algorithm for association rule mining, i.e. one is interested in finding relationships of the form $\mathcal{S} \Rightarrow \mathcal{S} \cup \{j\}$, where $j \in \mathcal{J}$ is a feature index that is not yet contained in \mathcal{S} , i.e. $\mathcal{S} \cap \{j\} = \emptyset$. The association rule holds with confidence c , if $c\%$ of the samples in the data set that contain \mathcal{S} also contain $\{j\}$, and the association rule has support s , if $s\%$ of samples in the dataset contain $\mathcal{S} \cup \{j\}$. The goal is to find association rules, that exceed both, a user defined minimum support, as well as a user-defined minimum confidence [169].

Agrawal, Srikant, et al. [169] developed an algorithm for the efficient search of such association rules, called the **Apriori algorithm**. It is based on a two-step procedure, where in the first step all itemsets that exceed the user defined support are enumerated, giving rise to a set of frequent itemsets, and in the second step, subsets of those frequent itemsets are searched for association rules that satisfy the confidence threshold. The first step of this Apriori algorithm is based on the observation that subsets of frequent itemsets have to be frequent themselves. This observation has profound implications for finding the Tarone threshold as well, and we will define it formally. Importantly, depending on whether the minimum or the maximum pattern indicator function is used, the Apriori properties proposes a monotonic, or an anti-monotonic behaviour.

Proposition 3.4.1 (Apriori property using the maximum indicator function). *Given two patterns $\mathcal{S}, \mathcal{S}' \subset \mathcal{J}$, such that $\mathcal{S} \subset \mathcal{S}'$. Then the corresponding supports of the patterns $z_{\mathcal{S}}$ and $z_{\mathcal{S}'}$ with respect to the maximum pattern indicator function g^{\max} satisfy the following monotonicity property: $z_{\mathcal{S}} \leq z_{\mathcal{S}'}$.*

Proof. For the purpose of the proof, let us decompose the pattern \mathcal{S}' into \mathcal{S} and its complement $\bar{\mathcal{S}} = \mathcal{S}' \setminus \mathcal{S}$. For an individual sample i , we can decompose the maximum

3. Introduction to significant pattern mining

Algorithm 3.1 Significant pattern mining with Tarone’s correction

Input: Data set $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, target family-wise error rate α
Output: $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}})$
1: $n \leftarrow \mathbf{y}.size$
2: $n_1 \leftarrow \mathbf{y}.sum$
3: $\delta_{\text{Tar}}, \mathcal{H}_{\text{test}}(\delta_{\text{Tar}}) \leftarrow \text{FIND_TARONE_THRESHOLD}(\mathbf{X}, n, n_1, \alpha)$
4: $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}}) \leftarrow \{\mathcal{S} \mid \mathcal{S} \in \mathcal{H}_{\text{test}}(\delta_{\text{Tar}}), p_{\mathcal{S}} \leq \delta_{\text{Tar}}\}$

combination of patterns as $g_{\mathcal{S}'}^{\max}(i) = \max(g_{\mathcal{S}}^{\max}(x_i), g_{\bar{\mathcal{S}}}^{\max}(x_i))$. Hence, $g_{\mathcal{S}'}^{\max}(x_i) \geq g_{\mathcal{S}}^{\max}(x_i)$ for each sample i . Since $z_{\mathcal{S}} = \sum_{i=1}^n g_{\mathcal{S}}^{\max}(x_i)$ and $z_{\mathcal{S}'} = \sum_{i=1}^n g_{\mathcal{S}'}^{\max}(x_i)$, it follows that $z_{\mathcal{S}'} \geq z_{\mathcal{S}}$. \square

Proposition 3.4.2 (Apriori property using the minimum indicator). *Given two patterns $\mathcal{S}, \mathcal{S}' \subset \mathcal{J}$, such that $\mathcal{S} \subset \mathcal{S}'$. Then the corresponding supports of the patterns $z_{\mathcal{S}}$ and $z_{\mathcal{S}'}$ with respect to the minimum pattern indicator function g^{\min} fulfil the following anti-monotonicity property: $z_{\mathcal{S}} \geq z_{\mathcal{S}'}$.*

Proof. For the purpose of the proof, let us decompose the pattern \mathcal{S} into \mathcal{S}' and its complement $\bar{\mathcal{S}} = \mathcal{S}' \setminus \mathcal{S}$. For an individual sample i , we can decompose the minimum combination of patterns as $g_{\mathcal{S}'}^{\min}(x_i) = \min(g_{\bar{\mathcal{S}}}^{\min}(x_i), g_{\mathcal{S}'}^{\min}(x_i))$. Hence, $g_{\mathcal{S}'}^{\min}(x_i) \leq g_{\mathcal{S}}^{\min}(x_i)$ for each sample i . Since $z_{\mathcal{S}} = \sum_{i=1}^n g_{\mathcal{S}}^{\min}(x_i)$ and $z_{\mathcal{S}'} = \sum_{i=1}^n g_{\mathcal{S}'}^{\min}(x_i)$, it follows that $z_{\mathcal{S}'} \leq z_{\mathcal{S}}$. \square

3.4.2. A generic significant pattern mining algorithm

The last sections introduced the concepts underlying significant pattern mining and Tarone’s procedure to control the FWER. Herein, we explain how those concepts can be combined to give rise to efficient algorithms to solve the statistical and computational challenges of significant pattern mining. We do so by means of a generic pattern mining algorithm (this section follows Section 8.3 by Llinares-López & Borgwardt [166]).

The main routine in significant pattern mining is given in Algorithm 3.1. We will go through each line of the code separately to explain the individual steps.

At initialisation, the method requires two different types of input. The first one is the data set \mathcal{D} , consisting of the binary data matrix $\mathbf{X} \in \{0, 1\}^{n \times d}$ that describes the presence ($\mathbf{X}_{i,j} = 1$) or absence ($\mathbf{X}_{i,j} = 0$) of feature j in sample i , as well as the binary label $\mathbf{y} \in \{0, 1\}^n$ that assigns each sample to one of two distinct classes. Without loss of generality, we assume the minority class to have label 1, i.e. $n_1 \leq n - n_1$ in a contingency table as the one indicated in Figure 3.1b. Furthermore, the target FWER α has to be supplied. Line 1 and 2 set the number of samples and the number of samples in the minority class with label 1.

The main contribution lies in Line 3, i.e. the call to the function that determines the Tarone-corrected significance threshold δ_{Tar} and the testable patterns at this threshold, $\mathcal{H}_{\text{test}}(\delta_{\text{Tar}})$. Importantly, this function does not depend on the label vector, but only on the marginal counts n and n_1 . It is explained in greater detail in Algorithm 3.2.

3. Introduction to significant pattern mining

Algorithm 3.2 Significant pattern mining with Tarone’s correction: specifics

```

1: function FIND_TARONE_THRESHOLD( $\mathbf{X}, n, n_1, \alpha$ )
2:   Initialise global variables  $\hat{\delta} \leftarrow \alpha, \mathcal{H}_{\text{test}} \leftarrow \emptyset$ 
3:   PROCESS_PATTERN( $\emptyset, n, n_1, \alpha$ )
4:    $\delta_{\text{Tar}} \leftarrow \alpha / |\mathcal{H}_{\text{test}}|$ 
5:   return  $\delta_{\text{Tar}}, \mathcal{H}_{\text{test}}$ 
6: end function
7:
8: function PROCESS_PATTERN( $\mathcal{S}, n, n_1, \alpha$ )
9:   Compute the minimum  $p$ -value  $\Phi(z_{\mathcal{S}})$  given  $n, n_1$ 
10:  if  $\Phi(z_{\mathcal{S}}) \leq \hat{\delta}$  then ▷ Pattern is testable.
11:     $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \cup \mathcal{S}$  ▷ Add to list of testables.
12:    while  $\hat{\delta} \times |\mathcal{H}_{\text{test}}| \geq \alpha$  do
13:      Decrease  $\hat{\delta}$  ▷ Adapt significance threshold.
14:       $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \setminus \{\mathcal{S} \mid \Phi(z_{\mathcal{S}}) \geq \hat{\delta}\}$  ▷ Remove untestable patterns from  $\mathcal{H}_{\text{test}}$ .
15:    end while
16:  end if
17:  if not IS_PRUNABLE( $\mathcal{S}$ ) then
18:    for all super-patterns  $\mathcal{S}'$  of  $\mathcal{S}$  do
19:      PROCESS_PATTERN( $\mathcal{S}', n, n_1, \alpha$ ) ▷ Explore all super-patterns of  $\mathcal{S}$ .
20:    end for
21:  end if
22: end function

```

After the Tarone-corrected significance threshold δ_{Tar} has been computed, all testable patterns are tested for association with the label vector \mathbf{y} . Two things should be noted here: (i) this is the first time the label vector is used in conjunction with the patterns, namely to compute the actual contingency table and the corresponding p -value $p_{\mathcal{S}}$, and (ii) the set of significant hypothesis is a subset of the testable hypothesis at threshold δ_{Tar} , i.e. $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}}) \subset \mathcal{H}_{\text{test}}(\delta_{\text{Tar}})$. The subset that have p -values $p_{\mathcal{S}}$ below the corrected threshold δ_{Tar} are considered statistically significant, and the set is denoted by \mathcal{H}_{sig} .

The function FIND_TARONE_THRESHOLD is the core routine of the significant pattern mining algorithm with Tarone’s correction. It is described in Algorithm 3.2. Upon initialisation, two global variables are set: $\hat{\delta}$, which is the significance threshold that determines the current set of testable patterns, and that will be decreased successively as more and more patterns are explored. The second variable is the set $\mathcal{H}_{\text{test}}$ that will hold testable patterns at the current significance threshold $\hat{\delta}$. At the beginning, this set is empty, as none of the patterns have been explored. Line 3 calls the main routine PROCESS_PATTERN, a recursive function that explores patterns organised in a pattern enumeration tree using a depth-first search strategy. An example of a pattern enumeration tree for the task of itemset mining can be found in Figure 3.4. Those trees are organised such that every node represents a pattern \mathcal{S} , and moving deeper into the tree corresponds to extending, or growing, the pattern by adding additional features. With this, the relation $\mathcal{S}' \in \text{child}(\mathcal{S}) \Leftrightarrow \mathcal{S} \subset \mathcal{S}'$ holds. We will see later that this organisation enables efficient pruning of the search space \mathcal{H} .

3. Introduction to significant pattern mining

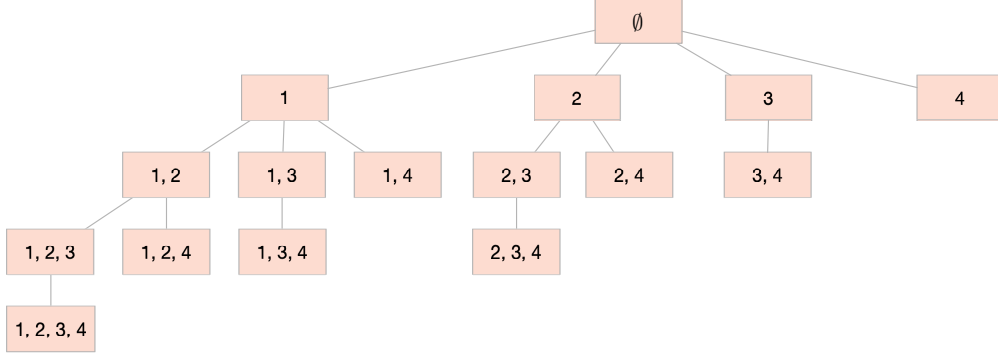


FIGURE 3.4.: Example of a pattern enumeration tree for $d = 4$ features. The tree contains all possible 2^d itemsets, such that each node corresponds to one itemset \mathcal{S} , and if a itemset \mathcal{S}' is in the set of children of \mathcal{S} , the relationship $\mathcal{S} \subset \mathcal{S}'$ holds. (This figure is adapted from Figure 8.6 in Llinares-López & Borgwardt [166]).

The function `PROCESS_PATTERN` takes a pattern \mathcal{S} as input, and evaluates its support $z_{\mathcal{S}}$ and minimum p -value $\Phi(z_{\mathcal{S}})$ given n, n_1 . The latter is in the next step (Lines 10 to 15) used to assess testability of the pattern. If a pattern is testable, i.e. if $\Phi(z_{\mathcal{S}}) \leq \hat{\delta}$, the pattern is added to the set of testables $\mathcal{H}_{\text{test}}$. Since the number of testable patterns increased, the FWER estimate $\hat{\delta} \times |\mathcal{H}_{\text{test}}|$ might have changed, and line 12 checks whether the current threshold still guarantees control of the FWER at the target threshold α . If this is the case, nothing has to be done. On the other hand, if the FWER criterion is violated, the threshold $\hat{\delta}$ is currently too high, and has to be decreased. In the case of Pearson's χ^2 and Fisher's exact test, there are only a finite number of minimum p -values that can be attained. To be precise there exist at most $n+1/2$ such values (see Figure 3.2), and those values can be precomputed, such that upon decreasing, $\hat{\delta}$ will take on the next-lowest value. This results in a *decremental* search strategy, which has a very important implication: patterns can only become untestable when lowering the threshold, but never become testable. In other words: a pattern that has been shown to be untestable will always remain untestable when lowering $\hat{\delta}$, such that a pattern that has been deemed untestable does not have to be considered when the threshold is modified. The next step in Line 14 removes those patterns from $\mathcal{H}_{\text{test}}$ that are not testable at the new threshold $\hat{\delta}$. The lowering $\hat{\delta}$ and updating $\mathcal{H}_{\text{test}}$ are repeated until the FWER estimate $\hat{\delta} \times |\mathcal{H}_{\text{test}}|$ falls below the target FWER α .

After the threshold $\hat{\delta}$ has been updated, Lines 17 to 21 check if super-patterns of the current pattern can be pruned from the search space. This step has profound implications for the computational efficiency of the algorithm, as it greatly prunes the search space \mathcal{H} , such that commonly only a small fraction of patterns in \mathcal{H} have to be enumerated. To enable pruning, two observations are essential:

1. Since patterns are explored according to a pattern enumeration tree (see Fig. 3.4) it follows from the Apriori property in Proposition 3.4.1 that when moving deeper into the tree, the support $z_{\mathcal{S}}$ can only increase.
2. If the support exceeds $n - n_1$, we can see that the minimum p -value $\Phi(z_{\mathcal{S}})$ is mono-

3. Introduction to significant pattern mining

tonically increasing in z_S for Pearson’s χ^2 test and Fisher’s exact test (i.e. we are on the right of the rightmost minimum in Figure 3.3).

Taken together, we can conclude that if a pattern \mathcal{S} is untestable, and its support $z_S \geq n - n_1$, any pattern \mathcal{S}' containing \mathcal{S} is untestable as well, and hence can be pruned from the search space. Since patterns are organised in the pattern enumeration tree, we know that these super-patterns are the children of \mathcal{S} in the tree. In summary, if for a pattern \mathcal{S} holds that $\Phi(z_S) > \hat{\delta}$ and $z_S > n - n_1$, all children of \mathcal{S} in the search tree can be pruned from the search space. If a pattern cannot be pruned, the function `PROCESS_PATTERNS` is called recursively on all children of \mathcal{S} in the enumeration tree.

After the complete pattern enumeration tree has been explored and hence all patterns have been processed, the recursion of `PROCESS_PATTERN` stops and the set $\mathcal{H}_{\text{test}}$ contains the final set of testable patterns. From this, the Tarone-corrected significance threshold is computed as $\delta_{\text{Tar}} = \alpha/|\mathcal{H}_{\text{test}}|$, together with the set of testable patterns.

3.5. Accounting for pattern dependence: extension to permutation testing

The implementation of Tarone’s procedure is a major leap forward in addressing the problem of multiple hypothesis testing in significant pattern mining. Not only does it address the MHT problem, but it also gave a criterion for efficient pruning of the search space. Despite those advantages, a big challenge that effects the ability of significant pattern mining approaches to detect true positive associations is the presence of dependencies between patterns [see e.g. 184]. This dependence is caused by the high correlation between patterns due to the sub- and superset relationships among them. Many procedures to correct for multiple hypothesis testing assume independence between the tests, for example the Bonferroni correction, as well as Tarone’s correction introduced earlier in this chapter. If this assumption of independence is violated, those procedures tend to give more conservative estimations of the FWER, which in turn leads to a decrease in statistical power. Hence, methods that take dependencies between tests into account are required.

A common framework to estimate the FWER in the presence of dependent tests are permutation tests. In permutation testing, the distribution of the test statistic is inferred from the data rather than assumed to follow a parametric distribution (e.g. a χ^2 distribution for Pearson’s test, or the hypergeometric distribution for Fisher’s exact test). This can be achieved by conducting so-called Westfall-Young permutations [185], i.e. repeatedly permuting the label vector \mathbf{y} at random, a process that destroys any true association that is present in the data set. Hence, any statistically significant signal that occurs for a random permutation of the label inevitably corresponds to a false positive association.

More formally, for a fixed significance threshold δ , and a random permutation $\mathbf{y}^{(p)}$ of the label vector, we define p_{\min} as the smallest p -value obtained for any pattern \mathcal{S} , i.e. $p_{\min}^{(p)} = \min\{p_S \mid \mathcal{S} \in \mathcal{P}(\mathcal{I})\}$. If the smallest p -value obtained under the permutation is smaller than the threshold, $p_{\min}^{(p)} \leq \delta$, at least one false-positive association has occurred. If this procedure is repeated a sufficient number of times, that is mostly between 1’000

3. Introduction to significant pattern mining

and 10'000 random permutations of the label vector, the following can be used to estimate the FWER at significance threshold δ :

$$\text{FWER}(\delta) = \frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1} \left(p_{\min}^{(p)} \leq \delta \right), \quad (3.30)$$

where $p_{\min}^{(p)}$ corresponds to the smallest p -value found for the p -th permutation of the label vector, and $\mathbf{1}(\cdot)$ corresponds to the indicator function that evaluates to 1 if the argument is true, and to 0 otherwise. To control the FWER at a predefined significance level α , the per-hypothesis significance level δ is chosen as the lower α -quantile of the set of all smallest p -values $\left\{ p_{\min}^{(p)} \right\}_{p=1}^{n_p}$.

While permutation testing results in a tight control of the FWER in theory, it is impossible to realise for a significant pattern mining problem in practice due to computational restrictions. In the last sections we have seen that the task of enumerating and testing the complete hypothesis space \mathcal{H} is an incredibly demanding task on its own, and additionally generating permutations complicates this even further.

However, Terada et al. [160] presented a novel algorithm called FAST WESTFALL-YOUNG (FASTWY) that exploited Tarone's concept of the minimum p -value of discrete test statistics to make Westfall Young permutations feasible in a significant pattern mining setting. The method was further improved by Llinares-López et al. [161], giving rise to the **Westfall-Young light** algorithm, a fast and efficient algorithm for permutation testing in significant pattern mining. It is built on the observation that the margins of a contingency table, and hence the corresponding minimum p -value $\Phi(z_{\mathcal{S}})$, for a pattern \mathcal{S} are unaffected by permutations of the p -value. The **Westfall-Young light** algorithm constitutes the foundation for algorithms developed in this thesis, and this Section is devoted to establishing the basic principles. The method was originally described in [161], and slightly modified in [164] to incorporate (i) the minimum pattern indicator function, and (ii) an additional check for testability of the patterns which constitutes an additional algorithmic improvement. However, the method described in the latter publication is a special case for sequence-like patterns, and has to be modified slightly to be applicable to general patterns. Here, we introduce a hybrid of both approaches, that describes the method for the minimum pattern indicator function, including additional algorithmic improvements described in [164], and for any type of pattern \mathcal{S} , assuming that that the search space is organised in a pattern enumeration tree.

The main method is described in Algorithm 3.3. It expects three different types of input data. The first one is a data set $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ consisting of the $n \times d$ -dimensional, binary data matrix \mathbf{X} , and the n -dimensional binary label vector \mathbf{y} , that assigns every sample to one of two classes. As with the Tarone approach in the previous section, we assume without loss of generality that the class 1 constitutes the minority class with n_1 samples, such that there are $n - n_1$ samples in the class with label 0. We furthermore assume a target FWER α to be given, and a constant n_p indicating the number of permutations that should be conducted.

The main methods starts with the initialisation of three global variables in Lines 1-5, that

3. Introduction to significant pattern mining

Algorithm 3.3 Westfall-Young light: main.

Input: Data set $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, target family-wise error rate α , number permutations n_p

Output: Corrected significance threshold δ_{Perm}

```

1:  $\delta \leftarrow \alpha$ 
2: for  $p \in \{0, \dots, n_p\}$  do
3:    $\mathbf{y}^{(p)} \leftarrow$  random permutation of  $\mathbf{y}$ 
4:    $p_{\min}^{(p)} \leftarrow 1$ 
5: end for
6: PROCESS_PATTERNS( $\emptyset$ )
7:  $\delta_{\text{Perm}} \leftarrow \alpha$ -quantile of  $\left\{ p_{\min}^{(p)} \right\}_{p=1}^{n_p}$ 

```

is (i) the significance threshold $\hat{\delta}$ which is initialised to the target FWER; (ii) n_p random permutations of the class labels \mathbf{y} , as well as (iii) a vector that will store the smallest p -value across all n_p permutations, that is initialised to 1. This vector will be updated during the exploration of patterns, and is essential for the estimation of the FWER, as it will contain the smallest p -value that is obtained for any pattern \mathcal{S} for the p^{th} permutation of the labels.

The core of the method is implemented in PROCESS_PATTERN, a recursive function that explores the patterns organised in a pattern enumeration tree (see Figure 3.4) using a depth-first search strategy. For a given pattern \mathcal{S} , the support $z_{\mathcal{S}}$ and corresponding minimum p -value $\Phi(z_{\mathcal{S}})$ are computed to determine the testability of a pattern \mathcal{S} (note that the check for testability is not part of the original method [161], but was described in a later application [164]). If a pattern is untestable, we know that also all n_p permutations will be untestable, as the margins of the contingency table ($z_{\mathcal{S}}$, n_1 and n) are not affected by the permutation. Hence, if a pattern is untestable, it does not have to be accounted for during permutation testing, as, by definition of testability, none of the permutations will result in a false positive association at threshold $\hat{\delta}$. If a pattern is testable, its support is used to evaluate the p -values for all n_p permutations. This is done by calling the routine COMPUTE_PERMUTATION_PVALUES. This routine accesses the table count $a_{\mathcal{S}}^{(p)}$ and computes the p -value $p_{\mathcal{S}}(a_{\mathcal{S}}^{(p)})$ between the pattern and the p^{th} permutation for all $p = 1, \dots, n_p$. The p -value for the p^{th} permutation is stored in the vector $\mathbf{p}_{\min}^{(p)}$. In case of Fisher’s exact test, the p -values are pre-computed, as the computational complexity of evaluating the p -value for a single value $a_{\mathcal{S}}$ is the same as evaluating the p -values of all values $a \in [a_{\mathcal{S}}^{\min}, a_{\mathcal{S}}^{\max}]$.

After the p -values have been updated, the FWER is updated according to Equation 3.30 in Lines 5-9. In case the FWER estimate exceeds the target significance value α , the threshold δ is lowered. This lowering is identical to the one described in the previous section (Section 3.4.2), i.e. based on a pre-computation of the finite number of minimum p -values that can be attained for Pearson’s χ^2 test or Fisher’s exact test. The lowering of the threshold is repeated until the FWER is controlled at the target value α . Lines 11 to 15 check whether all patterns \mathcal{S}' containing \mathcal{S} , can be pruned from the search space \mathcal{H} . As for the classical Tarone approach, this pruning criterion follows from the Apriori property described in Proposition 3.4.1 and the monotonicity of the minimum p -value function

3. Introduction to significant pattern mining

Algorithm 3.4 Westfall-Young light: specific functions.

```

1: function PROCESS_PATTERNS( $\mathcal{S}$ )
2:   Compute the support  $z_{\mathcal{S}}$  and minimum  $p$ -value  $\Phi(z_{\mathcal{S}})$ 
3:   if  $\Phi(z_{\mathcal{S}}) \leq \hat{\delta}$  then
4:     COMPUTE_PERMUTATION_PVALUES( $z_{\mathcal{S}}$ )
5:     FWER( $\delta$ ) =  $\frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \delta)$ 
6:     while FWER( $\hat{\delta}$ )  $\geq \alpha$  do
7:       Decrease  $\hat{\delta}$ 
8:       FWER( $\delta$ ) =  $\frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \delta)$ 
9:     end while
10:  end if
11:  if not IS_PRUNABLE( $\mathcal{S}$ ) then
12:    for all super-patterns  $\mathcal{S}'$  of  $\mathcal{S}$  do
13:      PROCESS_PATTERN( $\mathcal{S}'$ )
14:    end for
15:  end if
16: end function
17:
18: function COMPUTE_PERMUTATION_PVALUES( $z_{\mathcal{S}}$ )
19:   If Fisher's exact test: Pre-compute  $p$ -values  $p_{\mathcal{S}}(a)$  for all  $a \in [a_{\mathcal{S}}^{\min}, a_{\mathcal{S}}^{\max}]$ 
20:   for  $p \in \{1, \dots, n_p\}$  do
21:     Compute  $a_{\mathcal{S}}^{(p)}$ 
22:      $p_{\min}^{(p)} \leftarrow \min(p_{\min}^{(p)}, p_{\mathcal{S}}(a_{\mathcal{S}}^{(p)}))$ 
23:   end for
24: end function

```

$\Phi(z_{\mathcal{S}})$ for $z_{\mathcal{S}} \geq n - n_1$. Details on the pruning condition can be found in Section 3.4.2).

After all patterns have been explored, $\{p_{\min}^{(p)}\}_{p=1}^{n_p}$ will contain the smallest p -value that has been obtained for any pattern in each of the n_p permutations. If any of the n_p entries fall below the significance threshold $\hat{\delta}$, then *at least one* false positive association did occur for that permutation. Hence, the final significance threshold obtained with permutation testing can be defined as the α quantile of values in $\{p_{\min}^{(p)}\}_{p=1}^{n_p}$.

3.6. Accounting for categorical covariates

One major drawback of Pearson's χ^2 test and Fisher's exact test is their incapability to correct for covariates. However, in significant pattern mining, it might be the case that significant results are confounded by an observable covariate, such that including the covariate into the model is indispensable if it is a goal to avoid false positive associations. While this is straightforward for linear models, where any type of covariate can be included as an additional additive factor, this is more complicated for models based on contingency tables. However, the Cochran-Mantel-Haenszel (CMH) test [108, 109] is able to correct for a categorical covariate. It has been shown that the computation of a minimum p -value is possible for the CMH test, and that an efficient pruning criterion can be derived, such that the CMH test can be used in combination with Tarone's procedure during

3. Introduction to significant pattern mining

TABLE 3.1.: A contingency table \mathcal{C}^r for the covariate class r in the CMH test. Derived from comparing the label vector \mathbf{y} with the binary pattern indicator function $\mathbf{g}_{\mathcal{S}}$ for samples in covariate class r .

	$\mathbf{g}_{\mathcal{S}}^r = 1$	$\mathbf{g}_{\mathcal{S}}^r = 0$	total row
$\mathbf{y}^r = 1$	$a_{\mathcal{S}}^r$	$n_1^r - a_{\mathcal{S}}^r$	n_1^r
$\mathbf{y}^r = 0$	$z_{\mathcal{S}}^r - a_{\mathcal{S}}^r$	$n^r - n_1^r - (z_{\mathcal{S}}^r - a_{\mathcal{S}}^r)$	$n^r - n_1^r$
column total	$z_{\mathcal{S}}^r$	$n^r - z_{\mathcal{S}}^r$	n^r

significant pattern mining, while enabling the correction for covariates [163]. This section is devoted to an introduction of the CMH test, including the testability and prunability criteria required for significant pattern mining with Tarone's correction, as described by Papaxanthos et al. [163].

3.6.1. The CMH test

In order to apply the CMH test, we assume the existence of a categorical covariate $\mathbf{c} \in \{1, \dots, r\}^n$. It assigns each sample in the data set to one of r distinct covariate groups. For each of the covariate groups, a separate contingency table is generated, as illustrated in Table 3.1, giving rise to a total of r different contingency tables per pattern \mathcal{S} . As a result, the previously (Person's χ^2 and Fisher's exact test) scalar table margins and table counts become r -dimensional vectors now, where each entry indicates the values for one of the r covariate classes, i.e. the support is denoted as $\mathbf{z}_{\mathcal{S}} = [z_{\mathcal{S}}^1, \dots, z_{\mathcal{S}}^r]$, the number of cases per covariate class as $\mathbf{n}_1 = [n_1^1, \dots, n_1^r]$, the number of samples per covariate class as $\mathbf{n} = [n^1, \dots, n^r]$, and the table counts as $\mathbf{a}_{\mathcal{S}} = [a_{\mathcal{S}}^1, \dots, a_{\mathcal{S}}^r]$. The total number of samples still is denoted as n , with $n = \sum_{i=1}^r n^i$. Note furthermore that the following equalities hold: $a_{\mathcal{S}} = \sum_{i=1}^r a_{\mathcal{S}}^i$, $n_1 = \sum_{i=1}^r n_1^i$, $z_{\mathcal{S}} = \sum_{i=1}^r z_{\mathcal{S}}^i$, where $z_{\mathcal{S}}$, $a_{\mathcal{S}}$ and n_1 are the support, table count and number of samples in the positive class for the complete data set, i.e. the quantities used for Pearson's χ^2 test and Fisher's exact test. The test statistic in the CMH test is the following quantity:

$$T_{\text{CMH}}(\mathbf{a}_{\mathcal{S}} \mid \mathbf{z}_{\mathcal{S}}, \mathbf{n}, \mathbf{n}_1) = \frac{\left(\sum_{i=1}^r a_{\mathcal{S}}^i - \frac{n_1^i z_{\mathcal{S}}^i}{n^i} \right)^2}{\sum_{i=1}^r \frac{n_1^i (n^i - n_1^i) z_{\mathcal{S}}^i (n^i - z_{\mathcal{S}}^i)}{(n^i)^2 (n^i - 1)}} \quad (3.31)$$

$$= \frac{\left(a_{\mathcal{S}} - \sum_{i=1}^r \frac{n_1^i z_{\mathcal{S}}^i}{n^i} \right)^2}{\sum_{i=1}^r \frac{n_1^i (n^i - n_1^i) z_{\mathcal{S}}^i (n^i - z_{\mathcal{S}}^i)}{(n^i)^2 (n^i - 1)}}. \quad (3.32)$$

Under the null hypothesis of independence it follows a χ^2 distribution with 1 degree of freedom, such that the p -value of the CMH test is computed as

$$p_{\mathcal{S}} = F_{\chi_1^2}(T_{\text{CMH}}(\mathbf{a}_{\mathcal{S}} \mid \mathbf{z}_{\mathcal{S}}, \mathbf{n}, \mathbf{n}_1)) \quad (3.33)$$

3. Introduction to significant pattern mining

where $F_{\chi_1^2}(\cdot)$ denotes the survival function of the χ_1^2 distribution. Interestingly, the p -value of the CMH test does not depend on the distribution of the table counts on the individual covariate classes, but solely the number of occurrences in positive cases. The CMH test can be interpreted as a meta-analysis of r disjoint data sets, where the idea is that if a significant pattern is associated to a covariate instead of the label, we can remove this spurious association by evaluating the association within each of the covariate classes separately. Within those classes, the association is presumably small, and hence the CMH test will not deem those patterns significant.

3.6.2. Significant pattern mining with Tarone's correction and the CMH test

In order to conduct significant pattern mining with Tarone's procedure as detailed in Algorithm 3.2, two criteria have to be fulfilled:

1. a minimum p -value has to be computed to evaluate testability, and
2. a criterion to check whether a pattern can be pruned from the search space has to be established.

Papaxanthos et al. [163] showed that both concepts, *testability* and *prunability* of patterns can be computed efficiently for the CMH test, such that it can be integrated into Algorithm 3.2. We give here a brief summary of their results.

Minimum p -value of CMH test

The survival function $F_{\chi_1^2}$ is a monotonically-decreasing function in the test statistic $T_{\text{CMH}}(\mathbf{a}_{\mathcal{S}} | \mathbf{z}_{\mathcal{S}}, \mathbf{n}, \mathbf{n}_1)$, and hence the p -value is minimised if the test statistic is maximised, and we denote the maximal test statistic as T_{CMH}^{\max} . This is achieved when the table count $a_{\mathcal{S}}$ takes on its maximum value $a_{\mathcal{S}}^{\min}$, or its minimum value $a_{\mathcal{S}}^{\max}$. Analogously to the tests discussed before, for each table $i = 1, \dots, r$ the table count $a_{\mathcal{S}}^i$ takes on values between $a_{\mathcal{S}}^{i,\min} = \max(0, n_1^i + z_{\mathcal{S}}^i - n^i)$ and $a_{\mathcal{S}}^{i,\max} = \min(z_{\mathcal{S}}^i, n_1^i)$. This implies that the global minimum is achieved at $a_{\mathcal{S}}^{\min} = \sum_{i=1}^r a_{\mathcal{S}}^{i,\min}$, and the global maximum is achieved at $a_{\mathcal{S}}^{\max} = \sum_{i=1}^r a_{\mathcal{S}}^{i,\max}$. Hence, the p -value of the CMH test reaches its minimum value at

$$\begin{aligned} p_{\mathcal{S}} &= F_{\chi_1^2}(T_{\text{CMH}}^{\max}) \\ &= F_{\chi_1^2}(\max(T_{\text{CMH}}(a_{\mathcal{S}}^{\min} | \mathbf{z}_{\mathcal{S}}, \mathbf{n}, \mathbf{n}_1), T_{\text{CMH}}(a_{\mathcal{S}}^{\max} | \mathbf{z}_{\mathcal{S}}, \mathbf{n}, \mathbf{n}_1))), \end{aligned} \quad (3.34)$$

and this value can be computed efficiently in $O(r)$ time for each pattern \mathcal{S} .

3.6.2.1. Pruning condition for the CMH test

We have seen that the support $z_{\mathcal{S}}$ of pattern \mathcal{S} gives an efficient pruning condition in Pearson's χ^2 test and Fisher's exact test, and the underlying strategy is to show that no pattern \mathcal{S}' containing \mathcal{S} has a smaller minimum p -value, i.e. $\Phi(z_{\mathcal{S}}) \leq \Phi(z_{\mathcal{S}'})$ for any $\mathcal{S} \subseteq \mathcal{S}'$. This holds true for all $z_{\mathcal{S}}$ that satisfy $n - n_1 \leq z_{\mathcal{S}} \leq n$, i.e. when the support lies in the range in which the minimum p -value function $\Phi(z_{\mathcal{S}})$ is monotonically increasing.

3. Introduction to significant pattern mining

For the CMH test there does not exist such a monotonicity criterion of the minimum p -value function.

Papaxanthos et al. [163] have shown that for all patterns that are potentially prunable, i.e. those patterns that satisfy $z_{\mathcal{S}}^i \geq n^i - n_1^i$, for all values $i = 1, \dots, r$, a monotonic lower bound $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}})$ to the minimum p -value function exists, i.e. $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}}) \leq \Phi(\mathbf{z}_{\mathcal{S}})$, which is referred to as the lower envelope of the minimum p -value. This envelope function depends on the pattern \mathcal{S} through its r -dimensional support vector $\mathbf{z}_{\mathcal{S}}$. It was shown that (i) this envelope is monotonic and hence can be used to prune the search space during significant pattern mining, and (ii) that the envelope can be computed efficiently in $O(r \log(r))$ time. While the complete proof of the prunability criterion is beyond the scope of this thesis, we outline the main steps in the following, and refer the interested reader to the supplementary material of the original publication [163].

The lower envelope of a pattern \mathcal{S} is defined as $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}}) = \{\Phi(\mathbf{z}_{\mathcal{S}'}) \mid \mathcal{S}' \supseteq \mathcal{S}\}$, i.e. the smallest minimum p -value attained by any pattern \mathcal{S}' fully containing \mathcal{S} . This lower envelope trivially satisfies the following monotonicity: if $\mathcal{S} \subseteq \mathcal{S}'$, then $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}}) \leq \tilde{\Phi}(\mathbf{z}_{\mathcal{S}'})$. Given the lower envelope, the pruning criterion of the CMH test follows analogously to Pearson's χ^2 or Fisher's exact test: Given a testability threshold $\hat{\delta}$ and a pattern \mathcal{S} that satisfies

1. $z_{\mathcal{S}}^i \geq n^i - n_1^i$ for all $i = 1, \dots, r$, and
2. $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}}) > \hat{\delta}$,

it follows from the definition of the lower envelope that for any pattern \mathcal{S}' that is a super-pattern of \mathcal{S} , i.e. for which $\mathcal{S} \subset \mathcal{S}'$ holds, its minimum p -value $\Phi(\mathbf{z}_{\mathcal{S}'})$ exceeds the significance threshold $\hat{\delta}$. Hence \mathcal{S}' is by definition non-testable at threshold $\hat{\delta}$. In this case, all patterns \mathcal{S}' that satisfy $\mathcal{S} \subset \mathcal{S}'$ can be pruned from the search space.

Due to the Apriori property in Proposition 3.4.2, it follows for the i -th component in the support vector that $z_{\mathcal{S}}^i \leq z_{\mathcal{S}'}^i$, for all $\mathcal{S} \subseteq \mathcal{S}'$, such that the lower envelope function can be rewritten as $\tilde{\Phi}(\mathbf{z}_{\mathcal{S}}) = \min_{\mathbf{z}_{\mathcal{S}'} \geq \mathbf{z}_{\mathcal{S}}} \Phi(\mathbf{z}_{\mathcal{S}'})$, where the inequality $\mathbf{z}_{\mathcal{S}'} \geq \mathbf{z}_{\mathcal{S}}$ holds component-wise. Naively computing the lower envelope by evaluating all possible configurations of the r contingency tables for each potentially prunable pattern is computationally infeasible. Papaxanthos et al. [163] derive an algorithm that solves the discrete optimisation problem

$$\min_{\mathbf{z}_{\mathcal{S}'} \geq \mathbf{z}_{\mathcal{S}}} \Phi(\mathbf{z}_{\mathcal{S}'}) \quad (3.35)$$

in $O(r \log(r))$ time. This result is stated in the following theorem.

Theorem 3.6.1 (Papaxanthos et al. (2016), theorem 2). *Let \mathcal{S} be a pattern, and $\mathbf{z}_{\mathcal{S}}$ its r -dimensional support vector, where the i^{th} entry $\mathbf{z}_{\mathcal{S}}^i$ counts the pattern in samples belonging to covariate class i , $i = 1, \dots, r$, and r being the number of states of the categorical covariate. We define $\beta_l^i = \frac{\mathbf{z}_{\mathcal{S}}^i}{n^i} \frac{n^i}{n^i}$ and $\beta_r^i = \frac{\mathbf{z}_{\mathcal{S}}^i}{n^i} \left(1 - \frac{n^i}{n^i}\right)$ for each $i = 1, \dots, r$. Furthermore, let π_l and π_r be permutations that order the sets $\{\beta_l^i \mid i = 1, \dots, r\}$ and $\{\beta_r^i \mid i = 1, \dots, r\}$ in ascending order. The solution $\hat{\mathbf{z}}$ to the discrete optimisation problem in Equation 3.35 that induces the lower envelope of pattern \mathcal{S} satisfies one of the following conditions:*

3. Introduction to significant pattern mining

1. $\hat{\mathbf{z}}^{\pi_l(j)} = \mathbf{z}_S^{\pi_l(j)}$ for $j \leq \rho$ and $\hat{\mathbf{z}}^{\pi_l(j)} = 0$ for $j > \rho$, $\rho \in \{1, \dots, r\}$
2. $\hat{\mathbf{z}}^{\pi_r(j)} = \mathbf{z}_S^{\pi_r(j)}$ for $j \leq \rho$ and $\hat{\mathbf{z}}^{\pi_r(j)} = 0$ for $j > \rho$, $\rho \in \{1, \dots, r\}$

This implies that, in order to find the solution to the optimisation problem in Equation 3.35, the $2r$ different configurations of the vector $\hat{\mathbf{z}}$ and their corresponding minimum p -values have to be evaluated. These operations give rise to a linear runtime $O(r)$. However, the overall runtime is dominated by the sorting of the r -dimensional vectors β_l and β_r , an $O(r \log(r))$ operation, such that the total time complexity of computing the lower envelope equals $O(r \log(r))$.

3.7. Significant pattern mining in biology and medicine

The procedures introduced in this chapter describe statistical and computational concepts that address three different challenges in significant pattern mining, that is (i) the multiple hypothesis testing problem that arises naturally when analysing higher order interactions between features, (ii) the problem of dependencies between patterns, and (iii) the problem of correcting for potentially confounding variables. While pattern mining approaches initially arose in fields such as market basket analysis, they have proven useful in other areas as well, one of them being biomedical research.

The paper that first introduced the Tarone procedure into significant pattern mining did so to solve the problem of finding higher order interactions between transcription factors in yeast and human breast cancer transcriptomes [158]. As the significant pattern mining approaches rely on binary input data, i.e. each feature must be assigned either one of two discrete values (commonly 0 and 1, indicating presence/absence, activity/inactivity, above/below threshold). In the case of the transcriptomic data used in [158], and in their follow-up work [159], this has been achieved by assessing whether a gene is over-/under-expressed with respect to a baseline condition. Reformulating this problem in terms of a significant pattern mining problem lead to the discovery of novel higher-order interactions for both the study in yeast as well as the study of human breast cancer.

Another prominent field of application of Tarone's procedure is in the realm of genetic studies. Llinares-López et al. [164, 165] focussed on the analysis of mutational patterns in plant and human DNA, where patterns were defined as coherent regions of mutation on the DNA. The required binary representation of the genetic data could be achieved by assessing whether or not a genetic locus contains a mutation, where a mutation was considered as a deviation from the majority in the data set. Those applications lead to the discovery of novel genetic loci in both, plant genetics and human genetics for a case-control study of patients suffering from chronic obstructive pulmonary disorder (COPD).

A third area that recently adopted a significant pattern mining perspective is the field of time series analysis. Bock et al. [186] developed a significant pattern mining approach based on Tarone's procedure to tackle the problem of finding significant shapelets, i.e. short fragments within time series, that occur significantly more often in one class of time series compared to a second class of time series. While their approach is in general agnostic to the type of time series, they apply it to find significant shapelets that are indicative of

3. Introduction to significant pattern mining

a future sepsis event, based on time resolved measurements of heart and respiratory rates as well as blood pressure in the ICU.

3.7.1. Significant pattern mining for network-guided association studies

Significant pattern mining approaches were successfully incorporated into various biomedical tasks [158, 159, 164, 165, 186], as described above, one of them being genetic association studies [164, 165]. They have shown to successfully tackle the multiple hypothesis problems arising when testing combinations of features. Furthermore, by incorporating the CMH test, covariates could be integrated that correct for confounders which often inflate the results obtained from genetic association studies. However, those approaches so far only considered the effect of neighbouring mutations on the DNA. As introduced in the beginning of this thesis, current research indicates the importance of interactions between genetic markers to further explain the contribution of genetics to the development and progression of complex traits and diseases. At the same time, our biological prior knowledge of genetic interactions is increasing at unprecedented rates, captured in the form of molecular networks.

While different routes were taken to integrate network information into genetic analysis, such as network diffusion or network regularisation, the existing methods suffer from at least one of the following shortcomings:

- i. Genes in the network are represented by summary statistics, losing the resolution of individual genetic markers
- ii. Reliance on heuristics, such as greedy exploration of the hypothesis space, instead of exhaustively exploring the hypothesis space
- iii. The statistical association of the reported interactions to the phenotype is not assessed
- iv. Focus on linear interactions between genetic markers, while ignoring non-linear or heterogeneity effects
- v. Large computational complexity restricting the use to small data sets

In this thesis, we aim at addressing the above challenges and shortcomings of existing methods, by (i) combining genetic data with molecular networks and (ii) developing algorithms that rely on the concepts and procedures from significant pattern mining introduced in the previous sections to analyse them. We show that the significant pattern mining interpretation of network genetics is a promising addition to the toolbox of methods for the network analysis of genetic data.

4. Network-guided testing of local neighbourhoods

Gumpinger, A.C., Llinares-López, F., Roqueiro, D., Borgwardt, K.M. *Network-guided association mapping of local neighbourhoods*. Unpublished (2019).

In this chapter we introduce an approach to find interactions between genes that are associated with a complex trait of interest. The approach is based on the principles of significant pattern mining that were presented in the preceding chapters. The problem of finding interactions between genes can be formulated as a significant pattern mining problem by identifying the interactions between genes as patterns. Since the hypothesis space of all interactions \mathcal{H} between genes scales exponentially with the number of genes, i.e. $|\mathcal{H}| = 2^d$, where d is the number of genes, or features, and d commonly lies in the tens of thousands, an exhaustive exploration of the search space exceeds our computational and statistical capacities. Hence, we include biological prior knowledge in the form of molecular networks, in which nodes constitute genes, and edges constitute interactions of various forms between them. In that sense, the network captures our prior beliefs about the features, and prioritises relationships between feature pairs. We use those networks as *guides* to our search for interactions. The advantages of this approach are twofold: (i) it reduces the number of interactions, addressing the statistical and computational bottlenecks, and (ii) using interactions within a network results in biologically meaningful sets of genes for downstream analyses.

There exist various ways how networks can be used to guide the search for significant interactions. A compelling strategy is to analyse all possible connected sub-graphs within a network. However, exhaustively enumerating all such sub-graphs scales exponentially in the number of edges in a network, and hence remains computationally and statistically inaccessible in large molecular networks. In order to circumvent the associated exponential runtime of the sub-graph enumeration, we use the concept of k -hop neighbourhoods in networks. Neighbourhoods have shown to contain valuable information across many different tasks, for example in the Weisfeiler-Lehman procedure to test whether two labelled graphs are isomorphic [187]. The concept of local neighbourhoods is also central in the field of graph and node classification with graph-convolutional networks [e.g. 188], it underlies network propagation [65], and has shown success in the field of cancer-gene identification [32]. Hence we consider neighbourhoods in graphs as promising candidate patterns for network guided association studies.

This section is organised as follows: we start with a theoretical introduction to the problem of identifying significant neighbourhoods in networks. This introduction comprises a formal problem statement, that introduces the data, and the types of patterns our proposed approach evolves around. We present how we can solve the problem of identifying

4. Network-guided testing of local neighbourhoods

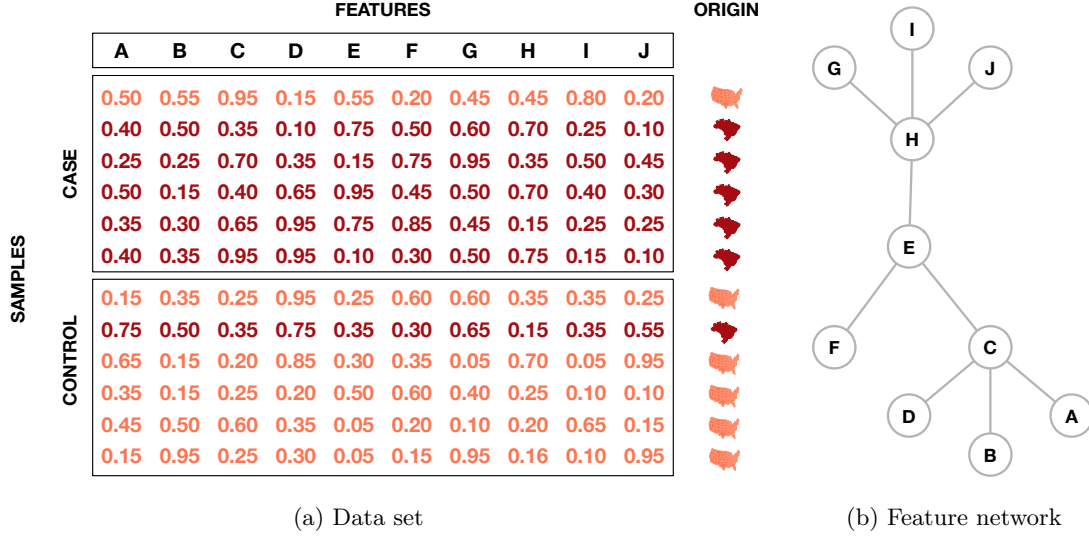


FIGURE 4.1.: Example of data sets used in \mathfrak{tNeAT} approach. (a) Example of the data matrix \mathbf{X} for 12 samples and 10 features. Each sample is assigned to the case or control class, resulting in the binary label vector \mathbf{y} . An optional covariate vector \mathbf{c} , here corresponding to the origin of the sample, can be included. In our case, the features correspond to genes, and each measurement in \mathbf{X} to the fraction of minor alleles within the gene. (b) Example of a graph that represents relationships between features, such as physical interactions between proteins defined by the genes in (a).

neighbourhoods that are significantly associated to the trait by interpreting it as a significant pattern mining problem which entails reformulating the problem such that it fits the premises presented in the preceding sections. Following this description of concepts and principles, we introduce our novel method \mathfrak{tNeAT} (*thresholded Neighbourhood Aggregation with Testing*) and its Westfall-Young permutation based counterpart $\mathfrak{tNeAT-WY}$ (*thresholded Neighbourhood Aggregation with Testing using Westfall-Young permutations*) in detail. We continue this chapter with an application to simulated data, and an application to a data set of patients suffering from migraine. We conclude with a discussion of the method, its usefulness as well as an outlook for further development.

4.1. Finding significant neighbourhoods in networks

4.1.1. Problem statement and notation

Consider a data set that contains data for n samples, where each of those samples comes from one of two phenotypic classes. We store this class assignment in a binary label vector $\mathbf{y} \in \{0, 1\}^n$. We furthermore have measurements of d features for each sample, and we denote the set of feature indices as \mathcal{J} . We assume all measurements to be normalised to the range $[0, 1]$, and to be stored in an $n \times d$ dimensional data matrix $\mathbf{X} \in [0, 1]^{n \times d}$. Additionally, each of the n samples is assigned to one of k discrete covariate classes, and this information is stored in a covariate vector $\mathbf{c} \in \{1, \dots, r\}^n$, such that the i^{th} entry \mathbf{c}_i contains the covariate class of sample i (see Fig. 4.1a for an example).

4. Network-guided testing of local neighbourhoods

Assume there is an additional layer of information readily available, that is information on relationships between the d features, that can be represented in form of a graph $\mathcal{G} = (V, E)$ with vertex set V and edge set E in the following way: in \mathcal{G} , every vertex in v corresponds to one of the d features measured in the data set \mathbf{X} , and an edge between two vertices u and v with $u, v \in V$, denoted as tuple $(u, v) \in E$ indicates the presence of a specific relationship between two adjacent features. Hence $E = \{(u, v) \mid u, v \in V\}$, and we denote the number of edges in the graph with m . Our goal is to find sets of features, that (i) show a statistically significant association to the binary phenotype y , and (ii) form a k -hop neighbourhood in the graph \mathcal{G} . An example of such a data set is illustrated in Figure 4.1.

4.1.2. Patterns as local neighbourhoods in a graph

In the introduction to significant pattern mining, all concepts were presented for arbitrary patterns \mathcal{S} . As we have seen, a pattern is any discrete substructure of the features \mathcal{J} , and hence an element of the power set of \mathcal{J} , i.e. $\mathcal{S} \in \mathcal{P}(\mathcal{J})$. The space of all patterns that we consider in an analysis was referred to as the search space, or hypothesis space, denoted by \mathcal{H} . We denote the full hypothesis space, that contains all possible patterns in the power set of features, as $\mathcal{H}_{\text{full}} = \mathcal{P}(\mathcal{J})$.

In our approach, where we assume the existence of a graph \mathcal{G} that represents relationships between features, we use this additional type of information as a guide to choose the features constituting a pattern \mathcal{S} , and hence to design our hypothesis space $\mathcal{H}_{\mathcal{G}}$. In other words, all patterns that do not obey our network-based design criteria are removed from the hypothesis space of *all* possible patterns $\mathcal{H}_{\text{full}}$.

A straightforward way to define network-based patterns is the following: instead of mining all possible patterns, we could restrict ourselves to those that form connected components in the network, thereby reducing the problem of mining all patterns in $\mathcal{H}_{\text{full}}$ to the problem of mining sub-graphs in a network. However, the enumeration of all possible connected sub-graphs is computationally intractable for large networks.

For this reason, we resort to mining k -hop neighbourhoods in a network, where $k \in \mathbb{N}_{\neq 0}^0$ is arbitrary. More formally, given the graph $\mathcal{G} = (V, E)$ of d vertices and m edges as described above, we denote a k -hop neighbourhood at anchor or source node v as \mathcal{N}_v^k . It corresponds to the set of all nodes that are reachable from v with at most k edges. It can be defined recursively as

$$\mathcal{N}_v^k = \mathcal{N}_v^{k-1} \cup \{u \mid (u, w) \in E, w \in \mathcal{N}_v^{k-1}\} \quad (4.1)$$

where the 0-hop neighbourhood corresponds to the anchor node itself, i.e. $\mathcal{N}_v^0 = \{v\}$. Those neighbourhoods from the central entity of interest in our proposed pattern mining approach.

4.1.3. Finding neighbourhoods by significant pattern mining

We aim at finding k -hop neighbourhoods in a network that are significantly associated with a binary trait of interest. In order to improve statistical power while controlling the number of false positive associations we turn to the framework of significant pattern mining, that is Tarone’s procedure and the concept of testability of discrete test statistics.

4. Network-guided testing of local neighbourhoods

Those approaches were described in great detail in Chapter 3. In essence, they exploit an observation made by Tarone [183] that allows to find a less stringent correction factor in the Bonferroni correction, while still guaranteeing control of the family-wise error rate. While the correction factor in the Bonferroni correction corresponds to the total number of hypotheses tested, in Tarone’s procedure this correction factor is reduced to a subset of all hypotheses, called the set of testable hypotheses (see Section 3.3.4.2). The set of testable hypotheses exists for discrete test statistics, and can be computed by removing those hypotheses that have a minimum p -value (see Section 3.3.4.1) above the significance threshold, and hence cannot become testable, from the hypothesis space. This results in an improved Bonferroni threshold, called Tarone threshold δ_{Tar} that is always greater or equal to the classical Bonferroni correction δ_{Bonf} , increasing statistical power to detect true positive associations. We showed how to integrate significant pattern mining and Tarone’s procedure for the Pearson’s χ^2 test and Fisher’s exact test in Section 3.4, and that permutation testing with Westfall-Young permutations can be enhanced by incorporating them with Tarone’s testability criteria in Section 3.5. We furthermore introduced how Tarone’s procedure can be integrated with the Cochran-Mantel-Haenszel test, an extension of Pearson’s χ^2 test that enables the correction for categorical covariates during testing (see Section 3.6). For the remainder of this section, we assume those concepts and ideas to be known.

To enable an efficient exploration of the neighbourhoods, the problem has to be formulated in a way that it fits the premises of significant pattern mining. In order to achieve this, the following two requirements have to be satisfied:

- i. We can define a binary pattern indicator function $g_S(\cdot)$ that indicates whether a pattern is present for a given sample, and
- ii. the patterns can be organised in a pattern enumeration tree that allows for efficient pruning of the hypothesis space.

We show in the following how both requirements can be fulfilled in our problem setting. More specifically, we will show how the design of the pattern indicator function allows us to test for genetic heterogeneity, a potential source of the missing heritability of genetic analysis (see Section 2.4).

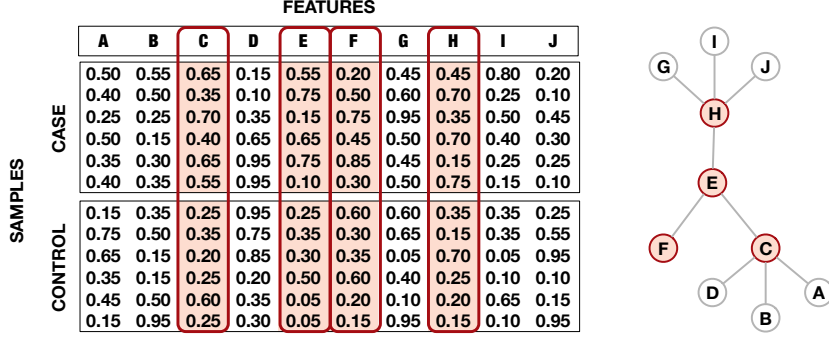
4.1.3.1. Formal definition of a pattern and a pattern indicator function

In the problem statement (Section 4.1.1) we saw that the data matrix \mathbf{X} contains measurements in the range $[0, 1]$, e.g. corresponding to the fraction of minor alleles per gene. Significant pattern mining approaches require a binary representation of each pattern. To achieve this binary representation, we define a set of thresholds \mathcal{T} in the range of $[0, 1]$, where each threshold increments the previous one by a value of 0.05. This means, $\mathcal{T} = \{0.00, 0.05, 0.10, \dots, 0.90, 0.95, 1.00\}$. These thresholds are used to binarise the data set \mathbf{X} , such that

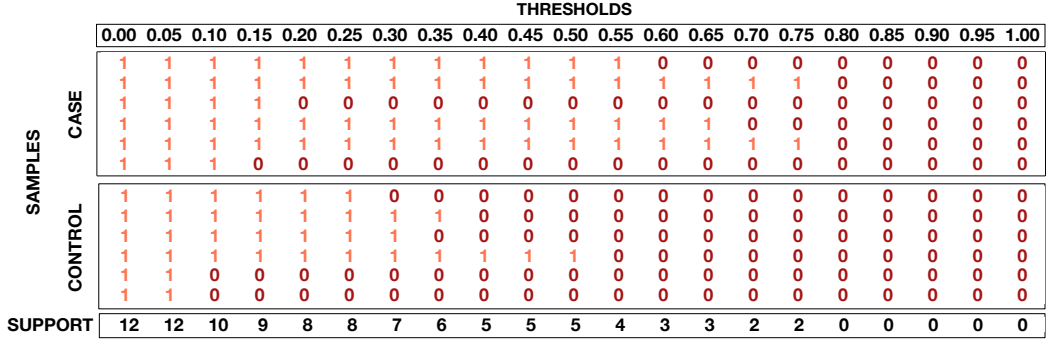
$$\mathbf{X}_{i,j}^{t_i} = \mathbf{1}(\mathbf{X}_{i,j} \geq t_i), \quad (4.2)$$

where $t_i \in \mathcal{T}$ and $\mathbf{1}(\cdot)$ is the indicator function that returns 1 if the statement is true, 0 otherwise. This procedure yields a binary data matrix for each of the thresholds in \mathcal{T} .

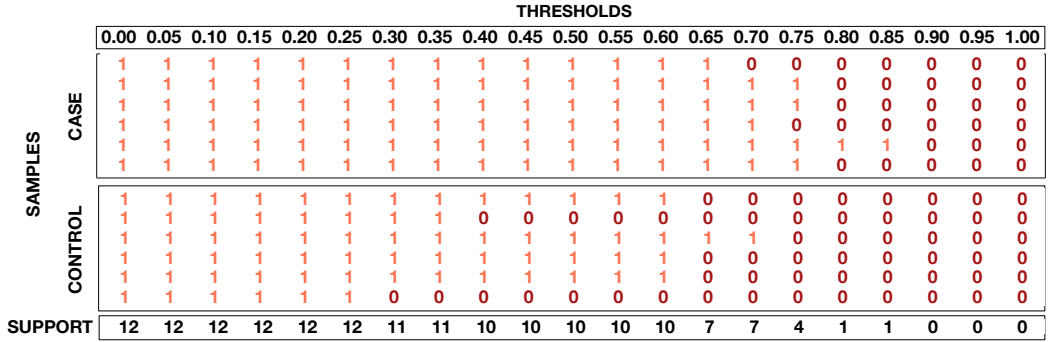
4. Network-guided testing of local neighbourhoods



(a) Example of neighbourhood



(b) Pattern indicator g_S^{\max} , where $S = \mathcal{N}_E^{0,t}$.



(c) Pattern indicator g_S^{\max} , where $S = \mathcal{N}_E^{1,t}$.

FIGURE 4.2.: Example of the maximum pattern indicator g_S^{\max} , where $S = \mathcal{N}_v^{k,t}$ at varying thresholds. (a) Example of data matrix \mathbf{X} and feature network \mathcal{G} . The 1-hop neighbourhood of anchor vertex E , denoted as \mathcal{N}_E^1 is highlighted in the data matrix and in the network on the left. Figures (b) and (c) show the values of the pattern indicator function of the patterns $\mathcal{N}_E^{0,t}$, i.e. the anchor vertex E , and the patterns $\mathcal{N}_E^{1,t}$, i.e. the 1-hop neighbourhoods, at all binarisation thresholds $t \in \mathcal{T}$, respectively. Supports $z_v^{k,t}$ are indicated with the row at the bottom.

4. Network-guided testing of local neighbourhoods

TABLE 4.1.: Contingency table for a pattern $\mathcal{S} = \mathcal{N}_v^{k,t}$. In case of Fisher’s exact test and Pearson’s χ^2 test this contingency table is created once for every pattern. In case of the CMH test, this contingency table is built r times for each pattern, one for each covariate class in the data set (see Section 3.6.1 for details).

	$g_{\mathcal{S}} = 1$	$g_{\mathcal{S}} = 0$	total row
$y = 1$	$a_v^{k,t}$	$n_1 - a_v^{k,t}$	n_1
$y = 0$	$z_v^{k,t} - a_v^{k,t}$	$n - n_1 - (z_v^{k,t} - a_v^{k,t})$	$n - n_1$
column total	$z_v^{k,t}$	$n - z_v^{k,t}$	n

We defined a pattern \mathcal{S} as a k -hop neighbourhood \mathcal{N}_v^k of vertex v in the network \mathcal{G} (Section 4.1.2). The threshold t introduces an additional parameter, such that we write $\mathcal{N}_v^{k,t}$ for a pattern, with v indicating the anchor vertex, k the neighbourhood, and t the binarisation threshold. For notational simplicity, we continue to denote a pattern as \mathcal{S} in the pattern indicator function. Given the binary matrix \mathbf{X}^t as defined in Equation 4.2, where $t_i \in \mathcal{T}$, we define a pattern indicator function $g_{\mathcal{S}}(\mathbf{X}_i)$ for the i -th sample as follows:

$$g_{\mathcal{S}}(\mathbf{X}_i) = \max \left(\left\{ \mathbf{X}_{i,j}^{t_i} \mid j \in \mathcal{S}, \text{ where } \mathcal{S} = \mathcal{N}_v^k \right\} \right). \quad (4.3)$$

This implies that a pattern is present for sample i at threshold t_i , if at least one feature in the pattern \mathcal{S} exceeds the threshold t_i . An example of this binarisation of a pattern at various thresholds is illustrated in Fig. 4.2: Subfigure 4.2a contains a data matrix \mathbf{X} with the 1-hop neighbourhood of anchor vertex E highlighted in red. The Subfigures 4.2b and 4.2c contain the binarisation of the anchor vertex and the 1-hop neighbourhood, respectively, for all thresholds $t_i \in \mathcal{T}$. This binary pattern indicator function can be used to set up a contingency table as in Table 4.1, which in turn is used to compute a p -value of association.

Remark 4.1.1 (Genetic heterogeneity). *We would like to emphasise at this point that the choice of this encoding allows us to incorporate the concept of **genetic heterogeneity** into the model. As explained in greater detail in Section 2.4, genetic heterogeneity refers to the phenomenon that the same phenotype might be caused by different variants for different individuals, and this mode of interaction might be instrumental to explain the **missing heritability** of genetic analyses. By encoding neighbourhoods as illustrated in Equation 4.3, the pattern will be considered present if a single gene within a neighbourhood exhibits mutation rates above the threshold. This holds true irrespective of which or how many genes in the neighbourhood show this behaviour.*

4.1.3.2. Construction of a pattern enumeration tree for an efficient pruning of the search space

Before constructing a pattern enumeration tree, we formalise the hypothesis space associated to our problem. By enumerating all k -hop neighbourhoods $\mathcal{N}_v^{k,t}$ of all vertices $v \in V$

4. Network-guided testing of local neighbourhoods

in a network, at all binarisation thresholds $t \in \mathcal{T}$, the hypothesis space becomes

$$\mathcal{H}_G = \left\{ \mathcal{N}_v^{k,t} \mid v \in V, t \in \mathcal{T}, k \in \mathbb{N}_+^0 \right\}. \quad (4.4)$$

This emphasises the dependency of the size of the hypothesis space \mathcal{H}_G on three different variables: the number of nodes in the network, $|V| = d$, the number of thresholds, $|\mathcal{T}| = 21$ as well as the largest possible neighbourhood, represented by the value k . This value is upper-bounded by the diameter of the network, i.e. the length of the longest shortest path within the network.

Analogously to the significant pattern mining approaches introduced in Section 3.4.2, our goal is to organise the hypothesis space \mathcal{H}_G in a way that allows for an efficient pruning when using the concept of the minimum p -value. This requires organising the patterns in a pattern enumeration tree such that the support $z_v^{k,t}$ of a pattern is monotonically increasing when moving deeper into the tree. This enables pruning of the search space analogously to the procedure described in Section 3.4.2 for Pearson's χ^2 and Fisher's exact test, and in Section 3.6.2.1. In the case of the k -hop neighbourhoods, there exist two important monotonicities of the support $z_v^{k,t}$ for anchor node v .

- i. monotonicity with respect to parameter k , i.e. $z_v^{k,t} \leq z_v^{k+1,t}$, i.e. the support is increasing upon *growing* the neighbourhood from k to $k+1$.
- ii. monotonicity with respect to threshold t_i , i.e. $z_v^{k,t_i} \geq z_v^{k,t_{i+1}}$, i.e. the support is increasing upon *lowering* the threshold from t_{i+1} to t_i .

The first monotonicity follows from the Apriori property (Proposition 3.4.1), as for any neighbourhood the following sub-/super-pattern relationship holds: $\mathcal{N}_v^{k,t} \subseteq \mathcal{N}_v^{k+1,t}$. We see this also for the example illustrated in Figure 4.2, where the 0-hop neighbourhood always has lower supports (see Figure 4.2b) at each threshold than the 1-hop neighbourhood (see Figure 4.2c).

The second monotonicity becomes apparent when considering an individual measurement of feature j for sample i , i.e. $\mathbf{X}_{i,j}$. If this measurement exceeds threshold t_i , i.e. $\mathbf{X}_{i,j} \geq t_i$, it will also exceed any threshold $t_j \leq t_i$, and hence

$$\mathbf{X}_{i,j}^{t_i} = 1 \implies \mathbf{X}_{i,j}^{t_j} = 1 \quad \forall t_j \leq t_i, \quad t_i, t_j \in \mathcal{T}. \quad (4.5)$$

This monotonicity is illustrated in Figures 4.2b and 4.2c, where we can also observe that the support of each pattern $\mathcal{N}_v^{k,t}$ increases with decreasing thresholds $t \in \mathcal{T}$.

Remark 4.1.2 (Monotonicity criteria for the CMH test). *We would like to note that both monotonicity criteria hold true for all discrete tests described in the introduction, that is Fisher's exact test, Pearson's χ^2 test and the CMH test. While for the first two, the support of a pattern $z_v^{k,t}$ is a scalar, in case of the CMH test it is a vector of length r , where r is the number of discrete covariate classes (see Section 3.6 for details). In this case, the monotonicity criteria hold element-wise, following the same argumentation as for Pearson's and Fisher's tests.*

Those monotonicity criteria can be used to explore and prune the hypothesis space. This is illustrated in Figure 4.3 for all patterns grown from an individual anchor vertex v . Given

4. Network-guided testing of local neighbourhoods

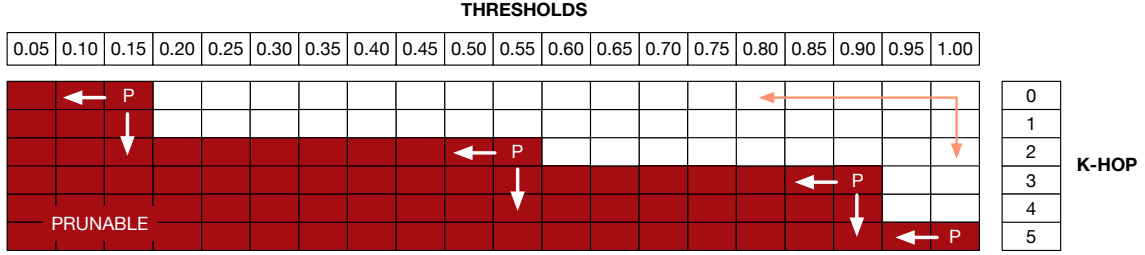


FIGURE 4.3.: Two pattern enumeration schemes for a single anchor vertex v , indicated by the two orange arrows. Each field in the table represents a pattern $\mathcal{N}_v^{k,t}$. As soon as a pattern $\mathcal{N}_v^{k_p,t_p}$ can be pruned (indicated with a white P on red background), all patterns with $k \geq k_p$ and $t \leq t_p$ can be pruned from the search space as well. This is indicated with white arrows on red background. The dark red fields indicate the part of the search space that can be pruned.

v , patterns can be enumerated either by first exploring all k -hop neighbourhoods at a fixed threshold t , or by exploring all thresholds $t \in \mathcal{T}$ at a fixed value of k . These directions are indicated with the orange arrows. As soon as a pattern can be pruned from the search space, the two monotonicity criteria listed above allow us to remove all super-patterns from the search space, where a super-pattern is a pattern that either exhibits a higher value of k , or a lower value of t .

We implement our method **tNeAT** using the first exploration scheme, that is by fixing a threshold and iterating the neighbourhood-depth k . This is beneficial from a computational perspective, as the computational bottleneck of the exploration is the aggregation of neighbourhoods. By growing them successively, we can use information from the k hop-neighbourhood to compute the pattern indicator function of the $k + 1$ -hop neighbourhood. We furthermore process all anchor vertices in parallel, corresponding to a breadth-first like search strategy of all nodes in the network. The next section details our novel proposed algorithms.

4.2. The tNeAT method

Our proposed approach *thresholded Neighbourhood Aggregation and Testing* (**tNeAT**) efficiently combines key concepts from significant pattern mining discussed before. In essence, it is a novel algorithm to find feature sets that (i) form neighbourhoods in a feature-graph \mathcal{G} , and that (ii) are statistically significantly associated to a phenotype of interest. The association is evaluated for different instances of the binarised data, when using a set of thresholds \mathcal{T} . We describe the method in Algorithm 4.1. The pseudocode is kept agnostic to the type of test used, where possible options are Pearson’s χ^2 test, Fisher’s exact test and, if a categorical covariate is available, the CMH test. We discuss modifications to the code that are necessary to accommodate the specific tests in the description of the code.

4. Network-guided testing of local neighbourhoods

Algorithm 4.1 tNeAT (thresholded Neighbourhood Aggregation with Testing), main body

Input: Data matrix \mathbf{X} , class labels \mathbf{y} , covariate vector \mathbf{c} (optional), network $\mathcal{G} = (V, E)$, target family-wise error rate α

Output: corrected significance threshold δ_{Tar} , set of significant neighbourhoods $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}})$

- 1: Initialise **global** $\hat{\delta}_{\text{tar}} \leftarrow 1$, and **global** $\mathcal{H}_{\text{test}} \leftarrow \emptyset$,
 - 2: Initialise thresholds $\mathcal{T} \leftarrow \{0.05, 0.1, \dots, 0.95, 1.0\}$;
 - 3: Initialise pruning marker $\mathbf{x}_{\text{prune}} \leftarrow \{d - 1\}^d$
 - 4: **for** t in \mathcal{T} .reversed **do**
 - 5: $\mathbf{X}^t \leftarrow$ binarise \mathbf{X} at threshold t
 - 6: $k \leftarrow 0$
 - 7: Compute supports $\{z_v^{k,t}\}_{v \in V}$
 - 8: **while** $\sum_{i=1}^d \mathbf{1}(\mathbf{x}_{\text{prune}}[i] = 0) > 0$ and not all k explored **do**
 - 9: $\mathbf{x}_{\text{prune}} \leftarrow$ PROCESS_NEIGHBOURHOOD($\{z_v^{k,t}\}_{v \in V}, k, \mathbf{x}_{\text{prune}}$)
 - 10: $k \leftarrow k + 1$
 - 11: Update supports $\{z_v^{k,t}\}_{v \in V}$
 - 12: **end while**
 - 13: **end for**
 - 14: $\delta_{\text{Tar}} \leftarrow \frac{\alpha}{|\mathcal{H}_{\text{test}}|}$
 - 15: $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}}) \leftarrow \{\mathcal{N}_v^{k,t} \mid \mathcal{N}_v^{k,t} \in \mathcal{H}_{\text{test}} \text{ and } p_v^{k,t} \leq \delta_{\text{Tar}}\}$
-

Main body

The main routine of the tNeAT method is outlined in Algorithm 4.1. It starts with the initialisation of the global variables, that is the testability threshold $\hat{\delta}$ is set to one, and the set of testable hypotheses $\mathcal{H}_{\text{test}}$ to an empty list. In Line 2 we set the binarisation thresholds \mathcal{T} that cover the range from 0.05 to 1.0 in 0.05 increments. The data set will be binarised at those thresholds, and they determine the granularity of testing. Line 3 sets a vector with d entries, corresponding to the d anchor vertices in the graph. This vector is essential to the pruning strategy: its i -th entry will store the value of k at which the preceding pattern anchored at vertex i became untestable. The vector is initialised to $d - 1$, as this is the largest possible k -hop value for any arbitrary graph (to be precise, the line graph). Line 4 to 13 contain the heart of the algorithm, that is the enumeration of patterns, and the computation of the corrected significance threshold. Line 4 starts with the iteration of the thresholds in reversed order, i.e. the first processed threshold is $t = 1.00$. The data set is binarised at this threshold as described in Section 4.1.3.1, and the k -hop parameter is initialised to 0, which means that in the first iteration individual vertices are explored. The supports for each vertex are initialised in Line 7 (we should note that they just correspond to the measurements in the binarised data matrix \mathbf{X}^t). Subsequently, a while-loop is called that will (i) process the current k -hop neighbourhoods (Line 9), and continue to grow the patterns by increasing k (Lines 10/11), unless all possible values of k have been explored, or all patterns have been deemed prunable. Whenever the value of k is increased, the supports have to be recomputed (Line 11). This step is a major computational bottleneck, and it can be optimised by only updating the support of those vertices v that are not yet deemed prunable (stored in the vector $\mathbf{x}_{\text{prune}}$). Furthermore, the step can be trivially parallelised. After all thresholds have been processed, the Tarone

4. Network-guided testing of local neighbourhoods

threshold δ_{Tar} is computed, and p -values of all testable hypotheses are computed. Those that are significant at threshold δ_{Tar} are stored in $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}})$ and, together with the Tarone threshold $\hat{\delta}_{\text{Tar}}$, reported as final output.

Processing of neighbourhoods

The core of the method is the function `PROCESS_NEIGHBOURHOOD` (see Algorithm 4.2). It processes the current neighbourhoods to find the testable ones, adapts the testability threshold $\hat{\delta}$ to guarantee control of the FWER, and determines whether patterns are prunable. It takes the supports of the current patterns, as well as the current value of k , and the vector indicating the k -values at which each pattern was prunable as input. Line 2 invokes an iteration over all vertices, and skips those that are prunable at the current value of k , and the ones that are redundant (Lines 3-5). A pattern is redundant if its support did not change during the last update of k , i.e. $z_v^{k-1,t} = z_v^{k,t}$. This step is included to avoid testing highly redundant patterns multiple times. In the next step, the minimum p -values corresponding to the current pattern are computed (Line 6), and if the pattern is found to be testable, it is added to the set of testable patterns $\mathcal{H}_{\text{test}}$. The increase in the number of testables potentially affects the FWER, and the significance threshold is lowered until the FWER criterion is fulfilled. In Section 3.4.2 we explained that there exist at most $\frac{n+1}{2}$ minimum p -values that can be attained in Pearson's χ^2 test and Fisher's exact test, and that those values can be precomputed, sorted and $\hat{\delta}$ is lowered along those values. For the CMH test, we define a fixed range of those values, as precomputing all attainable minimum p -values is computationally expensive. By lowering the significance threshold, previously testable patterns might become untestable, and those are removed from $\mathcal{H}_{\text{test}}$ in Line 11. After the significance threshold has been adapted, Lines 14-17 test whether a pattern can be pruned from the search space. This depends on the choice of the test, i.e. whether a correction for covariates is included or not. In case of non-existence of covariates, Pearson's χ^2 test or Fisher's exact test is applied. For those, the support $z_v^{k,t}$ of a pattern is a scalar, and the function `IS_PRUNABLE_NO_COV` is invoked, that checks whether the minimum p -value lies in the range of monotonic increase (see Figure 3.3), and whether the pattern is untestable. If both criteria are satisfied, the pattern is prunable, and the function evaluates to *true*. The check for prunability in case of the CMH test is similar, and outlined in the function `IS_PRUNABLE_COV`. In contrast to the other two test, the support $z_v^{t,k}$ is a vector of length r , where r is the number of covariates, and counts the support per table in each covariate class. In Section 3.6.2.1, we saw that the minimum p -value function is not monotonic for the CMH test, but that a monotonic lower bound to the minimum p -value can be found, called the lower envelope $\tilde{\Psi}(\cdot)$. With this lower bound, the check for prunability follows analogously to the one in the non-covariate based tests. After all vertices have been processed, the for-loop initiated in Line 2 finishes, and the updated vector $\mathbf{x}_{\text{prune}}$ is returned.

4.3. The tNeAT-WY method

One major challenge when testing neighbourhoods in a network are the sub- and super-set relationships between the patterns. We have seen in Section 3.5 that dependencies

4. Network-guided testing of local neighbourhoods

Algorithm 4.2 tNeAT functions.

```

1: function PROCESS_NEIGHBOURHOOD( $\{z_v^{k,t}\}_{v \in V}, k, \mathbf{x}_{\text{prune}}$ )
2:   for  $v$  in  $V$  do
3:     if  $\mathbf{x}_{\text{prune}}[i] < k$  or IS_REDUNDANT( $\mathcal{N}_v^{k,t}$ ) then
4:       continue
5:     end if
6:     Compute the minimum  $p$ -value  $\Phi(z_v^{k,t})$  ▷ described in Section 3.3.4.1.
7:     if  $\Phi(z_v^{k,t}) \leq \hat{\delta}$  then ▷ Pattern is testable.
8:        $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \cup \mathcal{N}_v^{k,t}$  ▷ Add to list of testables.
9:       while  $\hat{\delta} \times |\mathcal{H}_{\text{test}}| \geq \alpha$  do
10:        Decrease  $\hat{\delta}$  ▷ Adapt significance threshold.
11:         $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \setminus \{\mathcal{N}_v^{k,t} \mid \Phi(z_v^{k,t}) \geq \hat{\delta}\}$  ▷ Remove untestables.
12:      end while
13:    end if
14:    if IS_PRUNABLE_SPEC( $z_v^{k,t}$ ) then
15:       $\mathbf{x}_{\text{prune}}[i] \leftarrow k$  ▷ Mark super-patterns as prunable.
16:    end if
17:  end for
18:  return  $\mathbf{x}_{\text{prune}}$ 
19: end function
20:
21: function IS_PRUNABLE_NO_COV( $z_v^{k,t}$ ) ▷ Pearson's  $\chi^2$  or Fisher's exact test.
22:  Compute minimum  $p$ -value  $\Phi(z_v^{k,t})$  ▷ described in Section 3.3.4.1.
23:  if  $z_v^{k,t} \geq n - n_1$  and  $\Phi(z_v^{k,t}) > \hat{\delta}$  then
24:    return True
25:  else
26:    return False
27:  end if
28: end function
29:
30: function IS_PRUNABLE_COV( $z_v^{k,t}$ ) ▷ CMH test.
31:  Compute lower envelope  $\tilde{\Phi}(z_v^{k,t})$ , ▷ described in Section 3.6.2.1.
32:  if  $z_v^{k,t} \geq n - n_1$  and  $\tilde{\Phi}(z_v^{k,t}) > \hat{\delta}$  then
33:    return True
34:  else
35:    return False
36:  end if
37: end function

```

4. Network-guided testing of local neighbourhoods

Algorithm 4.3 **tNeAT-WY** (thresholded Neighbourhood Aggregation with Testing and Westfall-Young permutations). Modifications of the **tNeAT** algorithm in Algorithm 4.1 required to incorporate Westfall-Young permutations are highlighted in red.

Input: Data matrix \mathbf{X} , class labels \mathbf{y} , covariate vector \mathbf{c} (optional), network $\mathcal{G} = (V, E)$, target family-wise error rate α , **number of permutations** n_p

Output: corrected significance threshold δ_{Perm} , set of significant neighbourhoods $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}})$

```

1: Initialise global  $\hat{\delta} \leftarrow 1$ , and global  $\mathcal{H}_{\text{test}} \leftarrow \emptyset$ ,
2: Initialise thresholds  $\mathcal{T} \leftarrow \{0.05, 0.1, \dots, 0.95, 1.0\}$ ;
3: Initialise pruning marker  $\mathbf{x}_{\text{prune}} \leftarrow \{\mathbf{d} - \mathbf{1}\}^d$ 
4: for  $i \in \{0, \dots, n_p\}$  do                                      $\triangleright$  initialisation of WY variables (see Section 3.5)
5:   Initialise global  $\mathbf{y}^{(p)} \leftarrow$  random permutation of  $\mathbf{y}$ 
6:   Initialise global  $p_{\min}^{(p)} \leftarrow 1$ 
7: end for
8: for  $t$  in  $\mathcal{T}$ .reversed do
9:    $\mathbf{X}^t \leftarrow$  binarise  $\mathbf{X}$  at threshold  $t$ 
10:   $k \leftarrow 0$ 
11:  Compute supports  $\{z_v^{k,t}\}_{v \in V}$ 
12:  while  $\sum_{i=1}^d \mathbf{1}(\mathbf{x}_{\text{prune}}[i] = 0) > 0$  and not all  $k$  explored do
13:     $\mathbf{x}_{\text{prune}} \leftarrow$  PROCESS_NEIGHBOURHOOD_WY( $\{z_v^{k,t}\}_{v \in V}, k, \mathbf{x}_{\text{prune}}$ )
14:     $k \leftarrow k + 1$ 
15:    Update supports  $\{z_v^{k,t}\}_{v \in V}$ 
16:  end while
17: end for
18:  $\delta_{\text{Perm}} \leftarrow$   $\alpha$ -quantile of  $\left\{ p_{\min}^{(p)} \right\}_{p=1}^{n_p}$ 
19:  $\mathcal{H}_{\text{sig}}(\delta_{\text{Perm}}) \leftarrow \{ \mathcal{N}_v^{k,t} \mid \mathcal{N}_v^{k,t} \in \mathcal{H}_{\text{test}} \text{ and } p_v^{k,t} \leq \delta_{\text{Perm}} \}$ 
20: return  $\delta_{\text{Perm}}, \mathcal{H}_{\text{sig}}(\delta_{\text{Perm}})$ 

```

between patterns can diminish the power of significant pattern mining, and when patterns correspond to neighbourhoods, this dependence might be strong. To address this we turn to Westfall-Young permutations, and integrate the **Westfall-Young light** method presented in Section 3.5 with the neighbourhood detection. We call this permutation-based method **tNeAT-WY** (*thresholded Neighbourhood Aggregation with testing and Westfall-Young permutations*) and describe it in more detail below.

Main body

The main corpus of the **tNeAT-WY** method can be found in Algorithm 4.3. It is based on the **tNeAT** method introduced in the preceding Section (Algorithm 4.1), but instead of estimating the FWER using the set of testable hypotheses, it estimates the FWER from permutations. In order to achieve this, the method **tNeAT** has to be modified in three places: (i) in the initialisation of additional parameters, (ii) in the function **PROCESS_NEIGHBOURHOOD_WY** that estimates the FWER from permutations, and (iii) in the computation of the corrected significance threshold. We highlighted those modified parts in the pseudocode in Algorithm 4.3 and 4.4 in red.

tNeAT-WY requires the same inputs as the **tNeAT** approach plus the number of permutations n_p that should be used to estimate the FWER. This hyperparameter is commonly set to

4. Network-guided testing of local neighbourhoods

$n_p = 1'000$, and its choice is the dominating factor in the computational runtime. As for **tNeAT**, the output is the corrected significance threshold, δ_{Perm} , and the neighbourhoods that are significant at that threshold $\mathcal{H}_{\text{sig}}(\delta_{\text{Perm}})$. Lines 1-4 initialise the same parameters that are used in the **tNeAT** approach, that is the temporary threshold $\widehat{\delta}$, the set of testable patterns $\mathcal{H}_{\text{test}}$, the binarisation thresholds \mathcal{T} and the pruning marker $\mathbf{x}_{\text{prune}}$. In Lines 4-7, the Westfall-Young specific parameters are initialised. This entails the permutations of the label vector \mathbf{y} , as well as the set p_{min} , that will keep track of the smallest observed permutation p -value per permutation (for an in-depth description, see Section 3.5). Lines 8-17 contain the core of the method, i.e. the enumeration of patterns and the adaption of the significance threshold based on the estimation of the FWER rate. This part is identical to the core part of **tNeAT**, with the only exception that the function `PROCESS_NEIGHBOURHOOD_WY` is called, that estimates the FWER from permutations, in Line 13. After all patterns have been processed, the final significance threshold is computed as the α -quantile of the set $\{p_{\text{min}}^{(p)}\}_{p=1}^{n_p}$, and the p -values of all testable patterns in $\mathcal{H}_{\text{test}}$ are tested. Those that are significant, i.e. whose p -value falls below δ_{Perm} are reported as final output, together with the threshold δ_{Perm} .

Processing of edges (Westfall-Young)

The core of the method lies in the function `PROCESS_NEIGHBOURHOOD_WY`. Algorithm 4.4 contains a detailed description of the steps in the method. The method still closely follows the procedure of **tNeAT**, the only difference is the estimation of the FWER. While in **tNeAT** this is done based on the number of testable patterns, in Westfall-Young permutations this is achieved by evaluating the strongest observed association under the null hypothesis, i.e. for the permuted labels. This takes place in Lines 7 to 16 in the pseudocode. If a pattern $\mathcal{N}_v^{k,t}$ is testable, it is added to set of testable hypotheses, and the p -values of the n_p permutations for the pattern are computed by calling the function `COMPUTE_PERMUTATION_PVALUES`. The i^{th} entry of the set $\{p_{\text{min}}^{(p)}\}_{i=1}^{n_p}$ contains the lowest p -values for any processed pattern for permuted label $\mathbf{y}^{(i)}$. Hence, the set can be used to estimate the number of false-positives that occur at the current threshold $\widehat{\delta}$ (Line 10). If this exceeds the target FWER, the significance threshold has to be lowered until control of the FWER can be guaranteed (for details, see Section 3.5), which takes place in Lines 11-14. After the threshold $\widehat{\delta}$ has been adapted, untestable patterns are removed from the set $\mathcal{H}_{\text{test}}$ (Line 17), and prunability of patterns is assessed analogously to **tNeAT**.

4.4. Implementation details

The runtime of both methods, **tNeAT** and **tNeAT-WY**, scales with the number of nodes in the network d , the diameter of the network σ , and the number of thresholds $|\mathcal{T}|$. Since we fix the number of thresholds \mathcal{T} to 21, we treat this a constant, such that the runtime becomes $O(d\sigma)$, which is linear in the number of nodes. We implemented both methods in C++. One of the bottlenecks is the update of the supports of the neighbourhoods (Line 11 in Algorithm 4.1 and Line 15 in Algorithm 4.3). We use OpenMP to parallelise this step. In order to speed-up the execution times of the costly Westfall-Young permutations, we

4. Network-guided testing of local neighbourhoods

Algorithm 4.4 tNeAT-WY functions. Modifications of tNeAT specific functions (Algorithm 4.2) required to incorporate Westfall-Young permutations are highlighted in red.

```

1: function PROCESS_NEIGHBOURHOOD_WY( $\{z_v^{k,t}\}_{v \in V}, k, \mathbf{x}_{\text{prune}}$ )
2:   for  $v$  in  $V$  do
3:     if  $\mathbf{x}_{\text{prune}}[i] < k$  or IS_REDUNDANT( $\mathcal{N}_v^{k,t}$ ) then
4:       continue
5:     end if
6:     Compute the minimum  $p$ -value  $\Phi(z_v^{k,t})$  ▷ described in Section 3.3.4.1.
7:     if  $\Phi(z_v^{k,t}) \leq \hat{\delta}$  then ▷ Pattern is testable.
8:        $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \cup \mathcal{N}_v^{k,t}$  ▷ Add to list of testables.
9:       COMPUTE_PERMUTATION_PVALUES( $z_v^{k,t}$ )
10:       $\text{FWER}(\hat{\delta}) = \frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \hat{\delta})$ 
11:      while  $\text{FWER}(\hat{\delta}) \geq \alpha$  do
12:        Decrease  $\hat{\delta}$ 
13:         $\text{FWER}(\hat{\delta}) = \frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \hat{\delta})$ 
14:      end while
15:       $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \setminus \{\mathcal{N}_v^{k,t} \mid \Phi(z_v^{k,t}) \geq \hat{\delta}\}$  ▷ Remove untestables.
16:    end if
17:    if IS_PRUNABLE_SPEC( $z_v^{k,t}$ ) then ▷ See Algorithm 4.2
18:       $\mathbf{x}_{\text{prune}}[i] \leftarrow k$  ▷ Mark super-patterns as prunable.
19:    end if
20:  end for
21:  return  $\mathbf{x}_{\text{prune}}$ 
22: end function
23:
24: function COMPUTE_PERMUTATION_PVALUES( $z_v^{k,t}$ )
25:   for  $p \in \{1, \dots, n_p\}$  do
26:     Compute  $p$ -value of  $p^{\text{th}}$  permutation  $p_v^{k,t,(p)}$ 
27:      $p_{\min}^{(p)} \leftarrow \min(p_{\min}^{(p)}, p_v^{k,t,(p)})$ 
28:   end for
29: end function

```

furthermore parallelise the computation of the permutation p -values (function COMPUTE_PERMUTATION_PVALUES in Lines 25-28, Algorithm 4.4).

4.5. Simulation study

We first evaluate the performance of our methods tNeAT and tNeAT-WY on a range of artificially generated data sets. In this supervised simulation study, where the ground truth is known, we are able to assess the capacities of our methods with respect to type-I and type-II errors. We furthermore compare our methods against a variety of methods designed to find meaningful interactions in networks.

4. Network-guided testing of local neighbourhoods

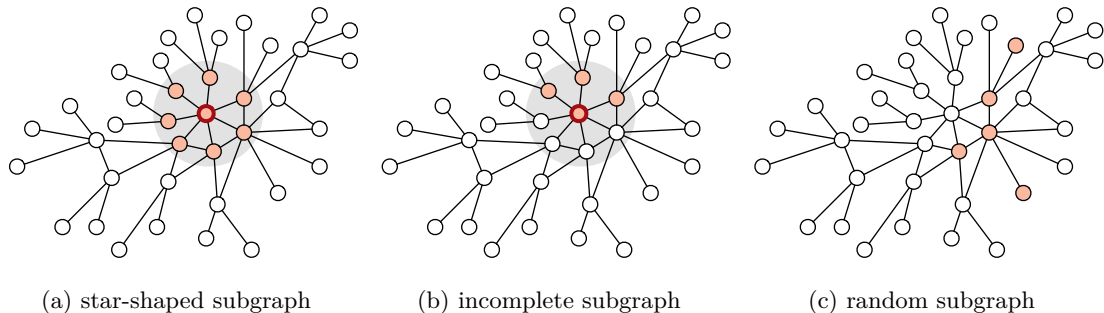


FIGURE 4.4.: Example of the three different configurations of induced subgraphs in the simulation study. (a) star-shaped subgraph, where genes in the truly-associated subgraph correspond to a complete neighbourhood in the network, (b) incomplete subgraph with $f = 0.5$, i.e. the genes in the truly-associated subgraph correspond to 50.0% of nodes of a neighbourhood, and (c) a random subgraph, where nodes in the truly-associated subgraph form any connected subgraph in the network. In (a) and (b), the anchor of the neighbourhood is highlighted in red, and the complete neighbourhood is shadowed in grey.

4.5.1. Experimental setup

4.5.1.1. Data set generation

We generate artificial data using the connectivity structure of a real KEGG pathway: (*hsa05205*) *Proteoglycans in cancer* [189, 190]. The pathway contains 224 nodes and 1'324 edges. We randomly generate data for 500 samples divided into two phenotypic groups, and assign each sample to one of two covariate classes. Each gene is represented with a fixed number of n_g SNPs that is drawn at random from the range $[1, 20]$, and for each sample we draw the number of observed minor alleles for that gene uniformly from the range $[0, n_g]$. We normalise the count of minor alleles with the length of the genes. This produces a data matrix $\mathbf{X}^{500 \times 224}$, where $0 \leq \mathbf{X}_{i,j} \leq 1$ with $i = 0, \dots, 499$ and $j = 0, \dots, 223$, a binary label vector \mathbf{y} and a binary covariate vector \mathbf{c} . We repeat this data generation process 50 times, and in each of those 50 random folds generate one *truly-significant* subgraph of size five, and a second subgraph of size five that is only associated to the covariate which in turn is associated to the phenotype. We call the latter a *confounded* subgraph. Both subgraphs are induced with strengths of association to the phenotype p_s and p_{con} , for the significant and confounded subgraph, respectively. We vary both p_s and p_{con} between 0 and 1, with 1 indicating a very strong association to the phenotype. A detailed description of the data generation process can be found in Appendix A.

4.5.1.2. Subgraph configurations

Our proposed methods tNeAT and tNeAT-WY test the hypothesis whether a complete k -hop neighbourhood is associated with the phenotype. In order to evaluate the performance of tNeAT/tNeAT-WY given that the ground truth deviates from this hypothesis, we simulate data for different types of truly-associated and confounded subgraphs. We consider three types of subgraph configurations: (i) *star-shaped*, (ii) *incomplete*, and (iii) *random*. In the *star-shaped* configuration, the truly-significant subgraph fits the hypothesis, i.e. it

4. Network-guided testing of local neighbourhoods

corresponds to all nodes in a 1-hop neighbourhood of an anchor node in the network (Figure 4.4a). In the *incomplete* configuration, the subgraph corresponds to a fraction f of the neighbourhood. For example if $f = 0.5$, half of the nodes in the neighbourhood correspond to the significantly associated subgraph, and half of the nodes are randomly simulated (Figure 4.4b). In the *random* configuration, we simulate a subgraph as a connected component without any constraints concerning the topology of nodes in the network (Figure 4.4c).

4.5.1.3. Comparison partners

We compare our methods **tNeAT** and **tNeAT-WY** against a variety of baselines and well-established network models. The baselines can be categorised into three different classes of methods: (i) Tarone-based methods, (ii) methods based on logistic regression followed by a likelihood ratio test, and (iii) network propagation.

Tarone-based methods

The comparison partners in this category use the same Tarone procedure to account for multiple testing as in **tNeAT**, but test other types of patterns. To be precise, we implemented two different methods:

- i. **tUniT** (thresholded Univariate Testing), that only tests individual genes, i.e. vertices in the network, at all binarisation thresholds.
- ii. **tEdgeT** (thresholded Edge Testing), that tests all pairs of genes that are connected by an edge in the network. It uses the same pattern indicator function as **tNeAT**, with the only difference that the pattern is an edge between two genes.

The purpose of those baseline comparison partners is to establish whether the neighbourhoods contain meaningful information that cannot be obtained from the individual genes or edges alone. Since they are based on Tarone’s procedure to correct for multiple testing, they benefit from the same increase in power as our proposed methods do.

Logistic regression-based method

Comparison partners in this category are based on a logistic regression followed by a likelihood ratio test to obtain a p -value of association. The patterns analysed with those methods are either the individual genes, or the complete 1-hop neighbourhoods. We furthermore create two different representations of the patterns: one counts the minor alleles in a pattern, creating an n -dimensional count vector for each pattern. Counting the number of minor alleles is a strategy that is commonly used in burden tests [119]. The second representation is based on the $n \times d_p$ -dimensional data matrix of the d_p genes in the current pattern. The representation of a pattern is given as input to a logistic regression model, together with the same binary covariate vector as is used by **tNeAT/tNeAT-WY**. These approaches for the representation of patterns lead to the following comparison partners:

- i. **burden_k0** that tests each individual gene using the counts of minor alleles in the gene to represent a pattern in the logistic regression.

4. Network-guided testing of local neighbourhoods

- ii. `burden_k1` that tests 1-hop neighbourhoods using the counts of minor alleles in the gene to represent a pattern in the logistic regression.
- iii. `regress_k1` that tests each individual gene for association, using the fractions of minor alleles per gene in a pattern in the logistic regression.
- iv. `regress_seq` sequential regression that is similar to `regress_k1`, but compares against a different null-model in the likelihood ratio test: instead of comparing the 1-hop neighbourhood against the null-model using no genetic information, it compares against the null-model that includes the anchor gene.

To correct for multiple hypothesis testing we apply a standard Bonferroni correction in those cases. Note that there exists no `regress_k0` baseline, as those results are identical to the ones obtained with `burden_k0`. The purpose of the logistic regression based baselines is to evaluate the effect of the encoding of a pattern, or in other words, to evaluate whether information is lost by binarising a pattern.

Network methods

This class of comparison partners contains two methods:

- i. `Hierarchical HotNet` [146], an approach that is based on a network propagation, and requires the computation of a summary statistic for each node in the network. Those so-called ‘scores’ are superimposed on nodes in the network and subsequently propagated through the network, until a steady state distribution is reached. This distribution serves as similarity measure between nodes, and as input for a hierarchical clustering. Hierarchical HotNet returns a so-called ‘hot subnetwork’ together with an empirical p -value indicating whether the size of the observed largest ‘hot subnetwork’ is likely to occur by random chance. In our application, we use the χ^2 statistic obtained from the likelihood-ratio test conducted with the `burden_k0` method as initial node scores in the network.
- ii. `dmGWAS` [63], an approach that was designed for a network guided exploration of GWAS summary statistics. Similarly to hierarchical HotNet, marginal statistics of the genes, commonly p -values, are superimposed on nodes in the network, and subgraphs in the network are identified as connected components in the network that are enriched in low p -value genes. Those components are identified with a greedy search strategy. In our application, we use the p -values from the `burden_k0` to represent genes in the network, and choose the hyperparameters of the method according to the recommendations of the authors.

4.5.2. Application of tNeAT to simulated data

We analyse the synthetically generated data sets with our proposed methods `tNeAT` and `tNeAT-WY` using the CMH test, to account for categorical covariates. The performance is evaluated in the form of the type-I and the type-II error. Since we generated 50 simulated data sets for each association strength p_s and each subgraph configuration, the reported results correspond to averages across 50 simulation runs for each combination of p_s and configuration.

4. Network-guided testing of local neighbourhoods

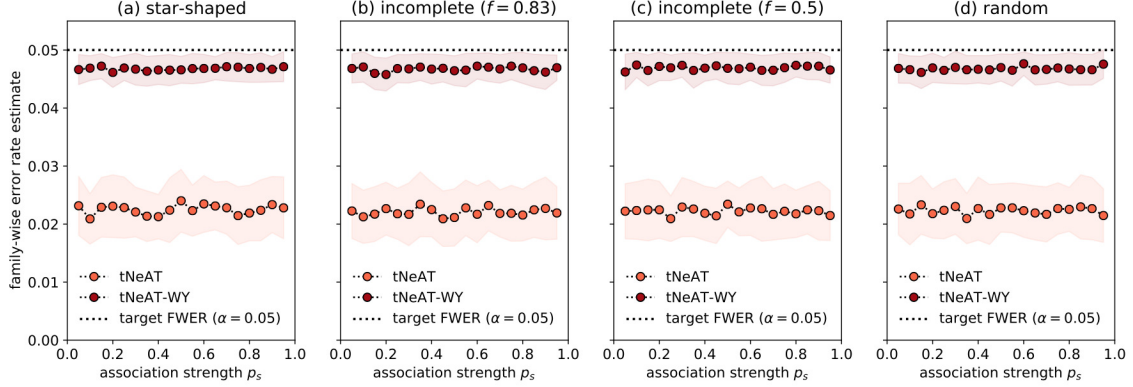


FIGURE 4.5.: Family-wise error rate estimates of \mathbf{tNeAT} / $\mathbf{tNeAT-WY}$ for varying association strengths p_s . Each figure corresponds to the estimate for a specific subgraph-configuration. The shaded areas around the lines indicate the standard deviation of FWER estimates obtained for the 50 simulations per p_s -values.

4.5.2.1. Type-I error analysis: estimate of family-wise error rate

Each run of \mathbf{tNeAT} and $\mathbf{tNeAT-WY}$ resulted in a significance threshold δ_{Tar} and δ_{Perm} , respectively. To evaluate the type-I error, that is the occurrence of false-positive associations, we estimate the family-wise error rate from each simulation using 1'000 random permutations of the phenotype vector. For those 1'000 permutations, we test all patterns at the respective significance threshold (δ_{Tar} , δ_{Perm}), and since the permutations destroyed any true signal in the data, each pattern that is deemed significant can be considered a false-positive association. The FWER is then estimated as the fraction of permutations for which at least one significant pattern was detected. We would like to note that, since $\mathbf{tNeAT-WY}$ is based on this permutation scheme, we use the FWER estimated from the execution of $\mathbf{tNeAT-WY}$ directly. The results of this analysis are illustrated in Figure 4.5 for all four types of subgraph configurations. All methods were invoked with a target FWER of $\alpha = 0.05$, and we observe that both methods successfully control the FWER at this target level (indicated with black horizontal line in figures). We furthermore observe that $\mathbf{tNeAT-WY}$ which takes dependencies between patterns into account results in a substantially higher power than its purely Tarone-based counterpart \mathbf{tNeAT} .

4.5.2.2. Type-II error analysis: estimation of power

The type-II error is the error incurred by missing a true association, that is a so-called false-negative. We estimate this error using power, i.e. the capability of a method to detect the true positive associations. We are able to evaluate this, as in the simulation study the underlying ground truth is known. To evaluate the performance of our proposed methods \mathbf{tNeAT} and $\mathbf{tNeAT-WY}$ with respect to power, we define power as the probability to completely recover the initially induced subgraph, and count it as a success if the truly-significant subgraph can be fully recovered with a method for a simulation. We then report the fraction of success across the 50 simulations as the final power of the method. The same procedure is applied for all comparison partners that rely on the discovery subgraphs

4. Network-guided testing of local neighbourhoods

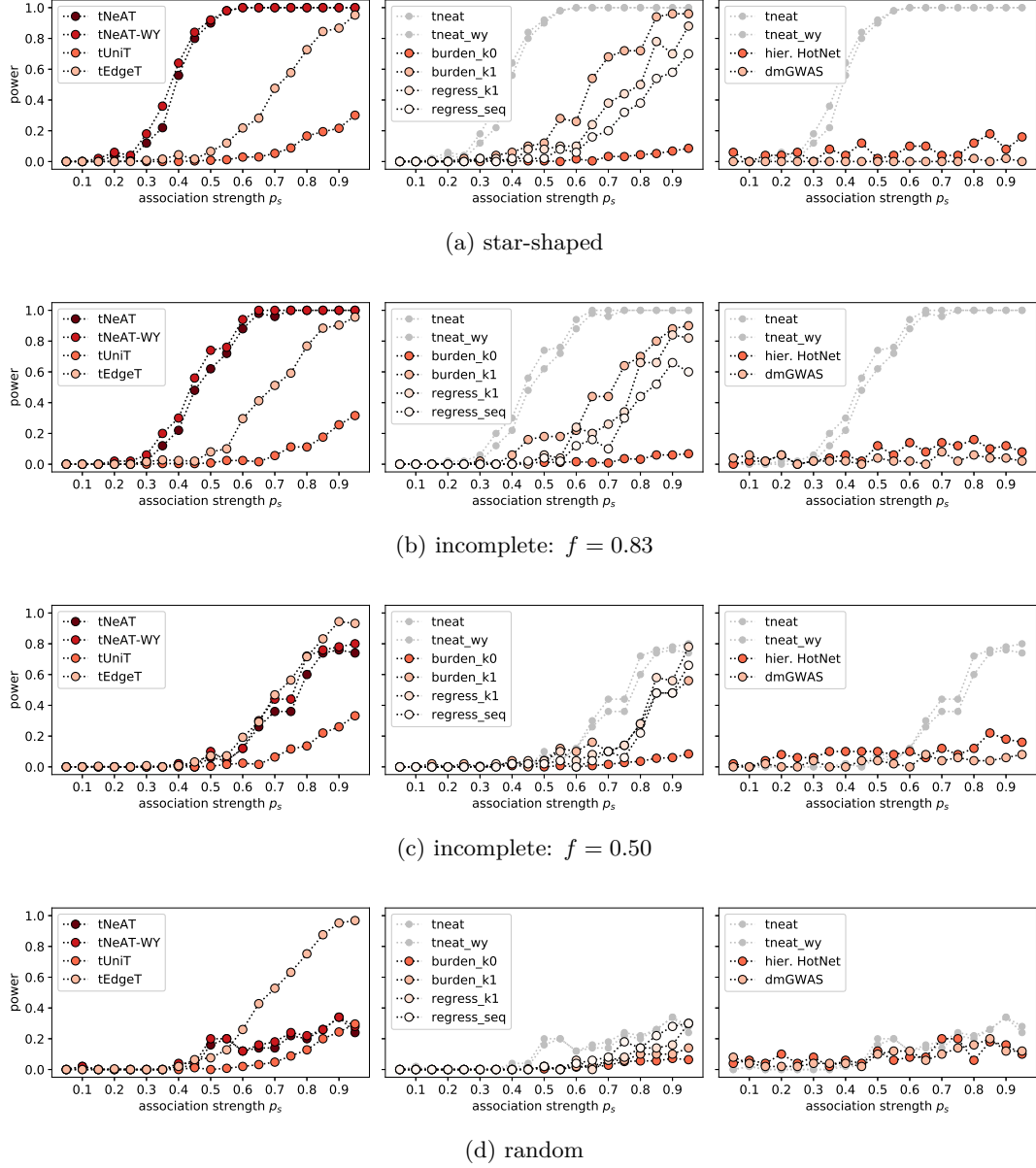


FIGURE 4.6.: Power analysis of **tNeAT**, **tNeAT-WY** and comparison partners on simulated data for different subgraph configurations: (a) star-shaped, (b) incomplete ($f = 0.83$), (c) incomplete ($f = 0.5$) and (d) random. The first column of each subplot contains the methods using Tarone, the second column contains methods based on logistic regression, and the third column contains network-based methods. In the plots of the second and third column, the power of **tNeAT** and **tNeAT-WY** is indicated in grey, to allow for an easier comparison.

4. Network-guided testing of local neighbourhoods

within a network. However, for methods that only test individual genes or edges this ‘success’ cannot be achieved due to the type of tested patterns. In those cases, we report the fraction of the truly-significant subgraph that could be recovered when combining all significant hits found with the respective method.

The results of this analysis are illustrated in Figure 4.6. Each row in this plot illustrates the power of methods falling into the three different categories of comparison partners, with Tarone-based methods on the left, regression-based methods in the centre, and network-based methods on the right. We observe that **tNeAT** and **tNeAT-WY** outperform all competitors for the star-shaped subgraph configuration, i.e. the configuration that matches the patterns tested in those approaches (see Figure 4.6a). This remains true, if the subgraph configuration only diverges little from the tested patterns, as is the case for the incomplete setting with $f = 0.83$ (i.e. the five truly-associated genes lie within a 1-hop neighbourhood of size 6, see Figure 4.6b). As this deviation gets larger, the stronger the decrease in power for **tNeAT** and **tNeAT-WY**. For the incomplete setting with $f = 0.50$, i.e. the truly-significant subgraph makes up only 50.0% of a neighbourhood, **tEdgeT** starts to perform very similar to **tNeAT** and **tNeAT-WY**, as do the regression-based methods that analyse 1-hop neighbourhoods (see Figure 4.6c). For the random subgraphs we observe a poor performance of all methods (see Figure 4.6d), except for the **tEdgeT** approach.

In general we observe that univariate analysis, i.e. those that only test individual genes for association, are underpowered to recover the truly-associated subgraph. The non-linear nature of interaction between genes results in genes that only exhibit a significant association upon combination, while the individual genes themselves do not necessarily show strong marginal associations to the phenotype. While the edge-based approach **tEdgeT** is not able to outperform **tNeAT** and **tNeAT-WY**, with exception of the random configuration of subgraphs, we observe that its performance stays similar across all different configurations of the true subgraph. Since **tEdgeT** is taking pairwise interactions into account, it is recovering some parts of the true underlying subgraphs, and the edge-based exploration allows for more flexibility to recover patterns that are not full neighbourhoods. As the data was simulated according to a model of genetic heterogeneity, we observe that methods that do not account for this non-linear mode of interaction, such as the regression models and the network-based models, have difficulties in recovering the truly-associated subgraph. This is especially the case for network-based methods, that rely on strong marginal signals of the genes in the network.

4.6. Application to 20 *Arabidopsis thaliana* phenotypes

The simulation study indicated that **tNeAT** and **tNeAT-WY** clearly outperform comparison partners if the true signal overlaps with genes that form neighbourhoods in a network. While a deviation from this configuration might lead to a loss in power, we still consider it worthwhile to analyse if there exist neighbourhoods in networks that are jointly associated to a binary trait under a model of genetic heterogeneity. We hence apply our methods to analyse a range of genetic data sets of the model organism *Arabidopsis thaliana* (short *A. thaliana*).

4. Network-guided testing of local neighbourhoods

TABLE 4.2.: Description of *A. thaliana* phenotypes. The column ‘class n_1 ’ contains the number of samples in the minority class, the column ‘baseline λ_{gc} ’ corresponds to the genomic inflation obtained with **tNeAT-WY** if no covariates are used. The column ‘best λ_{gc} ’ indicates the genomic inflation after using a categorical covariate to correct for population structure. The number of covariates used to achieve this is listed in the column r .

	samples	class n_1	baseline λ_{gc}	r	best λ_{gc}
<i>Anthocyanin10</i>	177	33	1.55	5	1.17
<i>Anthocyanin16</i>	176	70	1.21	4	1.00
<i>Anthocyanin22</i>	177	64	1.29	8	0.99
<i>Chlorosis10</i>	177	28	0.95	3	0.99
<i>Chlorosis16</i>	176	84	1.00	1	1.00
<i>Chlorosis22</i>	176	66	1.48	7	1.01
<i>Emco5</i>	86	17	1.16	5	1.12
<i>Emoy</i>	76	35	1.21	9	0.97
<i>Emwa1</i>	85	32	1.66	5	1.18
<i>Hiks1</i>	84	33	1.33	5	1.26
<i>LES</i>	95	21	2.21	9	1.18
<i>LY</i>	95	29	2.33	10	1.17
<i>Leafroll10</i>	177	78	1.55	5	0.99
<i>Leafroll16</i>	176	37	1.35	9	1.02
<i>Leafroll22</i>	176	31	1.38	10	1.32
<i>Noco2</i>	87	39	1.38	5	0.97
<i>avrB</i>	87	32	1.74	10	1.06
<i>avrPphB</i>	90	44	1.65	6	1.07
<i>avrRpm1</i>	84	28	1.69	2	1.07
<i>avrRpt2</i>	89	17	1.21	5	1.05

4.6.1. Experimental setup

We apply our methods to widely used *A. thaliana* data sets from Atwell *et al.* [191]. We downloaded those data sets from easyGWAS [192] and AraPheno [193, 194]. The data set contains genetic data of 1’307 *A. thaliana* accessions, and a total of 214’051 genotyped SNPs. Furthermore, there exists a collection of 107 phenotypes covering various traits of the plants. Among those, 21 phenotypes are dichotomous and hence qualify for an analysis with our proposed approach. Their respective sample sizes range between 84 and 177. After an initial analysis, we excluded the *YEL* phenotype due to its large class imbalance. We provide an overview of the phenotypes, the total samples sizes and the number of samples in the minority class in Table 4.2.

To represent genetic interactions, we use the *A. thaliana* protein-protein interaction network provided by ‘The Arabidopsis Information Resource’ (TAIR) [195, 196]. It is a protein-protein interaction network curated from literature by TAIR and BIOGRID. Originally, the network contains 1’627 interactions between 1’325 *A. thaliana* proteins. In order to map the genetic data in the form of SNPs to the network, we proceed as follows: In the first step, we map the SNPs in the Atwell data set to genes in the network. A SNP is mapped to a gene based on physical proximity, i.e. if it overlaps with introns or exons of a

4. Network-guided testing of local neighbourhoods

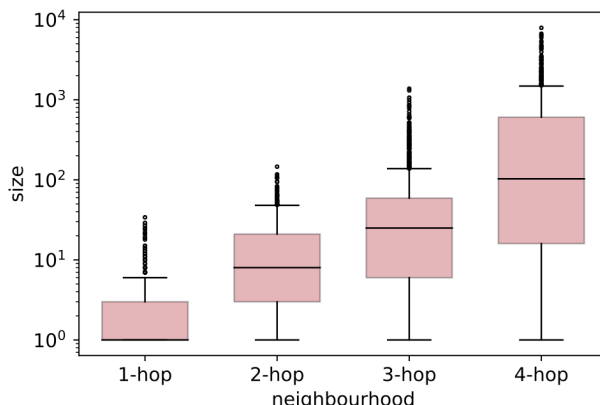


FIGURE 4.7.: Distributions of k -hop neighbourhood sizes in the TAIR (*A. thaliana*) network. The average degree (1-hop neighbourhood size) of the network is 2.39. For the 2-hop, 3-hop and 4-hop neighbourhoods, the average size increases to 13.96, 62.03 and 457.74 nodes, respectively.

gene (see Figure 2.4a). For this mapping we used the Araport11 genome annotation [197, 198]. With this, a gene can be represented by a set of SNPs, and we refer to the number of SNPs mapped to a gene as the **gene size**. In the next step, we created a representation of the gene based on the minor allele counts of SNPs within a gene for each sample. That is, if a gene is represented by n_g SNPs, it can contain between 0 and $2n_g$ minor alleles. Depending on the method, we either rescaled the gene-count to the range of $[0, 1]$ by dividing the minor allele counts by $2n_g$, or in case of the burden tests, worked on the count data.

We removed all genes and their adjacent edges from the network, if the gene in the network did not contain any of the 214'051 SNPs. This was the case for 88 genes. In total, we were able to represent 1'237 genes using the SNPs in the Atwell data set, resulting in a network with 1'237 vertices, and 1'435 edges between them. Figure 4.7 illustrates the sizes of the 1-hop to 4-hop neighbourhoods in the network, with average sizes of 2.39, 13.96, 62.03 and 457.74, respectively.

To account for population structure or cryptic relatedness between samples, we generated categorical covariates. For this purpose, we applied a k -means clustering to the three leading principal components of the empirical kinship matrix [following 164]. With this procedure, the value of k in k -means clustering determines the number of covariate classes, and eventually the number of contingency tables created for each CMH test. Since the best number of covariate classes is unknown apriori, we chose it by running the Tarone based methods (**tNeAT**, **tNeAT-WY**, **tUniT**, **tEdgeT**) for values of k in the range of 1 to 10, and report the results for the value of k that led to the best genomic inflation factor λ_{gc} .

4.6.2. Results

Since we observed a slightly improved performance of **tNeAT-WY** compared to **tNeAT** in the simulation study, we focus on the results obtained with the Westfall-Young permutation scheme. We focus the evaluation of the results on three different aspects: (i) the reduction

4. Network-guided testing of local neighbourhoods

TABLE 4.3.: Number of significant hits found with each method for the *A. thaliana* data sets. Each entry indicates the number of significant patterns detected for a phenotype, and the number in parentheses corresponds to the mean size of the patterns.

	tNeAT-WY	tUniT	tEdgeT	burden_k0	burden_k1	regress_k1
<i>Anthocyanin10</i>	-	2	-	1 (1.0)	-	1 (2.0)
<i>Anthocyanin16</i>	1 (2.0)	-	-	1 (1.0)	-	-
<i>Chlorosis10</i>	-	5	-	-	-	-
<i>Chlorosis22</i>	1 (1.0)	1	-	1 (1.0)	1 (2.0)	2 (5.5)
<i>Emco5</i>	-	-	-	-	-	4 (28.8)
<i>Emoy</i>	-	-	-	-	-	2 (32.5)
<i>Emwa1</i>	-	-	-	-	-	1 (35.0)
<i>Hiks1</i>	1 (3.0)	-	1	1 (1.0)	2 (6.0)	7 (11.7)
<i>LES</i>	-	-	-	1 (1.0)	-	7 (18.0)
<i>LY</i>	-	-	-	1 (1.0)	-	4 (22.8)
<i>Leafroll16</i>	-	-	1	-	-	-
<i>Noco2</i>	-	-	-	-	-	2 (19.5)
<i>avrB</i>	1 (1.0)	1	2	1 (1.0)	1 (2.0)	7 (10.7)
<i>avrPphB</i>	2 (1.5)	1	-	1 (1.0)	2 (2.0)	4 (17.3)
<i>avrRpm1</i>	1 (1.0)	1	3	2 (1.0)	1 (2.0)	7 (10.7)
<i>avrRpt2</i>	6 (2.0)	1	7	2 (1.0)	6 (2.2)	19 (8.7)

of the genomic inflation λ_{gc} , (ii) the number of significant hits detected with tNeAT-WY and the comparison partners, and (iii) the qualitative analysis of the significant hits found with tNeAT-WY.

The genomic inflation factor λ_{gc} is a well-established metric to determine the degree of in- or deflation of test statistics. Values close to 1.0 indicate absence of inflation, while values significantly larger or smaller indicate inflation and deflation of the test statistics, respectively. Deviations of more than 0.1 suggest a correction for hidden structure. This is mostly achieved by including covariates into the model. We initially applied tNeAT-WY to all 20 data sets without using covariates. In 18 out of 20 cases, test statistics were inflated by more than 0.1, in one case we observed a slight deflation (*Chlorosis10*), in one case the inflation factor was equal to 1.0 (*Chlorosis16*). As described in greater detail in the previous section, we included between 2 and 10 covariates derived from the empirical kinship matrix between samples into the analysis to correct for structure. We chose the best number of covariates as the one that resulted in the λ_{gc} -value closest to 1.0. The number of covariates and the resulting λ_{gc} values are listed in Table 4.2. We observe a consistent improvement of the genomic inflation and deflation across all 19 phenotypes upon including covariates. While the target range of [0.9, 1.1] can be achieved for 13 out of 19 phenotypes by including covariates, for six phenotypes the inflation still exceeds 1.1. However, similar dynamics were observed in the original publication of the data [see supplementary material in 191]. In summary, including covariates leads to the desired reduction of genomic inflation.

We compared our approach tNeAT-WY against the most promising comparison partners

4. Network-guided testing of local neighbourhoods

from the simulation study, that is the two significant pattern mining approaches based on Tarone’s procedure `tUnit` and `tEdgeT`, the burden tests `burden_k0` and `burden_k1`, and the regression test `regression_k1` (see Section 4.5.1.3 for details). In order to allow for a fair comparison, we give all comparison partners access to covariates to correct for population structure. In the case of `tUnit` and `tEdgeT` the procedure to find the best number of covariates is identical to the one applied in `tNeAT-WY`. For the regression and burden tests we incorporate the three leading principal components of the empirical kinship matrix as covariates into the test to correct for inflation. Those are the same principal components that are used to derive covariates for the `tNeAT-WY`, `tUnit`, and `tEdgeT` approaches. We run all methods on the 20 *A. thaliana* phenotypes. Table 4.3 reports the number of significant hits for each of the methods. In 16 out of 20 phenotypes, at least one method found significant hits. `tNeAT-WY` discovers significant patterns for 7 phenotypes, with patterns comprising between 1 and 3 genes. `tUnit` and `tEdgeT` discover significant patterns in 7 and 5 phenotypes, respectively. The number of significant hits discovered with burden and regression based tests exceeds those based on significant pattern mining approaches. Especially the `regress_k1` method, that tests the additive effect of all genes in 1-hop neighbourhoods discovers significant hits in 13 out of the 20 phenotypes, and we observe that the sizes of the neighbourhoods clearly exceed those of neighbourhoods detected with `tNeAT-WY`.

There exist two main hypotheses that attribute to the superiority of regression based approaches compared to the `tNeAT/tNeAT-WY` in this setting. The first main difference between the regression based approaches and the `tNeAT-WY` method is the representation of genes, and the thereby implicated mode of interaction between genes in neighbourhoods. While the regression based approach models a linear additive effect of interactions between genetic entities, `tNeAT` and `tNeAT-WY` test for genetic heterogeneity between genetic variants at various thresholds. Given this modelling assumption, `tNeAT` and `tNeAT-WY` are more closely related to the burden test, and we observe that the landscape of results obtained with `tNeAT-WY` resembles the one of the burden test. In summary, one potential explanation for the observed differences is this underlying model assumption.

Furthermore, a potential factor explaining the numerous significant results obtained with the regression based method is the reduced burden of statistical hypothesis testing due to the different hypothesis spaces. While `tNeAT-WY` accounts for all possible k -hop neighbourhoods at various threshold values, the number of tests conducted with the regression based test corresponds to the number genes in the network. Even when using the conservative Bonferroni correction, the number of tests by far falls below the number of hypotheses in the space explored by `tNeAT-WY`.

Last, we analyse those neighbourhoods that were deemed significant with the `tNeAT-WY` approach. A detailed summary of the significant hits can be found in Table 4.4. `tNeAT-WY` detected a total of 13 neighbourhoods across seven *A. thaliana* phenotypes. Five out of those thirteen neighbourhoods correspond to ‘0-hop’ neighbourhoods, i.e. genes in the network. The remaining significant hits are 1-hop neighbourhoods containing two or three genes. In the case of the *avrPphB* phenotype, the anchor gene of the significant 1-hop neighbourhood is found to be significant itself. A similar behaviour can be observed for the

4. Network-guided testing of local neighbourhoods

TABLE 4.4.: Analysis of significant **tNeAT-WY** hits in *A. thaliana*. Each row corresponds to one significant hit found with **tNeAT-WY**. The ‘gene name’ column indicates the name of the anchor gene, the ‘k-hop’ value the depth of th neighbourhood, and the ‘size’ column the size of the k-hop neighbourhood around the anchor gene. The threshold indicates the lowest threshold at which the pattern was found significant, with the corresponding *p*-value in the second column.

	<i>p</i> -value	gene name	k-hop	size	threshold	tUnit	tEdgeT	burden_k0	burden_k1	regress_k1
<i>Anthocyanin16</i>	8.77×10^{-7}	AT4G36800	1	2	0.05	0	0	0	0	0
<i>Chlorosis22</i>	2.43×10^{-7}	AT3G26730	0	1	0.20	1	0	1	1	1
<i>Hiks1</i>	8.40×10^{-7}	AT5G56280	1	3	0.20	0	1	1	1	1
<i>avrB</i>	1.05×10^{-7}	AT3G07040	0	1	0.25	1	1	1	1	1
<i>avrPphB</i>	2.61×10^{-9}	AT1G12220	0	1	0.05	1	0	1	1	1
	7.88×10^{-6}	AT1G12220	1	2	0.05	1	0	1	1	1
<i>avrRpm1</i>	6.31×10^{-7}	AT3G07040	0	1	0.25	1	1	1	1	1
<i>avrRpt2</i>	9.83×10^{-10}	AT4G08850	1	2	0.35	1	1	1	1	1
	1.44×10^{-9}	AT4G26090	0	1	0.30	1	1	1	1	1
	6.13×10^{-9}	AT3G45780	1	2	0.40	1	1	1	1	1
	2.11×10^{-8}	AT2G45960	1	3	0.40	1	1	1	1	1
	5.85×10^{-8}	AT3G01290	1	2	0.30	1	1	1	1	1
	2.99×10^{-7}	AT3G14210	1	2	0.40	1	1	1	1	1

avrRpt2 phenotype, for which six significant hits are detected. All 1-hop neighbourhoods that are found to be significant also contain the gene AT4G26090, that is itself significantly associated to the phenotype. For three other phenotypes, that is *Chlorosis22*, *avrB* and *avrRpm1* the only significant pattern are genes, but not *k*-hop neighbourhoods. Only in the case of *Hiks1* and *Anthocyanin16* we were able to discover significant 1-hop neighbourhoods that did not contain single genes that were identified as significant with **tNeAT-WY**. In summary, all significant neighbourhoods discovered with the **tNeAT-WY** approach were either driven by individual genes, or contained only up to three genes.

We next compared those 13 significant neighbourhoods against the significant hits from the comparison partners. To this end, we report whether a pattern deemed significant with **tNeAT-WY** overlaps with any pattern deemed significant by one of the comparison methods. Those results can be found in the last 5 columns in Table 4.4. We observe that in 12 out of the 13 **tNeAT-WY** patterns, at least one gene could also have been detected with other methods. Especially the regression based methods **burden_k0**, **burden_k1**, and **regress_k1** show a consistent overlap across all candidate patterns, with an exception for the 1-hop neighbourhood detected in the *Anthocyanin16* phenotype. When evaluating those results, one should take into account that the regression-based comparison partners in this study are analysing smaller hypothesis spaces. As mentioned above, this attributes those methods with higher power due the smaller number of hypotheses and a reduced multiple hypothesis testing burden compared to the Tarone-based methods. However, even

4. Network-guided testing of local neighbourhoods

if the regression-based methods are not considered in this comparison, the Tarone-based methods `tUnit` and `tEdgeT` are still able to partially recover the same 12 significant patterns detected with `tNeAT-WY`. This further corroborates the conclusion of the previous paragraph, that is that genetic heterogeneity can be best discovered at the level of individual genes or interactions between gene pairs, i.e. edges in the network.

4.7. Summary and discussion

In this section, we introduced two novel methods, called `tNeAT` and `tNeAT-WY`, to find sets of genes that are statistically significantly associated to a binary phenotype of interest under a model of genetic heterogeneity. Testing sets of genes jointly for association instead of testing individual genes allows us to identify sets of genes whose aggregated risk in the form of minor alleles is different between two phenotypic groups. Since it is computationally and statistically infeasible to enumerate all possible sets of genes, we included biological prior-knowledge in the form of molecular networks into the analysis. In those networks, genes correspond to nodes, and edges indicate the presence of physical interactions between genes. Instead of exploring all possible gene sets, we generated sets of genes as local k -hop neighbourhoods in the network, thereby (i) reducing the search space of all gene sets, and at the same time the burden of multiple hypothesis testing, as well as (ii) creating interpretable sets of genes for downstream analyses. Furthermore, these approaches exhibit runtimes that are linear in the number of nodes in the network, resulting in desirable linear runtimes.

To address the inherent problem of multiple hypothesis testing caused by the simultaneous testing of large numbers of hypotheses, we formulate the task of testing k -hop neighbourhoods in a network as a significant pattern mining problem. This allows us to leverage concepts and principles from this area of research, namely the Tarone procedure, that enable large-scale statistical testing in scenarios in which a classical Bonferroni correction restricts power to find associations. Our methods build on established methods in the field of significant pattern mining, and extend those existing approaches in two ways. Firstly, instead of testing arbitrary sets of features, k -hop neighbourhoods are explored. This results in interpretable sets of features and alleviates computational bottlenecks compared to classical itemset mining approaches. Secondly, `tNeAT` and its Westfall-Young permutation counterpart `tNeAT-WY` use non-binary input data, and apply a thresholding scheme to take full advantage of concepts that boost both, the processing and pruning of a feature set, as well as statistical power. It is important to highlight that, without taking advantage of testability and prunability of the search space, any implementation that considers all different thresholds to discretise data in a brute force manner would otherwise incur in a large correction for multiple hypothesis testing. And this, in turn, would have a negative effect in power.

We evaluated the performance of our methods on a wide range of simulated data sets, and analysed various local configurations of subgraphs and their effect on the type-I and type-II errors of the methods. We found that both proposed methods successfully control type-I error across all subgraphs configurations, where we report the type-I error in the form of the estimated family-wise error rate. Furthermore we observed that our method

4. Network-guided testing of local neighbourhoods

is well-powered to detect sets of genes that overlap with local neighbourhoods in networks under the proposed model of genetic heterogeneity. We benchmarked **tNeAT** and **tNeAT-WY** against various approaches based on the Tarone procedure and regression based tests, as well as two well-established network approaches, and found that it is competitive with those methods. However, when a significant subgraph does not obey the hypothesis underlying **tNeAT/tNeAT-WY**, i.e. that the significant subgraph corresponds to a full k -hop neighbourhood, we observed a drop in power. Notably, a Tarone-based method that tests individual edges in the network showed consistent power across all subgraph configurations. While this method, called **tEdgeT**, could not outperform the neighbourhood based methods in cases where the associated subgraph formed a k -hop neighbourhood-like subgraph in the networks, it outperformed methods based on local neighbourhoods if the truly-associated subgraph was a random connected component in the graph.

When applying our methods and the comparison partners to various real-world data sets of the plant *Arabidopsis thaliana*, we observed various interesting findings that are in line with observations made on the simulated data sets. For a total of 13 out of 20 *A. thaliana* phenotypes, significant neighbourhoods were detected with at least one method (**tNeAT/tNeAT-WY** or neighbourhood-based comparison partner). This suggests that local neighbourhoods in molecular networks contain valuable information to discriminate phenotypic groups, and hence constitute interesting patterns to analyse. However, we observe that methods based on genetic heterogeneity (**tNeAT** and **tNeAT-WY**) mainly discover small neighbourhoods, the largest discovered neighbourhood was of size three, while regression based approaches that assume additivity between genes in neighbourhoods discovered larger neighbourhoods (maximum size: 35 genes). We furthermore observed that the signal of neighbourhoods discovered with **tNeAT-WY** is in most cases driven by the signal of individual genes or pairs of genes. Combining those two findings motivates the following interpretation: neighbourhoods constitute interesting patterns, but our findings indicate that neighbourhoods might not be meaningful objects to study under the model of genetic heterogeneity, which mainly seems to be due to their size (see Figure 4.7). According to our observations, genetic heterogeneity does not take place on the scale of neighbourhoods in the network, but rather on the scale of genes or gene pairs.

Those findings pave the way for further research, and at the same time motivate the next chapter in this thesis: while neighbourhoods might not be the optimal candidate patterns to mine in biological networks under models of genetic heterogeneity, an interesting further direction is to focus on smaller entities in networks, such as individual interactions. Our findings indicate that edges in the network constitute promising candidates to study when considering genetic heterogeneity.

5. Network-guided testing of gene-pairs

Gumpinger, A.C., Rieck, B., Grimm, G.D., International Headache Genetics Consortium, Borgwardt, K.M. *Network-guided search for genetic heterogeneity between gene pairs*. In press at *OUP Bioinformatics* (2020).

One of the major challenges in genome-wide association studies is the phenomenon of missing heritability. It refers to the fact that individual genetic variants that have shown to be associated to a complex trait only explain a small fraction of the heritability of that trait. In the introduction to GWASs (Section 2) we argued that the missing heritability might be attributed to different phenomena, and one of them is omitting non-linear interaction effects between genetic variants. One such non-linear mode of interaction that was already discussed in previous sections of this thesis is **genetic heterogeneity**, meaning that different genetic variants influence a phenotype in a similar direction. Another mode of interaction is epistasis [120], which refers in the classical sense to non-linear interaction effects between pairs of single nucleotide polymorphisms (SNPs). Given the ever-increasing size of genetic data sets in the number of genotyped or sequenced variants, testing all pairwise interactions between SNPs poses not only a computational, but a statistical challenge due to the simultaneous testing of large numbers of hypothesis. In Chapter 3, we introduced the field of significant pattern mining which provides tools and concepts to address the multiple hypothesis testing problem that commonly arises in large-scale genetic analysis when considering (higher-order) interactions between genetic markers. However, testing all pairwise interactions between genetic markers in data sets with high-dimensional feature spaces largely remains an open challenge.

At the same time, biological networks that describe interactions between genes and their proteins products on various scales have become available for various model organisms. Those networks are themselves subject to the ever-accelerating collection of data, such that they are constantly growing in size, with respect to the number of genetic entities (nodes), and to the number of interactions (edges) (see Chapter 1). Those networks can be interpreted as a form of biological prior knowledge, deeming some interactions between genetic regions, such as genes, more likely than others. As such, they constitute an ideal guide for network analysis, and they have been leveraged in genomic analysis in various settings [e.g. 63, 146]. In the previous chapter we combined genetic data with molecular networks and set out to find neighbourhoods in networks that were associated to the trait of interest under a model of genetic heterogeneity. We found that the interesting interactions took place on the scale of edges, i.e. interactions, in the network, rather than on the level of complete neighbourhoods. This inspired us to analyse smaller network components under the model of genetic heterogeneity, that is individual edges. However, when genes get large, i.e. they are represented by many genetic variants, aggregating *all* variants within a gene might lead to a loss of power if not all variants in the gene are causal,

5. Network-guided testing of gene-pairs

or if they influence the phenotype in different directions. To overcome this, we propose a method that is based on testing interactions between sub-units of genes, so called gene segments, and we analyse interactions between those. However, we restrict our analysis to interactions between gene-segments that come from two genes that are connected in the network. This yields two main advantages: (i) it reduces the otherwise very large hypothesis space, alleviating the computational and statistical burden, and (ii) it yields interpretable interactions, as interactions between genes in the network have a biological interpretation assigned to them.

This chapter is organised as follows: we start with a formal problem statement, and continue to outline how the concepts from significant pattern mining (Chapter 3) can be leveraged to achieve the goal by defining segment interactions as patterns. Next, we explain our main contribution, that is the *Significant Interaction Mining in Networks* (SiNiMin) method, as well as its Westfall-Young permutation based counterpart (SiNiMin-WY). To evaluate the capacities of our methods, we conducted an extensive simulation study to explore different properties of the methods compared to well-established methods such as SKAT-0 [132] and FastLMM [103, 104]. Subsequently, we apply our methods to 20 *Arabidopsis thaliana* phenotypes, and conduct a study of low-frequency variants in migraine patients. We conclude this chapter with a summary and discussion of our results, followed by an outlook for network-guided association studies.

5.1. Finding significant segment interactions

5.1.1. Problem statement and notation

We consider a data set consisting of n samples, where each sample is represented by its g dimensional binary genotype. For each sample, the j^{th} genotype, or feature, could correspond to a genetic variant, such as a SNP or rare variant, in a dominant encoding (see Section 2.2.1). We store the data in an $n \times g$ dimensional data matrix, which we denote as $\mathbf{X} = \{0, 1\}^{n \times g}$. Each of the n samples comes from one of two phenotypic classes, such that a binary label vector $\mathbf{y} = \{0, 1\}^n$ exists. In case there exists an additional categorical covariate for each sample, with a total of r covariate classes, we store this information in the n dimensional vector $\mathbf{c} = \{1, \dots, r\}^n$. Such a covariate vector could correspond to the origin of a sample, or arise from a clustering of non-categorical data describing the samples. An example of such a data set is illustrated in Figure 5.1a. As for the last chapter, we assume the existence of prior knowledge about interactions between genetic regions that can be represented in the form of a graph $\mathcal{G} = (V, E)$, where V corresponds to the set of vertices in the graph, and E to the set of interactions, or edges, between those vertices, such that $E = \{(u, v) \mid u, v \in V\}$. This could for instance correspond to a protein-protein interaction network, where vertices correspond to genes, and edges correspond to physical interactions between the proteins defined by the genes (see Figure 5.1b).

We can combine those two types of data by mapping the genetic variants in the data set \mathbf{X} to vertices in the network \mathcal{G} . One way to do this is for example by physical proximity, i.e. a genetic variant is mapped to a vertex in the network, if it overlaps with the genetic location of the vertex. In case a vertex represents a gene, its genetic location could be defined as the entirety of its exons and introns. An example of this mapping is illustrated in Figure 5.1c.

5. Network-guided testing of gene-pairs

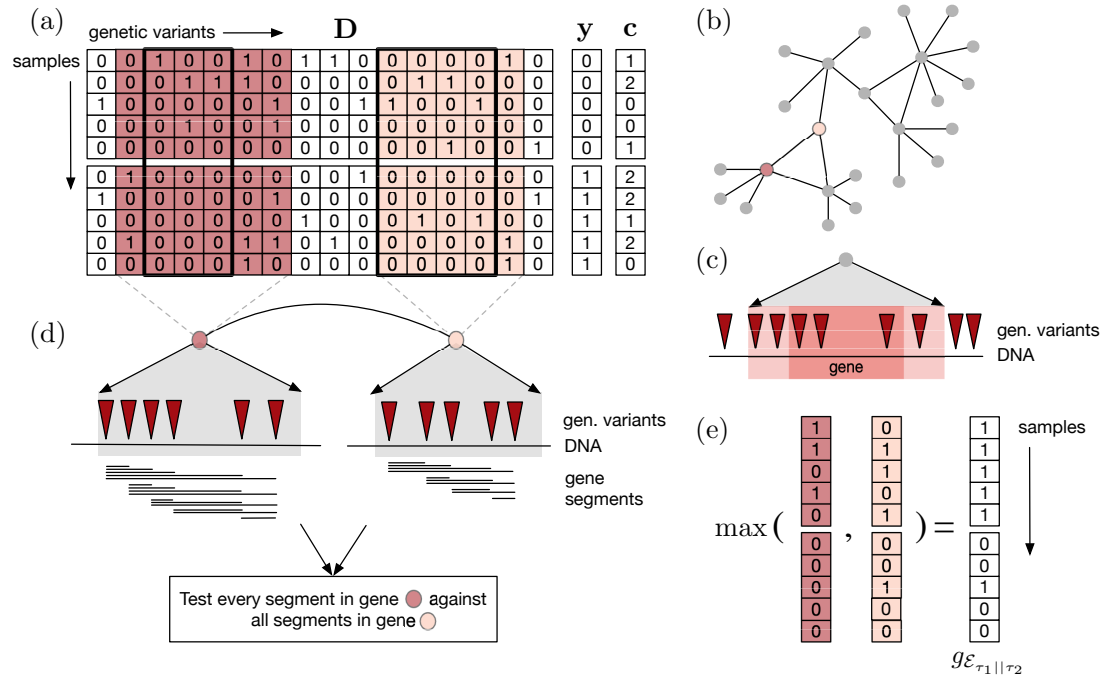


FIGURE 5.1.: Overview of concepts for gene-segment interaction detection. (a) The binary data set \mathbf{X} , the phenotype vector \mathbf{y} and the covariate vector \mathbf{c} . Two genes are highlighted in dark and light red, represented by six and five genetic variants, respectively. For each gene the black box indicates a segment within the gene. (b) The molecular network, where nodes correspond to genes, and edges correspond to interactions between genes. The dark and light red vertices indicate the same genes as in (a), and they are connected with an edge in the network. (c) The representation of a gene by the genetic variants that overlap with its introns and exons (darker red shade), as well as an optional window up- and downstream of the gene (light red). The triangles indicate the position of the genetic variants. (d) Illustration of the gene-segment interactions between genes highlighted in (a) and (b). All possible segments within each of the two genes are indicated with the black horizontal lines, and each possible combination between segments from both genes has to be evaluated. (e) Computation of the pattern indicator function $g_{\mathcal{E}_{\tau_1 || \tau_2}}$ for the two segments highlighted with black boxes in (a). This serves as the representation used for testing in a contingency table. (This figure is adapted from Figure 1 in Gumpinger et al. [124]).

5. Network-guided testing of gene-pairs

For the remainder of this chapter, we assume that vertices in the network correspond to genes. Note however, that networks could in general describe interactions between any type of genetic regions, and that all methods introduced in this chapter are applicable to any type of molecular network. The mapping represents each vertex in the network as a set of genetic variants. The object under study in this project are so-called **genetic segments**, that is a set of subsequent variants along the sequenced or genotyped genome (see Figure 5.1d). We denote a genetic segment as the tuple (s, l) , where s denotes the starting position of the segment, and l corresponds to its length. For example, the segment $(5, 3)$ corresponds to the set of variants at the 5th, 6th, and 7th position of the genome. We restrict ourselves to segments within genes, and refer to those as **gene segments** in the following.

The goal of this project is to discover pairs of gene segments that (i) are statistically significantly associated to the binary phenotype \mathbf{y} under a model of genetic heterogeneity and that (ii) interact within the network, i.e. each segment comes from one of two genes in the network that are connected by an edge.

5.1.2. Patterns as gene segment interactions

We have defined a pattern as any discrete substructure of the set of features \mathcal{J} , i.e. each pattern corresponds to a member of the power set of \mathcal{J} , denoted as $\mathcal{P}(\mathcal{J})$ (see Section 3.2). In the previous chapter, we restricted our analysis to patterns that represent neighbourhoods in an underlying feature network. In this chapter, we will also leverage information of interactions between features in the form of a network, but this time those two data sets live on different scales: while the data matrix represents individual genetic variants, such as SNPs or rare variants, the vertices in the network correspond to genes, i.e. specific regions in the genome. As opposed to the previous approach presented in Chapter 4, each vertex in the network is now represented by a set of consecutive features, instead of a single feature, as described above.

Hence, herein we define a pattern as a gene segment interaction, that is the interaction between any two consecutive feature subsets in any pair of interacting genes. The resulting hypothesis space $\mathcal{H}_{\mathcal{G}}$ will contain all possible such segment interactions. More formally, we describe a vertex $v \subset \mathcal{J}$ as a subset of the g features in the dataset, and we assume the set to be ordered in a meaningful way, e.g. the genetic variants are ordered by their position in the genome. A segment $\tau = (s, l)$ is then formally defined as an ordered subset of the vertex, i.e. $\tau \subset v$, where s denotes its starting position and l denotes its length. This leads to the following formal definition of a segment:

$$\tau = (s, l), \text{ corresponding to } \{j \mid s \leq j < s + l, j \subseteq v \subset \mathcal{J}\} \quad (5.1)$$

Given two segments $\tau_1 = (s_1, l_1)$ and $\tau_2 = (s_2, l_2)$ pertaining to two interacting genes (i.e. vertices) v_1 and v_2 in the network, where $v_1, v_2 \in V$, and $(v_1, v_2) \in E$, we denote the gene-segment interaction between segments τ_1 and τ_2 as $\mathcal{E}_{\tau_1 \parallel \tau_2}$. It corresponds to the union of features that fall in any of the two segments τ_1 or τ_2 , i.e.

$$\mathcal{E}_{\tau_1 \parallel \tau_2} = \tau_1 \cup \tau_2. \quad (5.2)$$

5. Network-guided testing of gene-pairs

By defining patterns as segment interactions, patterns naturally exhibit strong sub- and super-pattern relationships. This is the case already on the level of segments, and propagates to the level of segment interactions. Each segment $\tau = (s, l)$ is a super-pattern to $\frac{1}{2}l(l+1)$ other patterns. This implies that on the level of interactions the same sub- and super-pattern relationships hold. Given two segments $\tau_1 = (s_1, l_1)$ and $\tau_2 = (s_2, l_2)$, the interaction $\mathcal{E}_{\tau_1 \parallel \tau_2}$ is a super-pattern to $\frac{1}{4} l_1(l_1+1) l_2(l_2+1)$ segment interactions.

5.1.3. Significant pattern mining to find significant gene segment interactions

Our goal is to test interactions between gene segments for their association to the phenotype under a model of genetic heterogeneity. Due to the ever-increasing sizes of genetic data sets and networks, the hypothesis space containing all such segment interactions \mathcal{H}_G is large. It can effectively contain millions of pattern. This poses three challenges: (i) the enumeration of all segment interactions between any pair of interacting genes, i.e. within any edge in the network, (ii) the statistical testing of each hypothesis, which induces a large multiple hypothesis testing burden, and (iii) the correction for the dependency structure between the patterns.

The framework of significant pattern mining offers tools and concepts to address all three of the above mentioned challenges. Those concepts were introduced in great detail in Section 3, and we resort to a brief description of the main idea in the following. At the heart of those significant pattern mining approaches lies an observation made by Tarone [183] that facilitates the control of the family-wise error rate: for a discrete test statistic, a minimum p -value can be computed based on the margins of a contingency table for each hypothesis. Only if the minimum p -value lies below the significance threshold, a pattern can potentially become significant, and contributes to the family-wise error rate. Those hypothesis are called testable, and the number of testable hypothesis is used to compute an improved version of the classically conservative Bonferroni correction that still guarantees control of the FWER, but increases statistical power to detect significant hypothesis. Research in the field of significant pattern mining has leveraged this idea in various forms [158, 160, 161, 163], and the key concepts were presented in Chapter 3, where we showed how Tarone’s concept can be integrated with classical contingency based tests such as Pearson’s χ^2 test and Fisher’s exact test (Section 3.4). We furthermore introduced how Tarone’s idea can improve permutation-based testing with Westfall-Young permutations (Section 3.5), and how it can be incorporated into the Cochran-Mantel-Haenszel (CMH) test that extends Pearson’s χ^2 test such that categorical covariates can be incorporated (see Section 3.6).

The significant pattern mining framework offers the computational and statistical power required to enable testing of segment interactions in genes, which would otherwise be infeasible for large molecular networks due to computational and statistical limitations. In order to fully leverage those tools, we need to reformulate the problem of finding significant segment interactions, which implies that the following two requirements are satisfied:

- i. we have to define a binary pattern indicator function $g_S(\cdot)$ that indicates whether a pattern is present for a given sample, and

5. Network-guided testing of gene-pairs

TABLE 5.1.: Contingency table for a gene segment interaction pattern $\mathcal{E}_{\tau_1||\tau_2}$, where τ_1 and τ_2 correspond to two gene segments, coming from genes that are connected in the underlying graph \mathcal{G} . For the ease of notation, we drop the subscript $\mathcal{E}_{\tau_1||\tau_2}$ and write \mathcal{E} in the pattern indicator function instead. In case of Fisher’s exact test and Pearson’s χ^2 test this contingency table is created once for every pattern. In case of the CMH test, this contingency table is built r times for each pattern, one for each covariate class in the data set (see Section 3.6.1 for details).

	$g_{\mathcal{E}} = 1$	$g_{\mathcal{E}} = 0$	total row
$y = 1$	$a_{\tau_1 \tau_2}$	$n_1 - a_{\tau_1 \tau_2}$	n_1
$y = 0$	$z_{\tau_1 \tau_2} - a_{si}$	$n - n_1 - (z_{\tau_1 \tau_2} - a_{\tau_1 \tau_2})$	$n - n_1$
column total	$z_{\tau_1 \tau_2}$	$n - z_{\tau_1 \tau_2}$	n

- ii. the patterns can be organised in a pattern enumeration tree that allows for efficient pruning of the hypothesis space.

The first requirement is necessary for the generation of a contingency table, that is the representation of a hypothesis in significant pattern mining. From a biological perspective, it allows us to formulate our prior assumption about the mode of interaction between genetic components, and we focus on the analysis of genetic heterogeneity. The second requirement is essential for the computational efficiency of our approach. A well-chosen organisation of the hypothesis space enables pruning large parts of it, such that a large fraction of hypothesis does not have to be enumerated and tested. We will address both requirements in the following.

5.1.3.1. Definition of a pattern indicator function

In the problem statement, we assumed the existence of a binary data matrix $\mathbf{X} \in \{0, 1\}^{n \times g}$, where each of the g features corresponds to a genetic variant, and $\mathbf{X}_{i,j} = 1$ could indicate the existence of a mutation of variant j for sample i .

We create a binary representation of a segment interaction $\mathcal{E}_{\tau_1||\tau_2}$ in a two-step process. For this purpose, we first define the pattern indicator function for any general pattern $\mathcal{S} \subset \mathcal{J}$:

$$g_{\mathcal{S}}(\mathbf{X}_i) = \max(\{D_{i,j} \mid j \in \mathcal{S}\}). \quad (5.3)$$

This function encodes a pattern such that for the i^{th} sample, the pattern $\mathcal{S} \subset \mathcal{J}$ is present, if any of the features in the pattern are present. This was introduced as the maximum pattern indicator function in Equation 3.3 in Section 3.2.

In order to represent a segment interaction, we apply this pattern indicator first on the individual segments in the interaction, and subsequently on their resulting binary representations to represent the interaction, by taking the maximum. Formally, this is defined as

$$g_{\mathcal{E}_{\tau_1||\tau_2}}(\mathbf{X}_i) = \max(g_{\tau_1}(\mathbf{X}_i), g_{\tau_2}(\mathbf{X}_i)), \quad (5.4)$$

and is illustrated in Figure 5.1e. While this is equivalent to applying the pattern indicator

5. Network-guided testing of gene-pairs

function on the set $\tau_1 \cup \tau_2$ directly, this two-step procedure is computationally more efficient as the representation of each interval τ_1 and τ_2 can be stored, and hence we do not have to recompute it for every edge adjacent to vertices v_1 and v_2 , respectively.

Furthermore, this encoding emphasises two different types of genetic heterogeneity, that is *allelic* heterogeneity and *locus* heterogeneity. While allelic heterogeneity refers to genetic heterogeneity within a single locus, such as a gene, locus heterogeneity refers to genetic heterogeneity across different loci, i.e. multiple loci might lead to similar phenotypes. The combination of SNPs within each segment $g_{\tau_1}(\cdot)$ and $g_{\tau_2}(\cdot)$ represents allelic heterogeneity within a gene, since multiple variants in the segment lead to the same binary representation of the segment. The pairwise combination of segments $\mathcal{E}_{\tau_1 \parallel \tau_2}$ in turn describes locus heterogeneity, since the presence of a 1 at any one of the two loci, τ_1 and τ_2 results in the same binary representation.

The binary representation of a segment interaction, $\mathcal{E}_{\tau_1 \parallel \tau_2}$, can then be used to derive a contingency table (see Table 5.1), the central representation of a hypothesis in the significant pattern mining based approaches. We would like to note that in Table 5.1 a single contingency table is shown, which corresponds to the representation in Pearson's χ^2 test and Fisher's exact test. In the case of the CMH test, one such contingency table is derived for each of the r covariate classes, i.e. all samples with the same covariate class are grouped together. In those cases, all table counts are r dimensional vectors, as opposed to scalars.

5.1.3.2. Pattern enumeration

In order to derive a scheme for the enumeration of patterns that enables the efficient pruning of the search space, we start by formalising the hypothesis space. It contains all possible segment interactions between all interacting genes, i.e.

$$\mathcal{H}_G = \{ \mathcal{E}_{\tau_1 \parallel \tau_2} \mid \tau_1 \subseteq v_1, \tau_2 \subseteq v_2, v_1, v_2 \in \mathcal{J}, (v_1, v_2) \in E \} \quad (5.5)$$

The size of the hypothesis space depends on various parameters, that is (i) the number of edges m in the network, (ii) the size of the genes, i.e. the number of variants they are represented with, and (iii) the distribution of gene sizes across the network. The importance of the third point is discussed in more detail below, when analysing the runtime of our proposed approaches.

Especially for large molecular networks with thousands of nodes, the hypothesis space potentially becomes very large, such that an exhaustive exploration of all segment interactions becomes computationally challenging. However, we can exploit monotonicity criteria of the supports of patterns ($z_{\tau_1 \parallel \tau_2}$ in Table 5.1) to prune the search space, and avoid enumerating untestable patterns. Analogously to the approaches presented in Sections 3.4 and 4.1.3.2, we are going to exploit the Apriori property of frequent itemset mining to derive this monotonicity, see Proposition 3.4.1.

Proposition 5.1.1 (Monotonicity of support). *Assuming we are given interactions between two segments τ_1 and τ_2 , and their support is denoted by $z_{\tau_1 \parallel \tau_2}$. For any sub-segment $\hat{\tau}_1 \subseteq \tau_1$ and any sub-segment $\hat{\tau}_2 \subseteq \tau_2$, the support $z_{\hat{\tau}_1 \parallel \hat{\tau}_2}$ fulfills the following monotonicity*

5. Network-guided testing of gene-pairs

Algorithm 5.1 SiNIMin (*significant Network Interaction Mining*), main body

Input: data matrix \mathbf{X} , class labels \mathbf{y} , covariate vector \mathbf{c} (optional), edge-list E , target family-wise error rate α ;

Output: corrected significance threshold δ_{Tar} , set of significant segment interactions $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}})$;

```

1: Initialize global  $\hat{\delta} \leftarrow 1$ , global  $\mathcal{H}_{\text{test}} \leftarrow \emptyset$ ,
2: for  $(v_1, v_2) \in E$  do ▷ Iterate all edges.
3:   PROCESS_EDGE( $\mathbf{X}, \mathbf{c}, (v_0, v_1)$ )
4: end for
5:  $\delta_{\text{Tar}} \leftarrow \alpha / |\mathcal{H}_{\text{test}}|$  ▷ corrected significance threshold.
6:  $\mathcal{H}_{\text{sig}}(\delta_{\text{Tar}}) \leftarrow \{ \mathcal{E}_{\tau_1 || \tau_2} \mid \mathcal{E}_{\tau_1 || \tau_2} \in \mathcal{H}_{\text{test}} \text{ and } p_{\tau_1 || \tau_2} \leq \delta_{\text{Tar}} \}$  ▷ find significant patterns.

```

criterion:

$$z_{\hat{\tau}_1 || \hat{\tau}_2} \leq z_{\tau_1 || \tau_2}. \quad (5.6)$$

Proof. Follows directly from proof of Apriori property in Proposition 3.4.1. \square

Remark 5.1.1 (Monotonicity in the CMH test). *The monotonicity in Proposition 5.1.1 assumes a scalar support $z_{\tau_1 || \tau_2}$, which is valid in Pearson’s χ^2 test and Fisher’s exact test. For the CMH test, the support is an r dimensional vector, where r corresponds to the number of covariate classes. In this case, the monotonicity holds element-wise, i.e.*

$$z_{\hat{\tau}_1 || \hat{\tau}_2}^{r_i} \leq z_{\tau_1 || \tau_2}^{r_i}, \quad (5.7)$$

where r_i marks the support in the r_i^{th} contingency table, $1 \leq r_i \leq r$.

This monotonicity criterion can be used to enumerate the hypothesis space, and potentially prune patterns, i.e. segment interactions, from the search space. We have seen in the introduction to significant pattern mining (Section 3.4.2 and 3.6.2.1) that the pruning condition for all three discussed tests only depends on the support of the pattern, and that, once a pattern is found to be prunable, all patterns with higher support can be pruned from the search space as well. When patterns correspond to segment interactions, we enumerate them such that this monotonicity can be exploited efficiently, that is by successively growing patterns. The exact enumeration procedure is explained in the next section by means of pseudo-code.

5.2. The SiNIMin method

In this section, we introduce our method *Significant Network-Interaction Mining*, short SiNIMin. It is a novel algorithm to find interactions between gene segments that are (i) statistically significantly associated to a binary phenotype of interest, and (ii) originate from two genes interacting according to a molecular network \mathcal{G} . It addresses computational and statistical challenges associated with the exploration of a large hypothesis space by applying concepts from significant pattern mining. To be precise, it leverages Tarone’s procedure to derive an improved Bonferroni threshold δ_{Tar} , and tests all gene segment interactions for their association with a binary phenotype at that threshold. The main

5. Network-guided testing of gene-pairs

routine is outlined in Algorithm 5.1. The pseudocode is kept general, such that it is valid for Person’s χ^2 test, Fisher’s exact test and the CMH test. Whenever test-specific modification are required, we indicate them.

Main body

SiNIMin requires the binary $n \times g$ -dimensional data matrix \mathbf{X} , the binary, n -dimensional class labels \mathbf{y} , the edge list E as well as the target FWER α as input. In case a categorical covariate is available, it can be included. In this case the CMH test will be invoked. SiNIMin computes the Tarone-corrected significance threshold δ_{Tar} , and return those segment interactions that are significant at the threshold δ_{Tar} . At initialisation, the testability threshold $\hat{\delta}$ is set to 1, and the list of testable patterns $\mathcal{H}_{\text{test}}$ is set to the empty set (Line 1), both are global variables and accessible to all functions. The core of the method lies in Lines 2-4, and consists of the iteration of all edges in the network. For each edge, the method PROCESS_EDGE is called. The purpose of this function is fourfold: it (i) enumerates all segment interactions within the edge, (ii) adjusts the testability threshold $\hat{\delta}$ to ensure that the FWER is controlled at the target value α , (iii) reports testable patterns, and (iv) potentially prunes the search space. After all edges in the network have been processed, the set $\mathcal{H}_{\text{test}}$ contains all testable patterns at threshold $\hat{\delta}$, and the final Tarone threshold is set in Line 5, using the number of testable hypotheses as the correction factor. Line 6 finds those testable hypothesis that have p -values below δ_{Tar} , hence are significant. Importantly, this is the first time the phenotypes are used.

Processing of edges

The call to the method PROCESS_EDGE in Line 3 of the main SiNIMin body contains the core of the method. It constitutes SiNIMin’s depth-first search strategy, which successively increases the length of the gene segments. This depth-first strategy is outlined in Algorithm 5.2. It consists of four main steps, that is (i) the enumeration of a segment interaction, (ii) the assessment of testability of the interaction, (iii) the adjustment of the testability threshold $\hat{\delta}$ to guarantee the control of the FWER, and (iv) the potential pruning of patterns from the search space. The first step, i.e. the enumeration of segment interactions takes place in Lines 2 to 8 in the pseudocode. Segments are grown in a depth-first manner, i.e. that for a given starting position s_i , all segments starting at this position are enumerated by increasing the length of the segment. Only if the longest possible segment has been explored, the starting position is increased by one, and the length of the segment is reset to 1. This is done for both segments in parallel, such that initially the segment τ_1 in gene v_1 is kept constant, while exploring all segments in the second gene v_2 . Only after all segments in the second gene have been explored in a depth-first search, the segment τ_1 is grown, and again compared against all segments in the second gene. For each of the segment interactions $\mathcal{E}_{\tau_1||\tau_2}$, we compute the support $z_{\tau_1||\tau_2}$ and the corresponding minimum p -value $\Phi(z_{\tau_1||\tau_2})$ (Line 9), depending on the choice of the test. Independently of the test, the pattern is deemed testable if the minimum p -value falls below the testability threshold. In this case, the function PROCESS_TESTABLE is invoked. As in previously presented algorithms (see Algorithm 4.2 or 3.2), the testable pattern is added to the set of testable hypothesis. Since for Tarone’s procedure the estimate of the

5. Network-guided testing of gene-pairs

Algorithm 5.2 SiNIMin functions

```

1: function PROCESS_EDGE( $\mathbf{X}, \mathbf{c}, (v_1, v_2)$ )
2:   for  $s_1 \in v_1$  do                                     ▷ Process first gene.
3:      $l_1 \leftarrow 0$ 
4:     while  $s_1 + l_1 \leq |v_1|$  do
5:        $l_1 \leftarrow l_1 + 1$ ;  $\tau_1 \leftarrow (s_1, l_1)$    ▷ Enumerate segment  $\tau_1$ .
6:       for  $s_2 \in v_2$  do                                 ▷ Process second gene.
7:          $l_2 \leftarrow l_2 + 1$ ;  $l_2 \leftarrow 0$ 
8:         while  $s_2 + l_2 \leq |v_2|$  do                 ▷ Enumerate segment  $\tau_2$ .
9:           Compute the minimum  $p$ -value  $\Phi(z_{\tau_1||\tau_2})$ 
10:          if  $\Phi(z_{\tau_1||\tau_2}) \leq \hat{\delta}$  then             ▷ Evaluate testability.
11:            PROCESS_TESTABLE( $\mathcal{E}_{\tau_1||\tau_2}$ )
12:          end if
13:          if IS_PRUNABLE_SPEC( $z_{\tau_1||\tau_2}$ ) then       ▷ Evaluate prunability.
14:            break
15:          end if
16:        end while
17:      end for
18:    end while
19:  end for
20: end function
21:
22: function PROCESS_TESTABLE( $\mathcal{E}_{\tau_1||\tau_2}$ )
23:    $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \cup \mathcal{E}_{\tau_1||\tau_2}$        ▷ Add pattern to testables.
24:   while  $\hat{\delta} \times |\mathcal{H}_{\text{test}}| \geq \alpha$  do
25:     Decrease  $\hat{\delta}$                                        ▷ Adapt significance threshold.
26:      $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \setminus \{z_{\tau_1||\tau_2} \mid \Phi(z_{\tau_1||\tau_2}) \geq \hat{\delta}\}$ 
27:     end while                                         ▷ remove untestables.
28: end function
29:
30: function IS_PRUNABLE_NO_COV( $z_{\tau_1||\tau_2}$ )                ▷ Pearson's  $\chi^2$  or Fisher's exact test.
31:   Compute minimum  $p$ -value  $\Phi(z_{\tau_1||\tau_2})$            ▷ described in Section 3.3.4.1.
32:   if  $z_{\tau_1||\tau_2} \geq n - n_1$  and  $\Phi(z_{\tau_1||\tau_2}) > \hat{\delta}$  then
33:     return True                                       ▷ Pattern can be pruned.
34:   else
35:     return False                                     ▷ Pattern cannot be pruned.
36:   end if
37: end function
38:
39: function IS_PRUNABLE_COV( $z_{\tau_1||\tau_2}$ )                  ▷ CMH test.
40:   Compute lower envelope  $\tilde{\Phi}(z_{\tau_1||\tau_2})$ ,           ▷ described in Section 3.6.2.1.
41:   if  $z_{\tau_1||\tau_2}^{r_i} \geq n^{r_i} - n_1^{r_i} \forall r_i \in [1, \dots, r]$  and  $\tilde{\Phi}(z_{\tau_1||\tau_2}) > \hat{\delta}$  then
42:     return True                                       ▷ Pattern can be pruned.
43:   else
44:     return False                                     ▷ Pattern cannot be pruned.
45:   end if
46: end function

```

5. Network-guided testing of gene-pairs

FWER is computed based on the number of testable hypotheses (see Section 3.3.4.2), any addition to the set requires the re-evaluation of the current FWER estimate. Lines 24-27 check whether control of the FWER is still guaranteed, and adapt the testability threshold if this is not the case. Lowering the threshold might deem previously testable patterns untestable, such that those have to be removed from $\mathcal{H}_{\text{test}}$ in Line 26. Since the testability threshold $\hat{\delta}$ only decreases during execution, untestable patterns can never become testable, guaranteeing that the correction remains valid. After the segment interaction has been processed, Lines 13 to 15 check if super-patterns can be pruned from the search space.

Testing whether a pattern is prunable depends on the choice of test. In the case of Pearson’s χ^2 test and Fisher’s exact test, the super-patterns of any untestable pattern whose support fulfils $z_{\tau_1||\tau_2} \geq n - n_1$ can be pruned from the hypothesis space (see Section 3.4.2 for details). This test for prunability is computed by invoking the function `IS_PRUNABLE_NO_COV` in Lines 30-37 in Algorithm 5.2. For the CMH test it is more involved to check whether super-patterns of a segment interaction can be pruned (see function `IS_PRUNABLE_COV`). First, it is important to note that in the case of the CMH test, the support $z_{\tau_1||\tau_2}$ is an r dimensional vector, as opposed to a scalar. Checking the prunability is based on the lower envelope of a pattern, which is defined as the smallest p -value obtained for any super-pattern, and denoted as $\tilde{\Phi}(z_{\tau_1||\tau_2})$ (computed in Line 40). If this value lies above the current testability threshold $\hat{\delta}$, and furthermore $z_{\tau_1||\tau_2}^{r_i} \geq n^{r_i} - n_1^{r_i}$ is fulfilled for all covariate classes $r_i \in [1, \dots, r]$, the super-patterns of the current segment interactions can be pruned from the search space (see Section 3.6.2.1 for the more detailed explanation of prunability in the case of the CMH test).

5.3. The SiNIMin-WY method

Gene segment interactions exhibit by definition strong sub- and superset relationships (see Section 5.1.2). This violates the assumption of independence between patterns that underlies parametric significance tests, and reduces the statistical power in significance testing, as the dependence between patterns affects the null-distribution of the test-statistics. In Section 3.5, we introduced Westfall-Young permutations that address exactly this problem. Instead of assuming a parametric null-distribution, they estimate the distribution from the data by means of random permutations of the label vector, which can greatly improve the statistical power of tests, but comes at the price of an increased computational cost. We showed how Tarone’s procedure can be leveraged to enhance the costly Westfall-Young permutations for generic significant pattern mining approaches in Section 3.5. Here, we present the Westfall-Young based counterpart of the SiNIMin approach, which we call SiNIMin-WY (*Significant Network-Interaction Mining with Westfall-Young permutations*). The main body of this method is outlined in Algorithm 5.3. It differs from the SiNIMin approach (Algorithms 5.1 and 5.2) in the estimation of the family-wise error rate (see Section 3.5 for details). While for SiNIMin the FWER is estimated by means of the testable hypothesis, it is estimated from permutations in the SiNIMin-WY approach. We highlight additions and changes required to incorporate the permutations in the SiNIMin-WY pseudocode compared to SiNIMin in red.

5. Network-guided testing of gene-pairs

Algorithm 5.3 SiNIMin-WY (*significant Network Interaction Mining with Westfall-Young permutation testing*), main body

Input: data matrix \mathbf{X} , class labels \mathbf{y} , covariate vector \mathbf{c} (optional), edge-list E , target family-wise error rate α , **number of permutations** n_p

Output: corrected significance threshold δ_{Perm} , set of significant segment interactions $\mathcal{H}_{\text{sig}}(\delta_{\text{Perm}})$

- 1: Initialise **global** $\hat{\delta} \leftarrow 1$, **global** $\mathcal{H}_{\text{test}} \leftarrow \emptyset$,
- 2: **for** $i \in \{0, \dots, n_p\}$ **do** ▷ initialisation of WY variables (see Section 3.5)
- 3: Initialise **global** $\mathbf{y}^{(p)} \leftarrow$ random permutation of \mathbf{y}
- 4: Initialise **global** $p_{\min}^{(p)} \leftarrow 1$
- 5: **end for**
- 6: **for** $(v_1, v_2) \in E$ **do**
- 7: PROCESS_EDGE_WY($\mathbf{X}, \mathbf{c}, (v_0, v_1)$)
- 8: **end for**
- 9: $\delta_{\text{Perm}} \leftarrow \alpha$ -quantile of $\left\{ p_{\min}^{(p)} \right\}_{p=1}^{n_p}$
- 10: $\mathcal{H}_{\text{sig}}(\delta_{\text{Perm}}) \leftarrow \left\{ \mathcal{E}_{\tau_1 || \tau_2} \mid \mathcal{E}_{\tau_1 || \tau_2} \in \mathcal{H}_{\text{test}} \text{ and } p_{\tau_1 || \tau_2} \leq \delta_{\text{Perm}} \right\}$

Main body

The SiNIMin-WY method requires the same input as SiNIMin, i.e. the data matrix \mathbf{X} , the labels \mathbf{y} , the list of edges E in the network, the target family wise error rate α , and optionally the covariate vector \mathbf{c} . Additionally, the number of permutations n_p is required, and a common choice is $n_p = 1'000$. Similarly to the SiNIMin method, SiNIMin-WY returns the corrected significance threshold δ_{Perm} , and the set of significant hypothesis at that threshold δ_{Perm} .

At initialisation, the testability threshold $\hat{\delta}$ is set to 1, and the set of testable hypothesis $\mathcal{H}_{\text{test}}$ to the empty set, both are global variables. While $\mathcal{H}_{\text{test}}$ is not required to compute the corrected significance threshold in the case of Westfall-Young permutations, it is used to keep track of those hypothesis that will be tested for significance later on. Lines 2 to 5 in Algorithm 5.3 initialise the parameters specific to the Westfall-Young permutations. Those are the permuted phenotypes $\mathbf{y}^{(p)}$ and an n_p -dimensional vector to store the smallest p -value that has been observed per permutation. This vector is key to the permutation based testing, as it enables control of the FWER and the computation of the corrected significance threshold. Lines 6 to 8 contain the core of the method, that is a call to the function PROCESS_EDGES for each edge in the network. It iterates all hypothesis, thereby iteratively updating the set of testable hypothesis and the testability threshold to guarantee control of the FWER at the desired level. The function is described in Algorithm 5.4 and discussed in greater detail in the next paragraph. After all edges have been processed, all segment interactions have been processed as well (excluding the prunable ones), and $p_{\min}^{(i)}$ will contain the lowest p -value that has been observed for any hypothesis in $\mathcal{H}_{\mathcal{G}}$ for the i^{th} permutation. Hence, the final corrected significance threshold can be chosen as the α -quantile of $\left\{ p_{\min}^{(p)} \right\}_{p=1}^{n_p}$ (Line 9). Line 10 tests all hypotheses that are testable at threshold $\hat{\delta}$ for their significance, and all hypothesis with a p -value below the threshold will be reported as final output (the set $\mathcal{H}_{\text{sig}}(\delta_{\text{Perm}})$) together with the significance threshold δ_{Perm} .

Processing of edges (Westfall-Young)

Algorithm 5.4 contains a description of the `PROCESS_EDGE_WY` function, the central part of the `SiNIMin-WY` method. The enumeration of the gene segment interactions is identical to the enumeration in `SiNIMin` (see Algorithm 5.2), i.e. a gene-wise, nested depth-first search. This implies that for a fixed segment in the first gene, v_1 , all segments in the second gene v_2 are enumerated (Lines 2-8). `SiNIMin-WY` first computes the support $z_{\tau_1||\tau_2}$, and from the support the minimum p -value $\Phi(z_{\tau_1||\tau_2})$ for each enumerated interaction $\mathcal{E}_{\tau_1||\tau_2}$ between segments τ_1 and τ_2 (Line 9). If the interaction is testable, it is processed accordingly by calling the function `PROCESS_TESTABLE_WY`. Upon invoking this function, the pattern $\mathcal{E}_{\tau_1||\tau_2}$ is added to the set of testable hypothesis (Line 23), and the permutation p -values for all n_p permutations are computed for pattern $\mathcal{E}_{\tau_1||\tau_2}$ by calling the function `COMPUTE_PERMUTATION_PVALUES` in Line 24. If the p -value obtained for the i^{th} permutation is smaller than all observed p -values for any other pattern seen so far for the i^{th} permutation, the $p_{\min}^{(p)}$ will be updated accordingly. Hence, $\{p_{\min}^{(p)}\}_{p=1}^{n_p}$ will contain the smallest p -value for each permutation. If this value falls below the testability threshold $\hat{\delta}$, a false positive has occurred, and the threshold $\hat{\delta}$ might have to be lowered to control the FWER. This takes place in Lines 25 through 29 in Algorithm 5.4. After the testable pattern has been processed, the next step is to determine whether super-patterns of $\mathcal{E}_{\tau_1||\tau_2}$ can be pruned from the search space, see Lines 13 to 15. This is done by calling the function `IS_PRUNABLE_NO_COV` or `IS_PRUNABLE_COV` described in Algorithm 5.2. The choice of the function depends on the statistical test. In the case of Fisher’s exact test or Pearson’s χ^2 test, where the support $z_{\tau_1||\tau_2}$ is a scalar, a pattern can be pruned based on the support value and the corresponding minimum p -value $\Phi(z_{\tau_1||\tau_2})$. For the CMH test, where $z_{\tau_1||\tau_2}$ is an r -dimensional vector, the lower envelope $\tilde{\Phi}(z_{\tau_1||\tau_2})$ has to be computed to determine whether or not a pattern qualifies for pruning (see Algorithm 5.2 for details). After processing all segment interactions in the two genes v_1 and v_2 , the function returns to the main algorithm, and will be called to process the next edge in the network.

5.4. Implementation details

Since the hypothesis space linked to testing all gene segment interactions is commonly large, we implemented both methods, `SiNIMin` and `SiNIMin-WY`, in C++ to optimise performance. To improve the computational runtime, we include the following speed-ups that are not described in the pseudocode:

- i. Each gene is processed multiple times. Specifically, the degree of a gene determines how often it is included when invoking the function `PROCESS_EDGE` or `PROCESS_EDGE_WY`. In order to avoid re-computing the same segments multiple times, we created a data class to store the segments and their supports for each gene.
- ii. We furthermore observed a large redundancy of the pattern’s supports, i.e. many patterns share the same support. Hence, in order to avoid re-computing the minimum p -values for the same supports multiple times, we store the supports and their corresponding minimum p -values in a data structure.

5. Network-guided testing of gene-pairs

Algorithm 5.4 SiNIMin-WY functions. Highlighted in red are modifications of the SiNIMin specific functions (Algorithm 5.2) that are necessary to incorporate Westfall-Young permutations.

```

1: function PROCESS_EDGE( $\mathbf{X}, \mathbf{c}, (v_1, v_2)$ )
2:   for  $s_1 \in v_1$  do
3:      $l_1 \leftarrow 0$ 
4:     while  $s_1 + l_1 \leq |v_1|$  do
5:        $l_1 \leftarrow l_1 + 1; \tau_1 \leftarrow (s_1, l_1)$  ▷ Enumerate segment  $\tau_1$ .
6:       for  $s_2 \in v_2$  do
7:          $l_2 \leftarrow l_2 + 1; l_2 \leftarrow 0$ 
8:         while  $s_2 + l_2 \leq |v_2|$  do ▷ Enumerate segment  $\tau_2$ .
9:           Compute the minimum  $p$ -value  $\Phi(z_{\tau_1||\tau_2})$ 
10:          if  $\Phi(z_{\tau_1||\tau_2}) \leq \hat{\delta}$  then
11:            PROCESS_TESTABLE_WY( $\mathcal{E}_{\tau_1||\tau_2}$ )
12:          end if
13:          if IS_PRUNABLE_SPEC( $z_{\tau_1||\tau_2}$ ) then
14:            break
15:          end if
16:        end while
17:      end for
18:    end while
19:  end for
20: end function
21:
22: function PROCESS_TESTABLE_WY( $\mathcal{E}_{\tau_1||\tau_2}$ ) ▷ WY-specific processing.
23:    $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \cup \mathcal{E}_{\tau_1||\tau_2}$ 
24:   COMPUTE_PERMUTATION_PVALUES( $z_{\tau_1||\tau_2}$ )
25:    $\text{FWER}(\hat{\delta}) = \frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \hat{\delta})$ 
26:   while  $\text{FWER}(\hat{\delta}) \geq \alpha$  do
27:     Decrease  $\hat{\delta}$ 
28:      $\text{FWER}(\hat{\delta}) = \frac{1}{n_p} \sum_{p=1}^{n_p} \mathbf{1}(p_{\min}^{(p)} \leq \hat{\delta})$ 
29:      $\mathcal{H}_{\text{test}} \leftarrow \mathcal{H}_{\text{test}} \setminus \{\mathcal{E}_{\tau_1||\tau_2} \mid \Phi(z_{\tau_1||\tau_2}) \geq \hat{\delta}\}$  ▷ remove untestables.
30:   end while
31: end function
32:
33: function COMPUTE_PERMUTATION_PVALUES( $z_{\tau_1||\tau_2}$ )
34:   for  $p \in \{1, \dots, n_p\}$  do
35:     Compute  $p$ -value of  $p^{\text{th}}$  permutation  $p_{\tau_1||\tau_2}^{(p)}$ 
36:      $p_{\min}^{(p)} \leftarrow \min(p_{\min}^{(p)}, p_{\tau_1||\tau_2}^{(p)})$ 
37:   end for
38: end function

```

5. Network-guided testing of gene-pairs

- iii. In the case of `SiNIMin-WY`, we parallelise the computation of p -values of the permutations using OpenMP. This is a trivial parallelisation and reduces execution times.

We implemented a user-friendly command-line tool. All source code, accompanied by a documentation and small example of how to work with the tool is openly available on GitHub at <https://github.com/BorgwardtLab/SiNIMin>.

5.5. Simulation study

In order to show the potential of our newly proposed methods `SiNIMin` and `SiNIMin-WY` to detect gene segment interactions that are associated to a binary trait of interest, we design a comprehensive simulation study. We focus on various criteria to evaluate the performance of our methods, namely (i) the computational runtime, (ii) their type-I and type-II error estimates, as well as (iii) their robustness towards network modifications.

5.5.1. Experimental setup

5.5.1.1. Data set generation

Each simulated data set consists of four different components, that is (i) the network, (ii) the binary data, (iii) the binary phenotype, and (iv) a covariate vector. We generated the networks at random, each containing 75 nodes that represent the genes, and 100 edges, that represent interactions between genes. Next, we generate binary data for $n = 500$ samples. To obtain the binary data matrix $\mathbf{X} \in [0, 1]^{n \times g}$, we first draw the number of genetic variants mapping to each gene from a uniform distribution $\mathcal{U}(1, 10)$, and furthermore allow genes to overlap with a probability of 30.0%, where the overlap is limited to at most half of the variants mapping to a gene. Due to the random sampling of the number of genetic variants per gene, the resulting data matrix $\mathbf{X} \in [0, 1]^{n \times g}$ varies in its dimensionality, such that $75 \leq g \leq 750$. Furthermore, we assign each of the n samples to one of two binary phenotype classes, and to one of two covariate classes, where the covariate is correlated to the phenotype.

For each simulated data set we generate a truly significant segment interaction and a confounded segment interaction. The truly significant interaction is associated to the phenotype, while the confounded interaction is associated to the covariate, which in turn is associated to the phenotype. Both associations follow the model of genetic heterogeneity. We fix the combined length of the interaction to six, and draw the number of genetic variants per gene participating in the interaction at random to be either two or three. In other words, each induced segment interaction comprises two segments $\tau_1 = (s_1, l_1)$ and $\tau_2 = (s_2, 6 - l_1)$, where $l_1 \in \{2, 3\}$. We chose τ_1 and τ_2 such that they fall into two vertices/genes v_1 and v_2 , that are connected in the randomly generated network. Both segment interactions are induced with strengths of associations p_s and p_{con} for the truly significant, and the confounded segment interaction, respectively. For all simulation analyses, we set $p_s = p_{con}$, and vary p_s between 0 and 1, with 0 indicating low association, and 1 indicating a very strong association to the phenotype. A detailed description of the simulation study can be found in Appendix A.

5. Network-guided testing of gene-pairs

5.5.1.2. Comparison partners

We compare our models against a variety of baseline comparison partners that can be broadly categorised into two groups, depending on the genetic entities they analyse, and the degree of interaction information they use: (i) univariate approaches that do not take network information into account and (ii) network-based approaches that consider interaction between genetic loci, where interactions are defined based on the molecular network. We focus on three different tools, or frameworks, to test those entities: (i) Tarone-based methods, (ii) the **FastLMM** framework that uses linear mixed models for testing [103, 104], and (iii) the **SKAT/SKAT-O** framework that implements kernel-burden tests [131, 132]. We would like to note that this is a subset of well-established methods and frameworks that have proven successful in testing the joint effects of genetic variants under different assumptions. For example, the **FastLMM**-based methods use a linear mixed model to measure the joint effect of genetic variants on a phenotype of interest, while **SKAT** and **SKAT-O** are inspired by the classical rare variant burden tests.

Univariate approaches

Comparison partners that fall into this group of methods test one of three different types of genetic variants, that are

- i. individual genetic loci, i.e. for the $n \times g$ dimensional binary data matrix, g different tests are conducted.
- ii. gene segments, i.e. for every of the 75 genes in the data set, all possible gene segments $\tau = (s, l)$ are tested.
- iii. genes, i.e. for every of the 75 genes, one statistical test is conducted.

This leads to the following list of baselines:

- i. **FastLMM-single**, which tests all g genetic variants for their association with the phenotype using the **FastLMM** method [103], followed by a Bonferroni correction to control the FWER.
- ii. **FastLMM-segment**, which tests all gene segments in the 75 genes for their association with the phenotype using the **FastLMM-set** method [104], followed by a Bonferroni correction to control the FWER.
- iii. **SKATO-segment**, which tests all gene segments in the 75 genes for their association with the phenotype using the **SKAT-O** method [132], followed by a Bonferroni correction to control the FWER.
- iv. **FastLMM-gene**, which tests all 75 genes for their association with the phenotype using the **FastLMM-set** method [104], followed by a Bonferroni correction to control the FWER.
- v. **SKATO-gene**, which tests 75 genes for their association with the phenotype using the **SKAT-O** method [132], followed by a Bonferroni correction to control the FWER.

For all of the baselines listed above we include the binary covariate vector into the model to correct for confounding. The purpose of comparing against those methods is to analyse

5. Network-guided testing of gene-pairs

the effect of including interactions into the model for the discovery of the ground-truth association.

Network-based approaches

This group of methods comprises all comparison partners that include the network information, and we focus on the following patterns:

- i. edges, i.e. we conduct one test for each of the m edges in the network.
- ii. gene variant interactions, i.e. we test pairwise genetic variants that stem from genes interacting in the network. This corresponds to a variant of the gene segment interactions, where segment interactions are restricted to length 1.
- iii. gene segment interactions, i.e. the patterns central to our proposed approaches `SiNIMin` and `SiNIMin-WY`.

This results in the following list of comparison partners:

- i. `FastLMM-edge`, which jointly tests the genetic variants falling into any of the 100 edges for their association with the phenotype using the `FastLMM-set` method [104], followed by a Bonferroni correction to control the FWER.
- ii. `SKATO-edge`, which jointly tests the genetic variants falling into any of the 100 edges for their association with the phenotype using the `SKAT-0` method [132], followed by a Bonferroni correction to control the FWER.
- iii. `edgeEpi`, which tests gene-variant interactions while leveraging Tarone’s procedure to improve statistical power. It is identical to the `SiNIMin` approach when restricting the length of the segments to 1.
- iv. `edgeEpi-WY`, which is an extension of `edgeEpi` that leverages Westfall-Young permutations to account for dependence between patterns. It is identical to the `SiNIMin-WY` approach when restricting the length of the segments to 1.
- v. `FastLMM-interact`, which generates segment interactions as sets of features, and conducts a `FastLMM-set` [104] test for each enumerated segment interaction. This requires creating a file that contains a list of variant sets, where each variant corresponds to one segment interaction, prior to the analysis. The analysis is followed by a Bonferroni correction to control the FWER.
- vi. `SKATO-interact`, which generates segment interactions as sets of features, and conducts a `SKAT-0` test [132] for each enumerated segment interaction. As for `FastLMM`, this requires creating a file that contains a list of variant sets, where each variant corresponds to one segment interaction, prior to the analysis. The analysis is followed by a Bonferroni correction to control the FWER.

All methods include the binary covariate vector to correct for confounding. The purpose of this group of comparison partners is to analyse whether different network-based tests reach similar performance. In the case of the `edgeEpi` approaches, the hypothesis is changed with respect to the patterns, but still assumes a model of genetic heterogeneity. In case of `FastLMM-interact` and `SKATO-interact`, different modes of interaction are analysed

5. Network-guided testing of gene-pairs

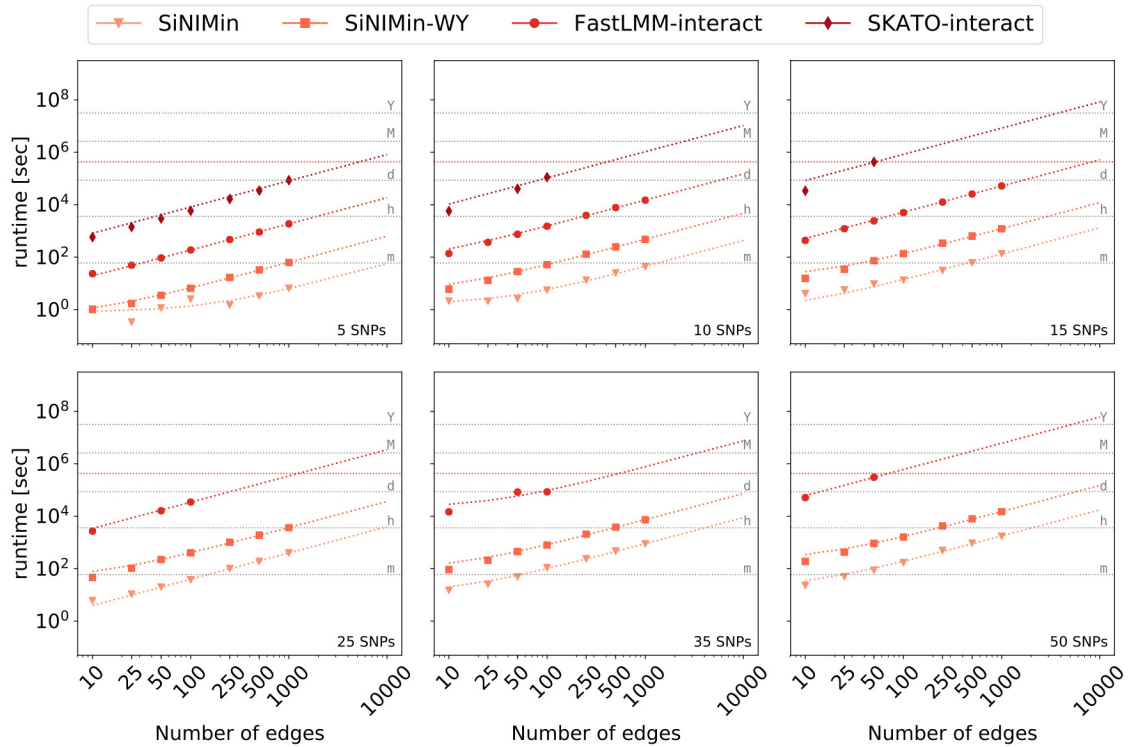


FIGURE 5.2.: Runtimes of methods for the detection of gene segment interactions. The markers indicate measured runtimes of the methods when executed on a high-performance cluster. The dotted lines correspond to linear interpolations and indicate estimates of unmeasured runtimes. The red horizontal line indicates 120h, the maximal tested runtime. Grey lines indicate one minute (m), one hour (h), one day (d), one month (M) and one year (Y). (This figure is adapted from from supplementary Figure S4 in Gumpinger et al. [124]).

on the same patterns. That is, while `SiNIMin`/`SiNIMin-WY` test for genetic heterogeneity, `FastLMM-interact` uses an additive model, and `SKATO-interact` uses a model inspired by burden-tests.

5.5.2. Application to simulated data

We apply our methods and the comparison partners to simulated data. We first show that our methods `SiNIMin` and `SiNIMin-WY` show improved runtimes compared to the two other baselines that test for the association between gene segment interactions and the binary phenotype. Next, we show that our proposed methods control the type-I error, and have larger power than the comparison partners to detect gene-segment interactions. Lastly, we evaluate the effect of network modifications in the form of adding and removing edges to the performance of our methods.

5. Network-guided testing of gene-pairs

5.5.2.1. Runtime analysis

The runtime of methods that test gene segment interactions for association to a phenotype are dominated by two factors, that is the number of edges m in the network, and the number of genetic variants that represent each gene, n_g . Those two factors determine the number of segment interactions that exist for a given data set. We vary those parameters such that $m \in \{10, 25, 50, 100, 250, 500, 1000\}$ and $n_g \in \{5, 10, 15, 25, 35, 50\}$, and simulate 10 data sets for each combination of n_g and m . While in a real data set gene sizes n_g commonly differ for each gene, we leave this number constant. While we have shown that the uniform distribution of genetic variants to genes leads to the worst case runtime [see supplementary Section 1.2 in 124], it gives us a comparable scenario across all simulations. We compare our methods **SiNIMin** and **SiNIMin-WY** against the two most direct competitors that test gene segment interactions, that is **FastLMM-interact** and **SKATO-interact**. Both competitors require the exhaustive enumeration of *all* segment interaction prior to the analysis, a time and memory consuming task. We do *not* count this pre-processing step towards the runtime. Importantly, **SiNIMin** and **SiNIMin-WY** enumerate the segment interaction on the fly, such that the enumeration step is counted towards the runtime for those methods. We compare the execution times of all four methods, when run on a high-performance cluster, and set the time cut-off to 120 h. The resulting runtimes are illustrated in Figure 5.2. We clearly observe that **SiNIMin** shows the most beneficial runtime, followed by its permutation-based counterpart **SiNIMin-WY**. Especially **SKATO-interact** could not be executed for most settings due to its prohibitive runtime. **FastLMM-interact** could be applied to obtain to some of the settings, such that estimates of runtimes for unobserved settings could be computed via extrapolation. We observe that for the most difficult setting, i.e. a network with $m = 1'000$ and $n_g = 50$, a **SiNIMin** execution takes less than an hour and a **SiNIMin-WY** execution takes few hours, while **FastLMM-interact** would require several months to run. One of the main reasons for this improved runtime is the pruning of the search space, i.e. while **FastLMM-interact** and **SKATO-interact** have to test the complete hypothesis space, **SiNIMin** and **SiNIMin-WY** prune the search space, such that only a fraction of hypothesis have to be evaluated.

5.5.2.2. Type-I error analysis: estimation of family-wise error rate

One of the major challenges when testing gene segment interactions for association with a phenotype is the resulting multiple comparisons problem. The commonly-used Bonferroni correction is too conservative and we have seen before that Tarone's procedure yields an improved significance threshold that still guarantees control of the type-I error in the form of the family-wise error rate. To validate that this holds true in practice, we evaluate the empirical FWER by randomly permuting the class labels 1'000 times. If for a permutation a significant association occurs at the given threshold, we know that this corresponds to a false-positive, since the permutations supposedly destroy any true association signal. The empirical FWER is then computed as the fraction of permutations for which at least one such false-positive association occurred. The results of this analysis are illustrated in Figure 5.3. We observe that both methods, **SiNIMin** and **SiNIMin-WY** successfully control the FWER at the target significance level $\alpha = 0.05$ across all association strengths p_s . Furthermore, we observe that the Westfall-Young permutations result in a less stringent

5. Network-guided testing of gene-pairs

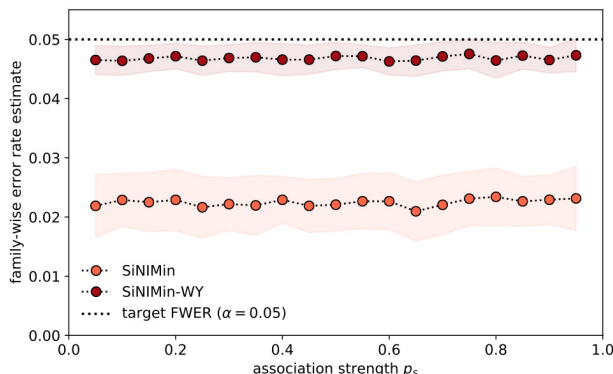


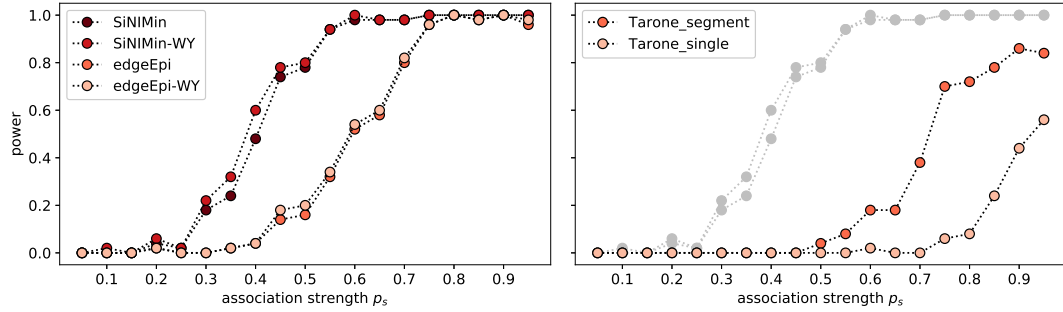
FIGURE 5.3.: Family-wise error rate estimates of SiNIMin/SiNIMin-WY for varying association strengths p_s . The shaded areas around the lines indicate the standard deviation of FWER estimates obtained for the 50 simulations per p_s -values. (This figure is adapted from Figure 2c in Gumpinger et al. [124].

control of the FWER. We will see in the following section that this leads to an increased power for the permutation based approach.

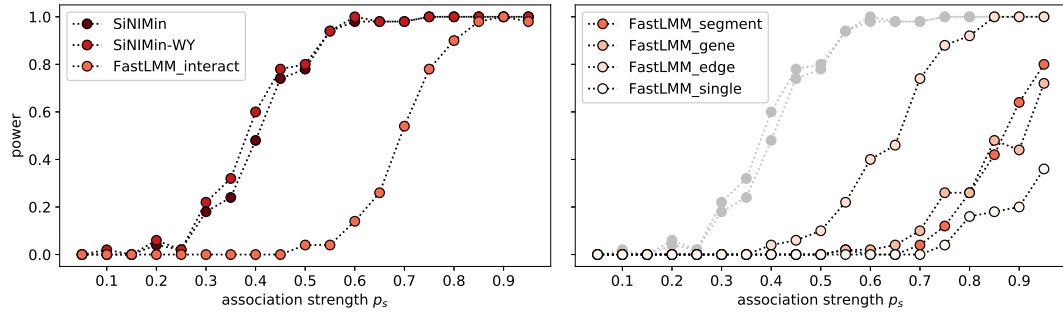
5.5.2.3. Type-II error analysis: estimation of power

Next we set out to evaluate the performance of our method by evaluating the statistical *power*, i.e. their capability to recover the truly significant segment interactions. The results of this analysis are visualised in Figure 5.4 for methods using Tarone’s procedure (Figure 5.4a), FastLMM (Figure 5.4b) and SKAT-O (Figure 5.4c), where plots in the left column contain methods that make use of the network information, and the right column contains methods that do not use the available network information. We simulated artificial data as described in Section 5.5.1.1, where the rather small network size is chosen to allow application of *all* comparison partners. For each association strength p_s we simulated 50 data sets, and the reported power in Figure 5.4 corresponds to the fraction of those data sets for which the truly significant segment interaction was detected. In case of segment interaction methods, we consider the truly-associated segment interaction to be recovered, if it is recovered exactly. In case of the remaining methods we consider it to be recovered if the detected hits overlap with any genetic variant contained in the truly associated segment interaction. We would like to emphasise four main observations made in the power analysis. Firstly, we observe that Tarone-methods based on Westfall-Young permutations (SiNIMin-WY and edgeEpi-WY) show a slight increase in power compared to their non-permutation based counterparts (SiNIMin and edgeEpi, respectively). Secondly, we observe that SiNIMin and SiNIMin-WY outperform all comparison partners that are based on Tarone’s procedure but do not consider gene segment interactions (see Figure 5.4a). Those methods also assume the model of genetic heterogeneity, but differ in the patterns they test for association. Thirdly, SiNIMin and SiNIMin-WY also outperform all methods based on FastLMM, the version that tests segment interactions (FastLMM-interact) as well as all ‘univariate’ methods (Figure 5.4b, right). This shows that if the association is caused by a model of genetic heterogeneity, linear mixed models are not equipped to

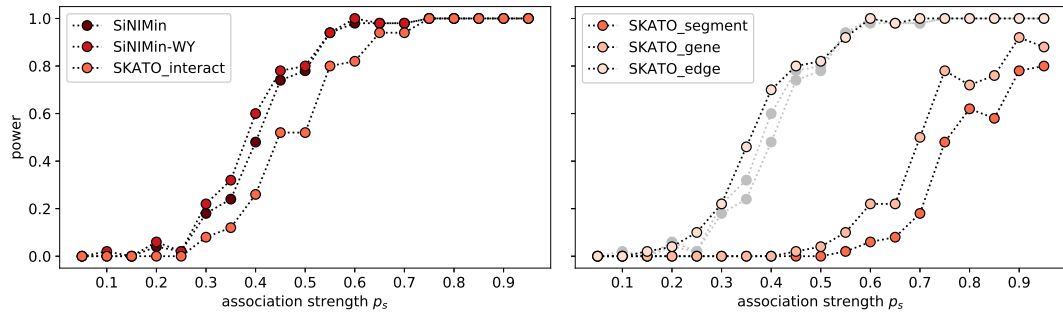
5. Network-guided testing of gene-pairs



(a) Tarone-based methods.



(b) FastLMM-based methods.



(c) SKAT-0-based methods.

FIGURE 5.4.: Power analysis of SiNIMin, SiNIMin-WY and comparison partners on simulated data. (a) Power of methods that build on Tarone's procedure. WY indicates that methods use Westfall-Young permutations to estimate the FWER and compute the significance threshold. (b) Power of methods that use the FastLMM framework, i.e. linear mixed models, to compute p -values. (c) Power of methods that use the SKAT-0 framework, i.e. tests inspired by burden tests. For all plots, the left column describes the power of methods that are based on network-guided associations, the right column those of methods that do not take interactions into account. To ease the comparison, the power of SiNIMin and SiNIMin-WY are indicated in grey in the right column. (This figure is adapted from Figure 2a/b in Gumpinger et al. [124]).

5. Network-guided testing of gene-pairs

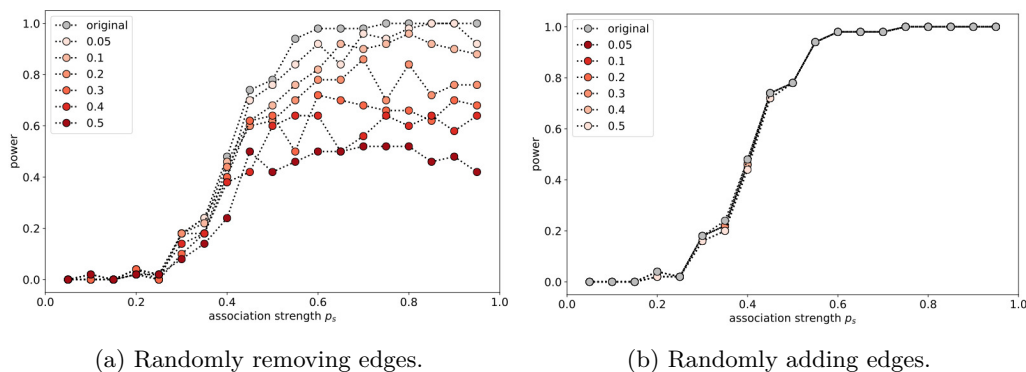


FIGURE 5.5.: Effect of network modifications on power of SiNIMin. (a) random removal of edges from network, (b) random adding of edges to network. The colors indicate the fraction of removed and added edges from the original network with 100 edges, the grey markers indicate power in the original network. (This figure is adapted from supplementary Figure S3 in Gumpinger et al. [124]).

explain the non-linear dependency between genetic variants. The fourth, and presumably most interesting observation, is that testing segment interactions and edges with SKAT-O shows results that are competitive with our proposed methods. While the SKATO-interact performance is comparable, we have seen in Section 5.5.2.1 that its application remains restricted to smaller networks and datasets. The second method that shows comparable performance to our proposed approaches is SKATO-edge, where all genetic variants within two connected genes are tested jointly for their association with the phenotype. While the method is well-powered to recover the true association, it tests all variants jointly, while our proposed methods have the advantage of analysing the data at a higher-resolution: we can pinpoint the variants that drive the association, as opposed to recovering the edge that harbours the signal.

5.5.2.4. Sensitivity to network modifications

Our proposed methods depend on a network to generate the hypothesis space \mathcal{H}_G . In this experiment, we tested the effect of modifying the hypothesis space, by randomly removing and adding edges to the network. The results of this analysis can be found in Figure 5.5. For this study, we used the same simulated data as in Sections 5.5.2.3 and 5.5.2.2, but randomly removed and added a fraction f_r of the original edges, with $f_r \in \{0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$. We observe a strong effect when removing edges from the network (Figure 5.5a): with an increasing fraction of removed edges, the power decreases substantially. This is to be expected, as with increasing values of f_r , the chance to remove the edge that harbours the true association increases as well. On the other hand we observe that our approach is robust to adding up to 50.0% of new edges to the network, which is an encouraging observation. It is a well-known fact that molecular networks are incomplete, as our knowledge about interaction between genetic entities is continuously increasing, and there exists a presumably large number of interactions that yet remain to be discovered [20]. However, one should note that with increasing the hypothesis space, the burden of multiple hypothesis testing is increasing as well, so that substantially increasing

5. Network-guided testing of gene-pairs

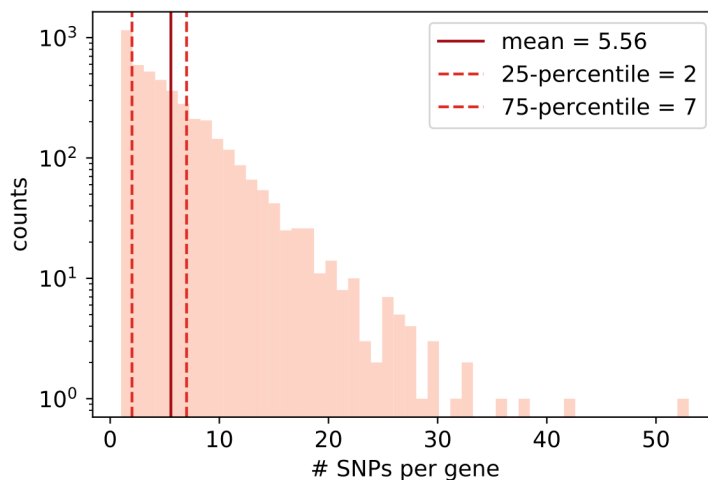


FIGURE 5.6.: Number of SNPs mapping to *A. thaliana* genes. The left dashed horizontal line corresponds to the 25-th percentile, the right dashed horizontal line to the 75-th percentile. On average, each gene is represented by 5.56 SNPs. (This figure is adapted from supplementary Figure S5 in Gumpinger et al. [124]).

the number of interactions might result in a loss of power due to the increased number of tested hypothesis.

5.6. Application to 20 *Arabidopsis thaliana* phenotypes

Our simulation study showed the potential of our proposed methods `SiNIMin` and `SiNIMin-WY` to detect gene segment interactions that are associated to a phenotype of interest under a model of genetic heterogeneity. To show the utility of our methods on real data, we apply it to a widely-used *Arabidopsis thaliana* (short *A. thaliana*) data set created by Atwell et al. [191].

5.6.1. Experimental setup

We downloaded the *A. thaliana* data set collected by Atwell *et al.* from easyGWAS [192], which contains a total of 214'051 SNPs for 1'307 *A. thaliana* samples. We represent each SNP using the dominant encoding introduced in Section 2.2.1, i.e. for each sample, a SNP is encoded as 1 if it contains at least one minor allele. Furthermore there exists a collection of 107 phenotypes, out of which 21 correspond to dichotomous traits. We downloaded those from AraPheno [194], and omit the *YEL* phenotype due to its large class imbalance. For each of the remaining 20 phenotypes, the data sets contain between 84 and 176 accessions (see Table 5.2). To represent interactions between genes, we use the Interactome network [199] that contains 11'373 interactions between 4'866 *A. thaliana* genes. We downloaded gene annotations from AraPort [197], and represent each gene with those SNPs that map to its exons and introns. We removed a gene and its adjacent edges from the network, if the gene could not be represented with at least one SNP in the data set. This led to a total of 4'431 genes and 9'380 interactions from the Interactome network.

5. Network-guided testing of gene-pairs

TABLE 5.2.: Description of *A. thaliana* data sets and their genomic inflation values. The column ‘base λ_{gc} ’ indicates the genomic inflation for the SiNIMin-WY method if one covariate is used, i.e. all samples are in the same covariate class. The column ‘best λ_{gc} ’ contains the lowest genomic inflation value reached when testing $r = 1, \dots, 5$ covariates. The column r contains the corresponding number of covariates that gave the result.

	samples	class n_1	baseline λ_{gc}	r	best λ_{gc}
<i>Anthocyanin10</i>	177	33	1.46	5	1.17
<i>Anthocyanin16</i>	176	70	1.22	5	0.99
<i>Anthocyanin22</i>	177	64	1.43	5	1.10
<i>Chlorosis10</i>	177	28	1.03	1	1.03
<i>Chlorosis16</i>	176	84	0.99	5	0.99
<i>Chlorosis22</i>	176	66	1.77	5	1.10
<i>Emco5</i>	86	17	1.10	4	1.01
<i>Emoy</i>	76	35	1.17	3	1.11
<i>Emwa1</i>	85	32	1.35	3	1.06
<i>Hiks1</i>	84	33	1.23	2	1.16
<i>LES</i>	95	21	1.87	4	1.30
<i>LY</i>	95	29	2.35	3	1.27
<i>Leafroll10</i>	177	78	1.57	5	1.13
<i>Leafroll16</i>	176	37	1.45	5	1.15
<i>Leafroll22</i>	176	31	1.14	1	1.14
<i>Noco2</i>	87	39	1.34	4	1.13
<i>avrB</i>	87	32	1.59	2	1.10
<i>avrPphB</i>	90	44	1.76	5	1.24
<i>avrRpm1</i>	84	28	1.51	2	1.09
<i>avrRpt2</i>	89	17	1.33	2	1.25

In total, 24’571 SNPs were mapped to genes. Figure 5.6 shows the distribution of SNPs to genes. On average, each gene in the network is represented by 5.56 SNPs. This results in an hypothesis space that comprises 6’614’466 gene segment interactions.

We furthermore generated categorical covariates to correct for population structure and cryptic relatedness between samples from the leading principal components of the empirical kinship matrix, following the approach suggested by Llinares-López et al. [165]. We used the leading three principal components, and applied a k -means clustering on those, with k varying between 2 and 10, resulting in $r = \{2, \dots, 5\}$ categorical covariate classes. We chose the ‘best’ categorical covariate by running SiNIMin/SiNIMin-WY and estimating the genomic inflation from the p -values of all testable patterns. We picked the number of covariate classes that resulted in the genomic inflation factor λ_{gc} closest to 1, and report the significant hits found for that number of covariates at the final result. This was done for all methods that rely on Tarone’s procedure (with and without Westfall-Young permutations). An overview of the genomic inflation values is given in Table 5.2.

5.6.2. Results

We apply our methods SiNIMin and SiNIMin-WY to the 20 dichotomous *A. thaliana* phenotypes described in Table 5.2. We analyse our results in three different directions: (i) reduc-

5. Network-guided testing of gene-pairs

tion of genomic inflation by including categorical covariates into the model, (ii) discovery of novel gene-segment interactions with **SiNIMin/SiNIMin-WY** and the comparison partners, and (iii) a qualitative analysis of the novel gene-pairs that harboured the significant gene segment interactions.

As described in the previous section, we include categorical covariates obtained from k -means clustering the leading principal components of the empirical kinship matrix to correct for population structure. Population structure is measured using the genomic inflation factor λ_{gc} , and we target a value of $\lambda_{gc} = 1$ that indicates the absence of structure in the data. While values in the range of $[0.9, 1.1]$ are widely considered acceptable, values of λ_{gc} outside of this range indicate that a correction for population structure should take place. We run **SiNIMin-WY** without covariates, and observe that 18 out of 20 phenotypes show inflation (column ‘baseline λ_{gc} ’ in Table 5.2). Upon including covariates, we observe that the genomic inflation factor decreases for all 18 phenotypes (see column ‘best λ_{gc} ’ in Table 5.2). Although the inflation factor could not be fully reduced to fall into the range $[0.9, 1.1]$ for 11 out of 18 phenotypes, we would like to point out that similar results could be observed in the original publication Atwell et al. [see supplementary material in 191]. However, since the inflation could be reduced consistently across *all* phenotypes, we consider this a success.

Next we set out to analyse the significant hits found with the **SiNIMin** and **SiNIMin-WY** approaches. Table 5.3 contains the number of significant hits that were detected with each of the baseline methods. In this table, each column corresponds to a different method, and the values in the table indicate the significant patterns, as well as the number of network components (i.e. genes and edges) the significant patterns fall into. We observe that **SiNIMin** found significant hits for 6 out of 20 *A. thaliana* phenotypes, and **SiNIMin-WY** found significant hits for 10 out of 20 *A. thaliana* phenotypes, highlighting the improved power of Westfall-Young permutations compared to the parametric test used in **SiNIMin**. Note that we do not report results for **SKATO-interact**, as the execution did not finish within 120 h. We also observe that testing genes and edges with **FastLMM** resulted in significant hits for phenotypes where **SiNIMin** and **SiNIMin-WY** did not recover any significant interactions. It is important to be aware of the difference in hypothesis spaces between those methods, i.e. the hypothesis space analysed in gene-segment interaction methods exceeds the one associated with gene- or edge-based testing by far, leading to a disproportional burden of multiple hypothesis testing. The fairest comparison partner is hence **FastLMM-interact**, which explores the same hypothesis space as **SiNIMin** and **SiNIMin-WY** do. We next determine the novel hits detected with our proposed methods. We consider a **SiNIMin/SiNIMin-WY** hit to be novel if none of the significant hits found with any comparison partners overlaps with the **SiNIMin/SiNIMin-WY** hit. We found this to be the case for 60 segment interactions, mapping to 9 gene-gene interactions across 7 different phenotypes for the **SiNIMin-WY** method (see Table 5.4). In the case of **SiNIMin**, 26 gene segment interactions were detected, falling into two gene interactions across two different phenotypes. Again, the increased amount of significant hits detected for the Westfall-Young permutation based method suggests that permutation tests improve statistical power. When restricting the length of the segments to 1, which yields the **edgeEpi-WY** methods, we discovered two additional SNP-SNP interactions falling into two edges for two different phenotypes.

5. Network-guided testing of gene-pairs

TABLE 5.3.: Number of significant hits found for the different *A. thaliana* phenotypes. The first count indicates the number of significant hits that were found with each method, the number in brackets indicates the number of higher-level genetic entities the hits maps to. For example, for the *avrRpt2* phenotype, SiNIMin-WY detected 114 significant segment interactions that map to 9 gene-interactions. The *Anthocyanin* phenotypes were abbreviated to fit the table. (This table is adapted from supplementary Table S3 in Gumpinger et al. [124]).

	SiNIMin-WY	SiNIMin	edgeEpi-WY	edgeEpi	FastLMM-interact	FastLMM-segment	FastLMM-edge	FastLMM-gene	FastLMM-single	SKATO-interact	SKATO-segment	SKATO-edge	SKATO-gene
<i>Anthocy.10</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Anthocy.16</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Anthocy.22</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Chlorosis10</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Chlorosis16</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Chlorosis22</i>	3 (2)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	1 (1)	1 (1)	4 (2)	x	1 (1)	1 (1)	0 (0)
<i>Emco5</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	15 (15)	15 (15)	2 (2)
<i>Emoy</i>	3 (2)	0 (0)	4 (2)	1 (1)	0 (0)	0 (0)	0 (0)	1 (1)	1 (1)	x	0 (0)	0 (0)	0 (0)
<i>Emwa1</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Hiks1</i>	26 (7)	0 (0)	3 (3)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	1 (1)	x	0 (0)	0 (0)	0 (0)
<i>LES</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	1 (1)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>LY</i>	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	1 (1)	0 (0)	x	5 (5)	5 (5)	1 (1)
<i>Leafroll10</i>	0 (0)	0 (0)	1 (1)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Leafroll16</i>	23 (1)	10 (1)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	1 (1)	x	0 (0)	0 (0)	0 (0)
<i>Leafroll22</i>	16 (1)	16 (1)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>Noco2</i>	2 (1)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	x	0 (0)	0 (0)	0 (0)
<i>avrB</i>	70 (6)	29 (4)	35 (5)	27 (5)	341 (6)	24 (2)	2 (2)	1 (1)	4 (2)	x	0 (0)	0 (0)	1 (1)
<i>avrPphB</i>	15 (1)	10 (1)	4 (1)	4 (1)	13 (1)	6 (1)	1 (1)	1 (1)	2 (1)	x	0 (0)	0 (0)	1 (1)
<i>avrRpm1</i>	74 (6)	43 (5)	37 (5)	28 (5)	827 (6)	20 (2)	2 (2)	2 (2)	4 (2)	x	2 (2)	2 (2)	1 (1)
<i>avrRpt2</i>	114 (9)	82 (9)	108 (11)	102 (11)	50310 (12)	57 (2)	10 (10)	2 (2)	6 (2)	x	7 (7)	7 (7)	3 (3)

We investigated the novel hits found with the SiNIMin-WY method from a biological perspective by using the TAIR resource [195]. We found that all gene-gene interactions in Table 5.4 consisted of gene-pairs that are involved in similar biological processes, are located in same cellular components, share molecular functions or are expressed either in similar plant structures or during similar developmental stages. We list the gene-gene interactions and their biological interpretations in the following. For the *avrB* phenotype and *avrRpm1* phenotype, both phenotypes of bacterial resistance, we discovered significant segment interactions between the two genes AT1G15750 — AT1G17380 with p -values $p = 6.76 \times 10^{-8}$ for *avrB* and $p = 1.15 \times 10^{-7}$ for *avrRpm1*. Both genes are known to be involved in the jasmonic acid-mediated signalling pathway. Jasmonic acid mediates the stress responses in plants [200]. Furthermore the gene AT1G17380 is known to be involved in the defence response and the response to wounding. For the *Leafroll16* phenotype, we found 23 significant segment interactions overlapping with genes AT5G25150 and AT5G45600 ($p = 3.89 \times 10^{-9}$). Both genes encode proteins that form subunits of the General Transcription Factor IID (TFIID) in *A. thaliana* [201]. In the case of the *Leafroll22* phenotype, we discovered 16 segment interactions within the AT3G18490 and the AT5G42980 gene. The most significant segment interaction exhibited a p -value of

5. Network-guided testing of gene-pairs

TABLE 5.4.: The last four columns contain the number of significant gene-segment interactions (SiNIMin, SiNIMin-WY) and SNP interactions (edgeEpi, edgeEpi-WY) in the novel hit. We report the lowest p -value for any pair of segments within the novel hit, and highlight that method in bold for which this p -value is obtained. (This Table is adopted from Table 3 in Gumpinger et al. [124]).

phenotype	gene-gene interaction	p -value	SiNIMin-WY	SiNIMin	edgeEpi-WY	edgeEpi
<i>avrB</i>	AT1G15750 - AT1G17380	6.76×10^{-8}	2	0	0	0
<i>avrRpm1</i>	AT1G15750 - AT1G17380	1.15×10^{-7}	2	0	0	0
<i>Chlorosis22</i>	AT1G74490 - AT2G41100	9.47×10^{-8}	2	0	0	0
<i>Hiks1</i>	AT1G15760 - AT5G43460	2.19×10^{-7}	10	0	0	0
<i>Hiks1</i>	AT4G16845 - AT5G57380	4.26×10^{-8}	1	0	0	0
<i>Hiks1</i>	AT4G19030 - AT5G43460	2.19×10^{-7}	2	0	0	0
<i>Leafroll16</i>	AT5G25150 - AT5G45600	3.89×10^{-9}	23	10	0	0
<i>Leafroll22</i>	AT3G18490 - AT5G42980	5.24×10^{-9}	16	16	0	0
<i>Noco2</i>	AT2G01950 - AT3G43850	1.75×10^{-7}	2	0	0	0.
<i>avrRpt2</i>	AT3G15660 - AT4G15730	1.15×10^{-7}	0	0	1	1
<i>Leafroll10</i>	AT2G04630 - AT3G56270	3.59×10^{-7}	0	0	1	0

$p = 5.24 \times 10^{-9}$. The two genes are involved in similar biological processes: AT3G18490 is involved in the response to water deprivation, and AT5G42980 is involved in the plants heat-response. Other gene-gene interactions that harboured significant segment interactions link related genes, as is the case for the interaction AT4G16845 — AT5G57380, discovered in *Hiks1*, a protist disease resistance phenotype. Both genes are involved in the vernalisation response in *A. thaliana*. However, the interactions connection to the phenotype *Hiks1* remains to be uncovered.

5.7. Study on low-frequency variants in migraine

As a second application, we use our novel methods to analyse two different migraine subtypes, that is *migraine with aura* and *migraine without aura*. Patients that suffer from migraine with aura commonly experience visual disturbances prior to the onset of migraine symptoms, such as flashes of light or blind spots. We set out to analyse the genetic causes discriminating those two subtypes. In this study, we focus on genetic variants with minor allele frequencies below 5.0%. It is important to note that by combining those low-frequency variants using the maximum pattern indicator function (see Equation 5.4), frequencies larger than 5.0% can still be achieved.

5.7.1. Experimental setup

We had access to five different migraine cohorts, namely a Dutch cohort with (DMA) and without aura (DMO), a German cohort with (GMA) and without aura (DMO) and

5. Network-guided testing of gene-pairs

TABLE 5.5.: Dimensions of migraine cohorts. The column ‘SNPs’ indicates the number of SNPs with minor allele frequency less than 0.05 that can be mapped to a gene (including a 50kb window up- and downstream of the gene). The column ‘genes’ indicates the number of genes that can be encoded in this way, and ‘edges’ corresponds to the number of interactions between the encoded genes in the InBio network, i.e. the number of interactions among which segment interactions will be tested. The columns ‘SNP interactions’ and ‘segment interactions’ contain the number of interactions between SNPs and segments that will be tested with the `edgeEpi`-based and `SiNIMin`-based approaches. (This table is adapted from supplementary Table S7 in Gumpinger et al. [124]).

data set	samples	cases	SNPs	genes	edges	SNP inter- actions	segment in- teractions
<i>MaMo</i>	5'013	2'275	15'935	7'994	81'793	629'191	8'743'261
<i>dMaMo</i>	1'849	734	16'285	8'092	82'567	646'407	9'133'634
<i>gMaMo</i>	2'231	1'071	15'947	7'993	81'057	629'923	8'502'931

a Finnish cohort with aura (FMA). We pooled the data from the five different cohorts by combining the cases, and created a binary phenotype by assigning the patients that suffer from migraine with aura the phenotype 1, and the patients that suffer from migraine without aura the phenotype 0. We did this for three different cohort combinations, giving rise to the following three data sets:

MaMo The data set originating from merging patients from all five of the above data sets, resulting in a data set containing 5'013 samples.

gMaMo The data set originating from merging patients from the German cohorts, i.e. GMA and GMO, resulting in a data set containing 2'231 samples.

dMaMo The data set originating from merging patients from the Dutch cohorts, i.e. DMA and DMO, resulting in a data set containing 1'849 samples.

Since we only had access to one Finnish cohort that contained patients suffering from migraine with aura (FMA), Finnish patients are only present in the *MaMo* data set. See Table 5.5 for a detailed description of the data dimensions.

In order to represent interactions between genes, we used the InBio protein-protein interaction network [25]. In order to map low-frequency variants to genes in the network, we first filter all SNPs from the three data sets above that exhibit minor allele frequencies below 5.0%. This resulted in the number of low frequency SNPs listed in Table 5.5. We mapped those variants to genes in the network that overlap with exons and introns of the genes, and additionally those variants that fell within a 50 kb window up- and downstream of the genes (see Figure 5.1c). In case a gene in the InBio network could not be represented with any of the low-frequency variants, we removed the gene and all of its adjacent edges from the network. The number of genes and edges that could be represented for each of the data sets are listed in Table 5.5.

Similarly to the *A. thaliana* data sets, we generate covariates to correct for structure in the data. For the migraine cohorts, we use two different data sources to generate them: (i) the leading principal components of the variance-standardised relationship matrix, and

5. Network-guided testing of gene-pairs

TABLE 5.6.: The genomic inflations λ_{gc} for different types of covariates and the three migraine data sets, obtained with `SiNIMin-WY` and `edgeEpi-WY`. (This table is adapted from supplementary Table S11 in Gumpinger et al. [124]).

(a) <code>SiNIMin-WY</code>			
	<i>gMaMo</i>	<i>dMaMo</i>	<i>MaMo</i>
None	1.19	1.21	4.85
PC	1.14	1.11	1.03
gender	1.19	1.22	4.84
gender + PC	1.17	1.19	1.05
(b) <code>edgeEpi-WY</code>			
	<i>gMaMo</i>	<i>dMaMo</i>	<i>MaMo</i>
None	1.10	1.12	5.19
PC	1.05	1.06	1.14
gender	1.11	1.11	5.17
gender + PC	1.06	1.07	1.15

(ii) the gender of the samples, as migraine is commonly more prevalent in women than in men. Importantly, we use *all* SNPs in the data set to compute the relationship matrix, not only the low-frequency variant ones. We create three different types of covariates, derived from:

- i. the leading principal components (PC), followed by a k -means clustering, such that k determines the number of covariate classes
- ii. the gender, i.e. a covariate vector containing two classes
- iii. the principal components combined with gender, followed by a k -means clustering, such that k determines the number of covariate classes

There are two hyperparameters that affect the genomic inflation, that is the number of leading principal components, as well as the number of covariate classes to generate. In order to determine the best setting, we let both parameters vary between 1 and 10, and conduct a grid search by running our proposed methods for all combinations. We report the one that resulted in the genomic inflation factor λ_{gc} closest to 1. An overview of the obtained genomic inflation values λ_{gc} can be found in Table 5.6.

5.7.2. Results

Similarly to the *A. thaliana* data set, we analyse the results we obtain for the three migraine data sets with respect to three different aspects: (i) the reduction of the genomic inflation factor λ_{gc} , (ii) the number of significant hits found with our newly proposed methods, as well as (iii) a qualitative analysis of the gene segments and the gene-gene interactions those gene segments fall into.

5. Network-guided testing of gene-pairs

As explained in the previous section, we correct for population structure using different types of covariates, derived either from the relationship matrix between patients, gender, or a mixture of both. The genomic inflations λ_{gc} before and after correction with the different types of covariates are displayed in Table 5.6, for the **SiNIMin-WY** and the **edgeEpi-WY** methods. First of all, we observe that the test statistics are inflated, with an especially high inflation in the *MaMo* cohort, i.e. the data set that comprises different populations, while the population-specific data sets *dMaMo* and *gMaMo* only show moderate inflation. When including covariates that are derived from the relationship matrix between samples (columns ‘PC’ and ‘PC + gender’), we observe a drop in inflation, while this is more pronounced for the *MaMo* data set, as it is for the *dMaMo* and *gMaMo* data sets. We hypothesise that the *MaMo* data sets exhibit stronger inflation due to population structure caused by the aggregation of different nationalities in the data set, while in the case of *gMaMo* and *dMaMo*, population admixture might be less pronounced. Principal components of relationship matrices have shown to successfully correct for inflation if the underlying cause is indeed population structure [98]. In the *dMaMo* and *gMaMo* data sets, the reason for the inflation might be a different one, hence explaining the only weak reduction of inflation.

We furthermore observe that using only the gender as a covariate does not lead to a decrease in inflation. This indicates that gender does not confound the results we obtain, and hence including the gender as a covariate does not lead to a reduction of genomic inflation. One possible explanation of this behaviour is the fact that we are *not* comparing migraine patients against healthy controls, but test two different subtypes (migraine *with* and *without* aura). While gender might confound a case-control study, where cases might be enriched with female patients, this does not seem to be the case in this ‘case-case’ study.

Next, we compare the results obtained with our newly-proposed methods **SiNIMin** and **SiNIMin-WY** against the baseline comparison partners. We first focus on the number of significant hits detected with each of the methods. Those results are listed in Table 5.9, where each row indicates one of the comparison partners, and each column indicates one of the three data sets. Those methods marked with an ‘x’ did not finish within 120 hours, and hence were terminated, such that no results can be reported. This is the case for the two comparison methods **FastLMM-interact** and **SKATO-interact**, i.e. the only other methods that test the same hypothesis space as our proposed methods do. This is a first indicator of the utility of our methods, as they are capable of exploring the hypothesis space of all segment interactions, which exceeds the computational capacities of existing methods. We observe that while **SiNIMin-WY** discovers the largest number of segment interactions across all five data sets, when restricting the length of the segments to 1, resulting in the **edgeEpi-WY** method, the obtained gene-gene interactions those significant hits map to are more diverse. For example, for the *MaMo* data set, we discover 1’304 segment interactions mapping to 199 gene interactions with **SiNIMin-WY**, and with **edgeEpi-WY** we discover 554 segment interactions mapping to 209 gene-gene interactions. Similar dynamics can be observed for the *dMaMo* and *gMaMo* data sets as well. We again observe the increase in statistical power when including Westfall-Young permutations, as can be seen by the increased amount of significant hits detected. The set of significant segment interactions detected with **SiNIMin** constitutes a subset of those detected with

5. Network-guided testing of gene-pairs

TABLE 5.8.: Number of significant hits found for the different migraine data sets. The counts in the table indicate the significant patterns detected with each method, the count in parenthesis indicates the number of network components (vertices/edges) the hits map to. For example for the `SiNIMin-WY` method, 1'304 significant segment interactions were detected that map to 199 edges in the network. (This table is adapted from adapted from supplementary Table S8 in Gumpinger et al. [124]).

	<i>MaMo</i>	<i>gMaMo</i>	<i>dMaMo</i>
<code>SiNIMin-WY</code>	1'304 (199)	737 (149)	1'126 (219)
<code>SiNIMin</code>	1'006 (166)	429 (93)	819 (181)
<code>edgeEpi-WY</code>	554 (209)	429 (216)	489 (220)
<code>edgeEpi</code>	527 (202)	319 (169)	467 (215)
<code>FastLMM-interact</code>	x	x	x
<code>FastLMM-segment</code>	35 (22)	40 (12)	40 (17)
<code>FastLMM-edge</code>	155 (155)	228 (228)	203 (203)
<code>FastLMM-gene</code>	18 (18)	10 (10)	16 (16)
<code>FastLMM-single</code>	27 (27)	15 (15)	21 (20)
<code>SKATO-interact</code>	x	x	x
<code>SKATO-segment</code>	20 (20)	40 (40)	x
<code>SKATO-edge</code>	181 (181)	179 (179)	x
<code>SKATO-gene</code>	9 (9)	13 (13)	19 (19)

TABLE 5.9.: Novel hits for migraine cohorts. The last four columns contain the number of significant gene segment interactions (`SiNIMin`, `SiNIMin-WY`) and SNP interactions (`edgeEpi`, `edgeEpi-WY`) in the novel hit. We report the lowest p -value for any pair of segments within the novel hit, and highlight that method in bold for which this p -value is obtained. (This table is adapted from Table S4 in Gumpinger et al. [124]).

data set	gene-interaction	p -value	<code>SiNIMin-WY</code>	<code>SiNIMin</code>	<code>edgeEpi-WY</code>	<code>edgeEpi</code>
<i>dMaMo</i>	EPHA6 - TIAM1	2.64×10^{-8}	1	0	1	1
<i>gMaMo</i>	BMP4 - BMPR1B	1.50×10^{-7}	0	0	1	0
<i>gMaMo</i>	HAO1 - VDAC3	1.33×10^{-7}	0	0	2	0

`SiNIMin-WY`, such that we focus on the significant results obtained with `SiNIMin-WY` for the next analyses.

To analyse the novelty of the results, we define a significant segment interaction to be novel if none of the genes the segment interaction maps to has been discovered by any other method. We would like to note that this puts our methods `SiNIMin` and `SiNIMin-WY` at a disadvantage, as they explore the by far largest hypothesis space, compared to all other explored methods (see Table 5.5), and hence have to cope with an increased multiple hypothesis testing burden. The only methods that are based on the same hypothesis space, that is `FastLMM-interact` and `SKATO-interact`, could not be evaluated on the migraine data sets due to their extensive computational demands.

We find one novel interaction for the *dMaMo* data set between genes EPHA6 and TIAM1

5. Network-guided testing of gene-pairs

($p = 2.64 \times 10^{-8}$). Both genes are involved in ephrin signalling, and the ephrin-B signalling pathway has been previously associated with migraine [202]. Notably, this is an interaction between length-1 segments, and hence could also be detected with `edgeEpi-WY`. When restricting the length of intervals to 1 (`edgeEpi-WY`), we detect two other novel significant hits for the *gMaMo* data sets. One interaction falls into the interaction between genes BMP4 and BMPR1B ($p = 1.50 \times 10^{-7}$). The two genes interact within the BMP signalling pathway, which in turn has been linked to neural development [203]. The interaction between the genes HAO1 and VDAC3 contained a significant segment interaction ($p = 1.33 \times 10^{-7}$). However the edge between those genes is a low-confidence interaction in the InBio network (score 0.247), which means it is not included in any of the underlying pathway or interaction data bases that build the foundation of the InBio network, but it could have been inferred e.g. by ontology. As a result, we could not deduct any biological interpretation of the interaction. However, reporting edges that show no strong evidence in networks so far illustrate the potential of our methods to provide further support for such low-confidence interactions, and might mark the starting point for further experiments.

5.8. Summary and discussion

In this chapter, we introduced two novel methods `SiNIMin` and `SiNIMin-WY`, short for *Significant Network Interaction Mining (with Westfall-Young permutations)* to find sub-units of pairs of genes, so-called segments, that are jointly associated to a binary phenotype of interest under a model of genetic heterogeneity. This approach was inspired by our contribution presented in Chapter 4, where we found that the drivers of genetic heterogeneity seem to be pairwise interactions, rather than higher-order interactions, between genes. While it would be desirable to test all possible such segment-interactions between any two genes in a data set, this creates a hypothesis space that cannot be explored exhaustively, neither from a computational nor a statistical point of view due to the resulting multiple hypothesis testing problem. To alleviate this, we included biological prior knowledge in form of molecular networks into our approach. Instead of exploring all possible segment-interactions between all pairs of genes, we focus our analysis on segment-interactions between pairs of genes that are connected in the network. The benefits of the inclusion of networks are twofold: first, it decreases the size of the hypothesis space, as segment-interactions between non-connected genes are removed, and thereby reduces the burden of multiple hypothesis testing. Second, it creates meaningful interactions that can be interpreted biologically.

Although including network information reduces the hypothesis space, the number of segment interactions to test still lies in the hundreds of thousands or millions, such that the computational and statistical challenges prevail. In Chapter 3, we gave an introduction to significant pattern mining, a field that is devoted to finding patterns that occur more frequently in one of two phenotypic classes. The methods developed for significant pattern mining have shown successful in addressing the statistical and computational challenges associated with the analysis of large-scale biological data sets [158, 164, 165]. Hence, we set out to formulate the problem of finding network-guided pairwise gene-segment interactions as a significant pattern mining problem. This reformulation requires binary data, both for the genotypic and phenotypic data, as well as categorical data for the covariates.

5. Network-guided testing of gene-pairs

While this requirement can be seen as a limitation at first, it enables the exploration of a hypothesis space that cannot not be tackled with other methods and hence deems a previously infeasible problem feasible. Using other methods, that for example rely on linear mixed models, falls short in two aspects: (i) there is no way to prune the search space efficiently, as can be done with approaches based on significant pattern mining, which leads to in an immense computational burden, and (ii) they suffer from an extensive multiple hypothesis testing problem that can prohibit the detection of significant interactions.

We evaluated the performance of our methods with respect to various aspects, namely (i) runtime, (ii) type-I and type-II error control, and (iii) robustness towards network modifications. We compared the runtimes of our methods against two baselines that test gene segment interactions using a linear mixed model approach and a burden-test inspired approach, and showed that our methods outperform the comparison partners significantly: while our methods remained applicable across all simulation scenarios, this was not the case for the baselines, with one of them not being applicable to intermediate-sized data sets. We furthermore confirmed by means of a simulation study that `SiNIMin` and `SiNIMin-WY` successfully control the type-I error, measured as the empirical family-wise error rate, and that they are well-powered to detect true associations in the data. While some of the competitors showed performances comparable to our proposed methods, they all fall short with respect to at least one of the two following aspects: they are either (i) restricted to small data sets due to excessive runtimes on larger data sets, or (ii) cannot resolve the signal at the same level of coarseness, that is on the level of gene-segment interactions. We furthermore evaluated the effect of network modifications onto the performance of our method, and observed that, while the method is robust to adding new interactions to the network, power drops if edges are removed. This is an encouraging observation for two reasons: (i) since molecular networks are known to be incomplete, i.e. there are supposedly many more interactions to be discovered, we will not observe a loss of power, and (ii) it indicates that by adding new edges to the network, we will most likely be able to discover more novel significant interactions.

To show the utility of our method, we applied it to two different real world data sets. The first data set was a set of 20 *A. thaliana* phenotypes, for which we were able to discover novel interactions in 9 out of the 20 phenotypes. The significant segment interactions that were detected fell into 11 gene-gene interactions, and we showed that our detected results are meaningful from a biological point of view. The second data set was a low-frequency study of two migraine subtypes, namely migraine *with* and *without* aura. While all novel significant hits detected with our proposed approaches corresponded to interactions between length-1 segments, i.e. individual genetic markers, our method was the only one to test gene-segment interactions *and* that could be applied to those data sets. All direct competitors that tested gene segment interactions could not be evaluated as their runtime requirements exceeded the capacities of the high-performance computing cluster. In general, the way we assessed the novelty of detected patterns put our methods in disadvantage, as other methods had to explore a significantly smaller hypothesis space, and hence did not suffer from a comparable burden of multiple hypothesis testing.

In contrast to methods that rely on the estimation of joint effects of variants based on the variant’s marginal statistics [e.g. 63, 146, 155], our approach is not restricted to detecting

5. *Network-guided testing of gene-pairs*

interactions in which at least one of the interaction partners shows marginal associations. Rather does the model of genetic heterogeneity allow us to detect interactions that become drivers of a signal upon combination. Furthermore, our method works on a more fine-grained level of genetic entities, that is individual variants, as opposed to whole genes, where signal might get diluted if not all variants within a gene are ‘causal’ or influence a phenotype in the same direction.

In summary, we consider our approaches to be well-powered to detect novel, biologically driven interactions between gene segments, while stringently controlling the number of false positive associations. The following chapter will give an outlook into future work in the realm of network-guided association studies.

6. Discussion and outlook

Summary

This second part of the thesis was devoted to the development of novel algorithms for network-guided genetic association studies. Both methods presented in the preceding chapters are motivated by the problem of missing heritability in genome-wide association studies, i.e. the phenomenon that individual genetic mutations, such as single nucleotide polymorphisms, can in most cases only account for a small fraction of the heritability of a trait. It is a widely-accepted paradigm that parts of the missing heritability could be explained by higher-order, non-linear interaction effects between multiple genetic variants.

The ever-increasing sizes of genetic data sets, especially with respect to the number of sequenced or genotyped variants, pose immense computational and statistical challenges when considering higher order interactions. For example, in a data set with only 100 variants, testing all possible combinations of variants results in $2^{100} \sim 1.268 \times 10^{30}$ statistical hypothesis tests. However, current data sets often contain hundreds of thousands to millions of variants, such that an exhaustive enumeration of interactions is infeasible. Apart from the computational problem of executing such a number of enumeration and testing steps, the sheer amount of pattern imposes a statistical challenge that exceeds the capacities of standard techniques for the absolutely necessary control of false-positives.

The two approaches **tNeAT** and **SiNIMin** (and their respective permutation based counterparts) address these challenges. To achieve this, they build upon two important pillars, the first one being the representation of interactions via molecular networks, and the second one being the exploitation of concepts from significant pattern mining. The idea underlying the incorporation of biological networks to guide interactions is inspired by the observation that genetic entities, such as genes. They do not randomly interact with each other, but are organised in complexes and pathways, and hence some interaction effects are more likely than others. It is further corroborated by the observation that genes that influence the same phenotype often cluster within the network [58]. Hence, in order to reduce the space of all interactions between genetic variants, we proposed to focus on interactions that are supported by the network. However, despite the incorporation of networks, exploring all possible interactions within a network remains computationally and statistically challenging in large networks. Those problems can be addressed with the second pillar, that is the reformulation of the network-guided association study in terms of a significant pattern mining problem. This reformulation provides us a toolbox to make the search for interactions feasible, both from a computational and a statistical point of view. This can be achieved by relying on an observation made by Tarone [183]: in the case of discrete test-statistics, a significance threshold can be found that improves upon the highly conservative Bonferroni correction, while still guaranteeing control of the family wise error rate. This procedure relies on the concept of the minimum p -value of a discrete

6. Discussion and outlook

test, and the observation that some hypotheses cannot become significant at a given significance threshold, in which case they cannot contribute to the number of false positives. Those *untestable* hypotheses hence do not contribute to the family-wise error rate. Not only does this observation reduce the statistical burden of the multiple hypothesis testing problem, it furthermore exhibits beneficial monotonicity properties that enable efficient pruning of the search space, and hence reduce the computational burden.

To this end, we contributed the two methods **tNeAT** and **SiNIMin** that differ in the way the data is represented and aggregated across the network. The **tNeAT** approach assumes a data set that represents each gene in the network with a scalar value in the range $[0, 1]$ for all samples. It explores all k -hop neighbourhoods of a gene in the network, binarised at a set of defined thresholds, and tests them for their association with a binary phenotype under a model of genetic heterogeneity. We evaluated the method on a range of simulated data sets, and applied it to a data set of twenty *Arabidopsis thaliana* phenotypes. While we found significant neighbourhoods that were associated with the phenotype, the main discovery of this project was the fact that interactions under a model of genetic heterogeneity appeared to exist on a level of individual genes, or pairwise interactions between genes, rather than by whole neighbourhoods. This observation sparked the development of our second contribution.

The **SiNIMin** approach works on the level of individual genetic variants, as opposed to genes. Those variants are mapped to genes in a network based on physical proximity, such that each gene in the network can be represented by a set of binary variants for each individual, as opposed to a scalar value in the **tNeAT** approach. Using this type of data, **SiNIMin** searches for gene-segment interactions, where a segment is defined as a range of subsequent variants along the sequenced gene, and we consider interactions between segments for any pair of segments that lie within two interacting genes. In this way, **SiNIMin** is capable of identifying pairs of genes that interact with each other, and that contain regions within the genes that are associated to the phenotype. We applied **SiNIMin** and its permutation-based counterpart **SiNIMin-WY** to a data set of 20 *Arabidopsis thaliana* data sets, and a study of low-frequency variants in migraine patients. Our findings revealed the presence of genetic heterogeneity at the level of gene-segment interactions that could not be discovered with established methods.

Outlook

Network-guided association studies are an exciting field of research that have been approached from different angles. Many valuable contributions have been made to address this challenging task, covering different techniques and concepts. Those are ranging from network regularisation approaches [134, 150, 155], to the greedy enumeration of subgraphs within networks [63, 64, 137] to network propagation approaches [65, 146, 148]. Our contribution to this field differ from existing approaches in various ways:

- i. we address the statistical and computational challenges inherent to network-guided genetic association studies by incorporating concepts and methods from the realm of significant pattern mining,
- ii. our proposed methods do not require any hyperparameters or heuristics,

6. Discussion and outlook

- iii. we explore the complete hypothesis space, the employed pruning criterion only removes tests from the hypothesis space that cannot become significant,
- iv. existing methods that rely on linear models usually assume an additive model of interaction between variants, while we assume a model of genetic heterogeneity,
- v. while existing methods compute summary-statistics of genes and represent those genes by scalars, our **SiNIMin** approach preserves the information of individual variants within the gene. The resulting high-resolution representation of genetic heterogeneity allows us to pinpoint the signal to variants within the gene, as opposed to the whole gene.

Given our proposed methods, we envision several interesting directions of further research in the field of network-guided associations studies supported by significant pattern mining, addressing some of the challenges and limitations of the current approaches.

Exploring other network-derived patterns

Our work has shown that patterns derived from the integration of molecular networks with the variant-based representation of genes seems to be a promising direction. Based on the results presented in Chapters 4 and 5, we hypothesise that genetic heterogeneity takes place on small scales: interactions between individual genes seem to contain more heterogeneity-interactions than larger network modules, such as neighbourhoods. Hence we believe that focussing on interaction effects between small sets of genetic variants, such as SNPs or rare variants, is more meaningful than first aggregating variants into one meta-representation for a gene and then testing combinations of multiple gene meta-representations, as we did in Chapter 4. One major reason for this hypothesis is the fact that the aggregation into gene-wise meta-representatives is very sensitive towards the number of aggregated variants. For large genes, aggregating tens to hundreds of variants dilutes the signal of those variants within a gene that truly follow a model of genetic heterogeneity due to the introduction of noise, thus biasing the results towards interactions between genes where genes are represented with only few variants. By analysing segments within genes, we removed this bias. While we proposed to focus on interactions between segments of genes that are connected in the network, two potential extensions would be to (i) focus on pairwise interactions between all subsets of genetic variants within genes, or (ii) focus on higher-order interactions between multiple segments that form subunits of connected genes as opposed to analyse pairwise interactions.

Approach (i) corresponds to a ‘within-gene itemset mining approach’ that requires the enumeration of all subsets of genetic variants within the gene. This would lead to a total of 2^{n_g} variant-sets for a gene of size n_g , and hence $2^{n_{g_1}} \times 2^{n_{g_2}}$ interactions for every edge in the network, where n_{g_1} and n_{g_2} correspond to the sizes of the adjacent genes. Depending on the size of the data set and the network, this would incur a large computational and statistical burden, and whether or not this remains feasible with Tarone based methods remains to be evaluated.

The second approach (ii), that is the mining of higher-order segment interactions, is an interesting extension as well. However, in order to achieve this, certain design choices that affect the composition of the hypothesis space have to be made. The most general

6. Discussion and outlook

design is to focus on all connected gene modules in the network, and enumerate all possible segment-interactions within those genes. However, this leads to an enormous hypothesis space, which in turn inflicts an extensive computational and statistical burden, that might well exceed the capacities of significant pattern mining approaches. A potential remedy to reduce the hypothesis space is to limit the length of segments, the order of interactions, or the size of the gene-modules that should be explored.

Pattern importance

The second important direction of further research evolves around the idea of pattern importance, interestingness, and the representative value of a pattern. Those concepts are well known in frequent pattern mining, and there exists a multitude of approaches, ranging from relatively simple techniques such as maximal and closed pattern mining, to elaborate generative model approaches [204], or approaches that set out to find the patterns that best describe the essence of a data set [e.g. 205, 206]. The rationale underlying the concept of interesting itemset mining is that patterns are informative, surprising, non-redundant or describe the dynamics in a data set. To some extent, mining significant patterns can already be understood as mining interesting patterns itself. An interesting line of research is to combine concepts from significant pattern mining with those concepts that focus on finding interesting patterns, to further reduce the hypothesis space. One option could be to determine interesting patterns a priori to hypothesis testing, such that the hypothesis space is filtered before testing. However, it might be the case that patterns that are interesting are not informative to discriminate between phenotypic classes. Another idea is to measure interestingness of patterns during the exploration of the hypothesis space, i.e. whenever a pattern is explored and evaluated with a Tarone-based approach, its interestingness with respect to an appropriate measure is evaluated, and the pattern will only be considered if it is deemed interesting. For such an ‘in-line’ approach, the choice of the interestingness measure is crucial, as it has to be deterministic and cannot depend on the previously enumerated patterns, as this might have an effect on the testability.

As opposed to the purely pattern-mining driven derivation of interestingness scores, another potential direction of further research is to focus on the derivation of biologically motivated interestingness. By using network information to guide the generation of segment interactions we already did include biological prior knowledge into significant pattern mining, as it determines which interactions to explore. For example, many biological networks contain the confidence of an interaction as an additional layer of information [e.g. 25] and a potential further development could be to integrate such confidence scores directly into the process of pattern exploration. Different computational biology methods leverage this type of information [64, 65], and especially network propagation approaches have successfully included weighted edges for different types of biological studies. Including edge weights into significant pattern mining approaches to generate and explore patterns could potentially be instrumental to further reduce the hypothesis space arising in network-guided genetic association studies, while at the same time increasing biological interpretability of the significant patterns.

Runtime improvements

One of the major practical shortcomings of the methods driven by significant pattern mining is the computational demand, especially for the permutation-based approaches. This is of high concern from a practical point of view. With the ever-increasing sizes of genetic data sets with respect to the number of features, the number of potential interactions between variants grows even-handedly, whether one includes network information, or not. An important future goal is to ensure that the significant pattern mining inspired methods remain applicable to data sets containing hundreds of thousands or millions of features. This goal could be achieved with two different approaches: (i) through algorithmic development, or (ii) through the reduction of the search space.

To achieve the first goal, that is speeding-up existing and future significant pattern mining approaches, an obvious approach is to exploit modern computational resources. This implies both, aiming at a parallelisation of the methods, as well as developing GPU implementations. While we already incorporated a parallelisation step in our network-guided approaches to compute pattern indicator functions and compute permutation based p -values, parallelising the exploration of multiple patterns at the same time is a non-trivial task due to the various dependencies between patterns. Furthermore, Tarone-based procedures rely on the sequential processing of patterns and updating of the family-wise error rate estimate, such that an update for one pattern would influence the testability of other patterns. Upon parallelisation, those relations have to be considered. To this end, Yoshizoe et al. [207] presented MP-LAMP, a parallelised version of the LAMP algorithm [158], which was the first algorithm to exploit Tarone’s procedure for the discovery of higher-order interactions between genetic variants. The proposed parallelisation relies on the message passing interface which has also been leveraged in other approaches to make the search for higher-order interactions between genetic markers feasible [208]. Furthermore, Terada et al. [209] developed a high-speed version of the computationally expensive Westfall-Young permutations, that relies on parallelisation on a GPU, that achieves a 619-fold runtime improvement compared to standard Westfall-Young permutations.

The second way to speed up the computationally demanding significant pattern mining approaches is through a reduction of the search space. Notably, by integrating molecular networks into the mining process, we already took a first step into this direction. However, this search space could be further reduced, e.g. by incorporating interestingness measures as described in the last paragraph which would reduce the redundancy between patterns.

Extension to real-valued features

One of the key requirements underlying significant pattern mining approaches is the binary nature of the data. Tarone’s principle is based on the application of a discrete test to evaluate the association of a pattern with the phenotype, as the concept of the minimum p -value is only valid if the set of attainable test-statistics is countable. While binary data can be derived relatively easy e.g. for single nucleotide polymorphisms, where a recessive or dominant encoding will result in the desired outcome, this might change for other settings. We observed this in the case of the tNeAT method, each gene was represented by the fraction of minor alleles across the SNPs mapping to the gene. This led to a real-

6. Discussion and outlook

valued data set, and in order to apply principles of significant pattern mining, the data was binarised at various thresholds.

This binarisation has various disadvantages: first and foremost, as opposed to conducting one statistical test for each pattern, we have to conduct as many tests as there are thresholds, imposing an increased burden of multiple hypothesis testing, as well as increasing computational demands. As a consequence, the granularity of thresholds hugely impacts the final outcome, as the multiple hypothesis testing problem is less severe for lower numbers of thresholds. As such, the choice of thresholds is a main concern when working with data that cannot be trivially binarised. By choosing a coarse set of thresholds one might reduce the burden of multiple hypothesis testing, but possibly miss patterns that would have been significant at different thresholds. By choosing a fine-grained set of thresholds, one reduces this risk, but at the same time loses statistical power, i.e. one might miss significant hits due to the increased burden of multiple hypothesis testing imposed by a larger number of thresholds. Hence, working on real-valued features directly, such that only a single test is conducted for each pattern, would help to circumvent this dilemma.

Sugiyama & Borgwardt [210] recently developed an extension of Tarone’s procedure to continuous features, which they named C-Tarone. The main gist of this approach lies in the computation of the *copula support* [211], which estimates the probability to observe a pattern of continuous features. Intuitively, this probability increases, if features in an interaction are ranked similarly. To test for association, a G-test is applied [212]. Its test-statistic is based on the Kulback-Leibler divergence between the observed and expected probabilities of a pattern across two phenotypic classes, which can be computed based on the copula supports, and follows a χ^2 distribution with one degree of freedom. Sugiyama & Borgwardt [210] showed that the test-statistic is upper-bounded, and that the upper bound can be computed based on the copula supports and class ratios. This enables the estimation of a minimum p -value for a pattern, which in turn can be used to deem patterns untestable. The copula support fulfils a monotonicity criterion similar to the Apriori property, enabling pruning of the search space.

Integration of the C-Tarone approach with molecular networks is a highly promising direction for further research. It resolves the problems associated to the binarisation of data, and opens the door to many more fields of applications. For example, an integration of networks with other continuous data types, such as gene expression data, could contribute to the further understanding of complex traits.

Part III.

Network-based identification of cancer driver genes

Context and overview

This third part of this thesis is devoted to the discovery of novel cancer driver genes, i.e. genes that are implicated in the development and progression of cancers. The previous chapters aimed at developing significant pattern mining approaches for network-guided genetic association studies, with the main goal to discover interactions between genes and their sub-units that are associated to a binary phenotype of interest under a model of genetic heterogeneity. While this was addressed with the tools of statistical association testing, in this chapter we change the realm of methods from statistical testing to supervised machine learning with biological networks. The projects are related in that they both leverage the vast information that is present in biological networks, but the way this information is used differs. While in the previously-discussed projects, the network was used to create a hypothesis space, now the network will be used to generate features that help the prediction of cancer driver genes. Furthermore, both projects combine the network information with genetic data. While the approaches presented so far assumed the availability of genetic data in the form of single nucleotide polymorphisms or rare variants for each sample, in this project we will be working on summary statistics of genes that are derived from mutational profiles of tumour cells. That is, while in previous projects, each gene could be represented by an $n \times n_g$ dimensional vector, where n corresponds to the number of samples, and n_g corresponded to either 1 for the method presented in Chapter 4 or to the number of SNPs that mapped to the gene for the approach presented in Chapter 5, in this project, each gene in the network will be represented by a scalar value. Superimposing those scalars onto the network opens the door for network-guided feature extraction. By leveraging the information on well-established cancer driver genes, we can create a supervised approach for the prediction of cancer driver genes.

Since the goal of this project is the prediction of novel cancer driver genes, we start with an introduction to cancer biology, and introduce the data sources used in this project. We continue to present well-established methods for the discovery of cancer driver genes, before delving into our contribution, which will be presented in Chapter 8. We end this third part of this thesis in Chapter 9 with a discussion of our method, as well as an outlook into promising next steps.

7. Introduction to computational cancer genetics

Cancer (also known as malignant tumours or neoplasms [213]) is an umbrella term for a collection of diseases that can affect any part of the human body. It is a disease of unchecked cellular growth, i.e. abnormal cells start to divide and grow beyond their boundaries. Often those cells start to invade other parts of the body and spread to different organs, a process called metastasis. The generation of such metastases contributes largely to the mortality of cancer-related diseases [213]. From a global health perspective, cancer-related diseases constitute an important problem. Alongside ischaemic heart diseases, stroke, and chronic obstructive pulmonary diseases, cancer is among the leading causes of deaths worldwide [214, 215], with an estimated 9.6 million deaths in 2018 [213].

7.1. Cancer as a result of genetic aberrations

Cancer is a genetic disease. This entails that it is caused by aberrations of genes that control normal cell functions such as cell growth, cell division and programmed cell death (called apoptosis) [216]. Aberrations in specific genes allow cancerous cells to ignore signals that control those functions in healthy cells, such that cancer cells continue to grow, divide and proliferate. Genes that are, upon alteration, linked to the development and progression of cancers are referred to as *cancer driver genes*, and previously-identified cancer driver genes can be broadly categorised into **oncogenes** and **tumour suppressor genes**. Oncogenes are altered forms of genes (so-called proto-oncogenes) that are involved in cell proliferation and apoptosis, or both [217]. Proto-oncogenes can be activated, thus turning into oncogenes, through various structural alterations, such as mutation, gene fusion, juxtaposition to enhancer elements, or amplification [217]. Upon activation, those genes may exhibit their cancerous potential by promoting uncontrollable cell growth, referred to as a *gain of function* [218]. Prominent examples of oncogenes are the KRAS, HER-2 and EGFR genes. Tumour suppressor genes on the other hand are genes that regulate diverse cellular activities, including cell cycle checkpoint responses, detection and repair of DNA damage, protein ubiquitination and degradation, mitogenic signalling, cell specification, differentiation and migration, and tumour angiogenesis [219]. Hence, tumour suppressor genes have a protective function by suppressing tumourigenicity, and become cancer driver genes through a *loss of function*. Prominent examples of well-known tumour suppressor genes are TP53, PTEN, BRCA1, and BRCA2. As opposed to oncogenes, tumour suppressor genes are commonly recessive, as a single copy of a functional gene is sufficient to maintain the protective effect [219]. In the literature, this is often referred to as the ‘two-hit’ inactivation of tumour suppressor genes [220], as both copies of the gene have to be affected by an aberration.

7. Introduction to computational cancer genetics

There is a multitude of factors that contribute to the genetic modifications incurring cancer risk. While a genetic predisposition to certain tumour types can be inherited from the parents in the form of germ-line mutations [221, 222], tumours typically arise due to somatic mutations that are acquired during an individual's lifetime, for example due to contact with carcinogenic agents, due to environmental effects and lifestyle choices [223, 224], as well as ageing [225–227].

According to the World Health Organisation (WHO), carcinogenic agents can be classified into three different categories, that is

- i. physical carcinogens, including ultraviolet and ionising radiation
- ii. chemical carcinogens, including asbestos, components of tobacco smoke, aflatoxin and arsenic
- iii. biological carcinogens, including infections from certain viruses, bacteria or parasites,

and exposure to those have been linked to cancer [list adapted from 213]. Another factor contributing to cancer risk and development is attributed to the lifestyle of individuals. Anand et al. [223] ranked diet, physical activity, stress, and environmental pollutants as important criteria that contribute to an individual's risk to develop cancer. Lastly, it has been observed that cancer incidence increases with age, and cancer is the top one cause of death for individuals between 60 and 79 years in the US [215], independent of gender. The mechanisms of cancer development and ageing both underlie the accumulation of cellular damage over time [225], and many of the hallmarks of ageing are shared with cancers [226]. Especially the increasing risk of cancer with increasing age [215] incurs a serious socio-economic challenge, as the advances in medicine, healthcare and technology lead to progressively longer life expectancies. Hence, the importance of cancer research cannot be overstated: better understanding the causes underlying the disease facilitates novel mechanistic, diagnostic and therapeutic insights that are urgently needed to help large numbers of patients.

7.1.1. Challenges in identifying cancer driver genes

The identification of genes involved in cancer requires the molecular characterisation of tumour profiles and matched normal samples to identify differences between the two, while correcting for differences specific to the patient. Collaborative projects such as The Cancer Genome Atlas (TCGA) [228] set out to sequence thousands of tumour exomes and to make them available to the scientific community, which sparked the development of a plethora of computational tools for the analysis of the genetic sequences [229]. Initial analyses revealed mutations in hundreds of genes in certain adult tumours, as well as the presence of gene copy number variants, translocations of sequence, and gene fusions [230]. The predominant mutation type as well as the mutation load varies across tumour subtypes, and between samples within the same tumour type [231]. One important observation is the so-called *long-tail* phenomenon, i.e. the distribution of somatic mutations typically includes only few genes that are mutated at frequencies larger than 10.0%, and most genes are altered at frequencies of 5.0% or lower [232]. Many of the putative cancer drivers genes identified from early analyses were shown to be false positive findings [230], and this large

7. Introduction to computational cancer genetics

number of false positives is partially attributed to this excessive background mutation in cancers, such that one important challenge becomes to distinguish true cancer-causing mutations from so-called **passenger mutations**, that randomly perturb genes [233]. Detecting infrequently mutated cancer driver genes requires large sample sizes and an accurate estimation of the background mutation rates, which is a highly non-trivial task [231]. Currently existing computational tools for the identification of cancer driver genes from somatic mutations or copy number variations [230, 233] have identified cancer drivers that exhibit high mutation rates. Developing computational methods to identify cancer driver genes that are mutated at intermediate and low frequencies is an active field of research, and methods range from the assessment of functional impact of variations [234, 235] to network-inspired methods [32, 143–146].

A central assumption of mathematical modelling and machine learning is that increasing the sample size increases the predictive power of the method to describe the data. The same assumption was made in the context of cancer gene identification: increasing sample sizes should lead to an increased power to detect true cancer driver genes as well as improve the estimation of the background mutation in tumour profiles. In practice, the opposite was observed: by increasing sample sizes, increasing numbers of significant, but implausible, cancer driver genes were reported [230]. While some of the reported genes did indeed correspond to known cancer drivers, they were often accompanied by genes whose involvement in cancer was questionable due to their functional impact or genomic properties. Lawrence et al. [230] attributed the occurrence of false-positive findings to the heterogeneity in the mutational processes of cancer, and the lack of methods to account for this heterogeneity. They show the effect of three different types of mutational heterogeneity on different scales, that is (i) heterogeneity across patients with the same cancer type, (ii) heterogeneity within the mutational spectrum of tumours and (iii) the regional heterogeneity across the genome. Statistical and mathematical models that are over-simplistic and do not take the mutational heterogeneities in cancer into account tend to discover large numbers of false-positive associations. This highlights the urgent need for sophisticated models that are capable of identifying cancer driver genes in the presence of the complex mutational patterns typically observed in cancer.

7.2. Data availability

The importance of cancer research from a global health perspective cannot be overstated. The advances in medicine, health care, and technology lead to ever-increasing lifespans of individuals, and hence an increasing occurrence of cancer diseases in the human population. Improving our understanding of the genetics underlying the development of the disease is hence key to enable prevention, diagnosis, treatment and rehabilitation of patients. In line with this challenge, multi-national landmark sequencing projects were founded with the self-proclaimed goal to identify all human cancer genes, thereby contributing to our understanding of cancer on a molecular level. In this chapter, we will be focussing on two such projects, the first one being *The Cancer Genome Atlas* [TCGA, 228] and the second one being the *Cancer Gene Census* [CGC, 236]. We briefly describe those programs in the following.

TCGA – The Cancer Genome Atlas

The Cancer Genome Atlas [228] is a project that aims at gaining a comprehensive understanding of the genomic alterations that underlie all major cancers by characterising them on a molecular basis. It is a joint effort between the National Cancer Institute and the National Human Genome Research Institute that started in 2006, and brings together researchers from different disciplines and institutes. To date, the TCGA contains molecular profiles of more than 20'000 tumours and the matched samples from more than 11'000 patients, covering 33 cancer subtypes. Different molecular data sets were collected, including clinical information, DNA sequencing (e.g. whole genome/exome sequencing, SNP arrays), copy number variation (e.g. SNP microarrays, copy number microarrays), imaging data (e.g. tissue images, radiological images), methylation data, miRNA via miRNA sequencing and protein expression, amongst others.

The TCGA makes their data publicly available for the research community, and the collected data have already lead to improvements in diagnosis, treatment and prevention of cancer diseases, either by mining efforts through TCGA researchers, or through independent researches that used the data provided by TCGA [237–240].

CGC – The Cancer Gene Census

The Cancer Gene Census [236] aims at creating a comprehensive catalogue of genes that have been implicated in cancers, the so called *Catalogue of Somatic Mutations in Cancer* (COSMIC). The COSMIC data base contains over 700 genes, including functional and mechanistic annotations describing their role in cancer development and progression in terms of key cancer hallmarks¹ and the impact of mutations on the gene and protein function. Genes in the COSMIC data base are functionally associated with the hallmarks of cancer, and are in most cases characterised by somatic or germ-line mutations in their coding regions, thereby changing the functional mechanisms of the gene products. The COSMIC data base contains genes that were curated from literature, and can be categorised into two tiers, with Tier 1 describing well-established cancer genes, and Tier 2 describing newly emerging cancer genes. A gene is classified as Tier 1, if it fulfils two requirements: (i) its role in cancer must be documented and reproducible, and (ii) evidence must exist that mutations of the gene change the activity of the gene product in a way that promotes tumourigenesis. Prominent examples of genes in Tier 1 include the BRCA1 and BRCA2 genes, PTEN and PIK3CA, as well as TP53 and KRAS. If a gene shows a strong indication of a role in cancer, but there is less strong mechanistic or functional evidence available, a gene is classified as a Tier 2 gene. This is mainly the case for two types of genes: genes with mutational patterns that are commonly found in oncogenes or tumour suppressor genes, but the functional evidence is less well established in the scientific literature, or genes for which functional evidence exists, but that exhibit unclear mutation patterns [236]. Genes contained in Tier 2 include SOX21 that presumably is involved in multiple types of myeloma, A1CF, and BMP5, both involved in melanoma.

¹Six biological capabilities acquired during the multi-step development of human tumours. As normal cells progress to tumour cells, they successively acquire those hallmark capabilities [218, 241]. The hallmarks are: sustaining proliferative signalling, evading growth suppressors, activating invasion and metastasis, enabling replicative immortality, inducing angiogenesis and resisting cell death.

At the time of writing this thesis, the COSMIC data base contained 723 genes, with 576 cancer driver genes in Tier 1, and 147 genes in Tier 2.

7.3. Detection of cancer driver genes from somatic mutations

Due to its prevalence and morbidity in the human population, the study of cancer has become one of the big topics in biomedical research. As we have seen in the last section, community efforts evolved to share data and knowledge to improve our understanding of the molecular causes of cancer. In line with the data collection efforts, many computational models and methods targeted the discovery of genes that are involved in the development, progression and treatment of cancer. Here, we focus on those methods that infer the cancer driver status from somatic mutations. Following Cheng et al. [242], we classify methods explicitly designed to fulfil this task into five different categories based on their major features:

Mutational frequency: The idea underlying the frequency-based approaches is to derive a score or p -value for each gene that quantifies the extent to which the mutational frequency in the gene exceeds the background mutation (or passenger-mutations). The models differ in how the background mutations are estimated, and in the types of mutational patterns that are analysed. Methods falling into this category include MuSIC [243], MutSig [233], and OncoDriveCLUST [244].

Functional impact of mutations: This group of methods includes tools such as RAE [235], OncoDrive [234], SIFT [245], and PolyPhen-2 [246]. Those methods assess the functional impact of mutations by estimating the deleterious effects of mutations. This is often done by evaluating amino acid conservation at the specific positions. Many methods that evaluate the functional impact are machine learning based, and those methods are challenged by the lack of both, a gold-standard positive set, as well as a high-quality negative data set to learn from.

Structural genomics: Another set of methods evolves around structural genomics, with the goal to understand the impact of somatic mutations on the level of protein structures. The functional features that lie at the core of such methods include specific protein regions, post-translational modification sites, or protein pockets. Results from studies that analyse the functional impact of mutations on the 3D structure of proteins constitute promising drug targets, and presumably find many applications in clinical settings. One major impairment of those approaches is the limited number of proteins for which high-resolution 3D structures are available (at the time of writing this thesis, the protein data base [247] contains a total of 2'236 human protein structures). Methods included in this group are ActiveDriver [248], iPAC [249], and Protein-Pocket [250].

Data integration: The combination of multiple data sources, such as somatic mutations, transcriptome, methylation and proteomics profiles of tumour and matched normal samples allows researchers to systematically analyse the state of potential cancer genes from a holistic perspective, thereby improving the detection of previously

7. Introduction to computational cancer genetics

unknown cancer driver genes. This is the rationale underlying e.g. the HELIOS method [251]. Tamborero et al. [252] suggested to combine multiple complementary methods to discover cancer driver genes, thereby leveraging tumourigenic potential of genes manifesting in different data types, and outperforming each individual method.

Pathways and networks: Another branch of methods to describe the molecular basis of cancers leverages the biological information present in molecular networks. These approaches are corroborated by the observation that cancers appear to be complex diseases that are caused by aberrations in multiple genes, as opposed to monogenic diseases [242]. As discussed earlier, high mutation rates in tumour samples complicate the differentiation between genes that only carry passenger mutations and rarely mutated cancer genes. One potential explanation of this is that genes interact in complex pathways and protein complexes, and the cancerous potential of a cell is caused by a disruption of the pathway, rather than *one* specific gene within the pathway [241]. This sparked the development of various methods that focussed on a network-based interpretation of cancer genomes. Prominent methods in this group are the series of network-propagation based HotNet methods [143–146], and a method based on local neighbourhoods in molecular networks called NetSig [32].

While this list is by no means a full review of all available methods for the identification of cancer driver genes, it constitutes an overview over the most commonly applied concepts in the field. For a more thorough review, we refer the reader to Cheng et al. [242] and Tokheim et al. [253].

8. Prediction of cancer driver genes through network based moment propagation of mutation scores

Gumpinger, A.C., Lage, K., Horn, H., Borgwardt, K.M., *Prediction of cancer driver genes through network based moment propagation of mutation scores*. In press at *OUP Bioinformatics* (2020).

Gaining a comprehensive understanding of the mechanisms that drive the development and progression of human cancers is an important target in biomedical research. As it is among the leading causes of death [215], combating cancer is a major goal from a global health perspective. While there exist many factors that influence the development of cancer, such as contact with carcinogenic substances, diet, and ageing, cancer is known to be a genetic disease. This entails that all those factors result in genetic aberrations that confer a selective growth advantage to affected cells, and eventually promote the step-wise transition of healthy cells into malignant tumour cells. Many years of research have shown that there exist mainly two classes of genes and mechanisms that are responsible for this transition: the gain of function of **oncogenes**, and the loss of function of **tumour suppressor genes**. While there also exists a hereditary component to tumour development in the form of germ-line mutations, the majority of tumours is driven by somatic mutations acquired during an individual's lifetime. Identifying genes that are causally implicated in various cancers promises key mechanistic, diagnostic and therapeutic insights, and is one of the major goals of cancer research.

Large collaborative projects, such as *The Cancer Genome Atlas* (TCGA) or the *Cancer Gene Census* (CGC) emerged that aim at collecting various types of molecular characteristics of tumour and matched normal samples. Those collections comprise data obtained from various experiments, spanning whole genome or exome sequencing, DNA methylation, and imaging, among others. The resulting data sets are made available to the research community to enhance the detection of novel cancer causing genes, henceforth referred to as *cancer driver genes*. The availability of high-quality data sets sparked the development of a plethora of computational tools for the discovery of novel cancer drivers, from various data types. One major mode of analysis is the statistical assessment of mutational burden of genes. i.e. whether a gene exceeds the mutational burden compared to the background distribution of somatic mutations. However, those analyses are complicated by the extensive mutational heterogeneity: commonly hundreds of genes are mutated in a small number of samples, while only very few genes are mutated across many samples [254]. The so-called *long-tail* phenomenon poses a challenge to identify rarely-mutated cancer genes from passenger mutations that do not contribute to the cancerous potential of the cells. Hence, methods that evaluate the excess mutation rate in genes can prioritise genes

8. Network-based classification of cancer driver genes

for follow-up studies, but fail to reliably identify driver genes that exhibit low mutation frequencies.

Vogelstein et al. [254] suggest to adopt a different perspective of the cancer genome, one that is based on pathways. They noted that all cancer driver genes identified share one property, that is the fact that they directly or indirectly confer a growth advantage, and that the number of signalling pathways through which such a growth advantage can be incurred, is limited. Those pathways can be broadly grouped into three major cellular processes, that is (i) cell survival, (ii) cell fate, and (iii) genome maintenance. This introduced the idea that several *different* genes can lead to the same selective growth advantage, by disrupting the same pathway [254], and inspired network-guided approaches for the identification of cancer driver genes.

The concept underlying most of the network-guided approaches is the computation of a score for each gene that measures the degree of implication in cancer, e.g. based on mutational frequency patterns [233, 243, 244], or functional impact of mutations [234, 235, 245, 246]. Those scores are then superimposed on nodes in a network representing gene-gene interactions, such as protein-protein interaction networks. Well-established methods use those vertex-weighted networks in an unsupervised network propagation setting to identify genes implicated in cancer [reviewed in 65, 143–146]. Other methods, such as **NetSig** [32], exploit the information contained in the direct neighbourhood of genes by aggregating scores in a node’s neighbourhood and computing a representation inspired by meta- p -values, followed by a permutation procedure to evaluate the significance of the node representations.

While those methods have led to the identification of novel genes implicated in cancer, they all approach the problem from an unsupervised perspective, disregarding one important layer of information: the existence of well-established cancer genes [236]. This idle pool of information is typically only used during post-processing of the results to validate the findings of new methods, but rarely integrated into the model itself. To the best of our knowledge, only very few existing methods use this additional data for the prediction of cancer driver genes, such as Bayesian modelling [251] and unsupervised network propagation [148].

In this project we set out to *learn from what we already know*, by formulating the problem of predicting new cancer driver genes as a supervised classification problem. In addition to the well-established cancer driver genes, we also leverage biological prior knowledge describing the interactions between genes in the form of protein-protein interaction networks. With this, we adopt a holistic view of the cancer genome, that does not analyse genes in isolation, but acknowledges processes within the cell that might confer the cancerous potential of the cell. We propose a novel approach for the supervised classification of cancer driver genes, leveraging the set of well-established cancer genes from the CGC in the COSMIC data base [236]. For this purpose, we formulate the problem of cancer gene prediction as a node-classification task in a protein-protein interaction network. The core of our contribution is a novel embedding of nodes in the network, which we named *moment propagation embeddings*. The embedding is based on the distributions of node-features in k -hop neighbourhoods, combined with network propagation. Those embeddings can then be subjected to state-of-the-art machine learning classification algorithms to achieve the

supervised prediction of each gene.

One major challenge induced by the definition of the cancer genes is that the positive class only contains few hundred genes, and that there does not exist a high-quality negative class. Any gene that is not classified as a cancer driver gene might potentially be a yet-to-discover cancer gene. To address this, we develop a sophisticated cross-validation scheme that takes both types of biases into account, and allows for a robust prediction of cancer driver genes. This chapter is organised as follows: it starts with an introduction of the notation, followed by the problem statement. We continue to introduce our contribution, that is the moment propagation embeddings. The major part of this chapter is devoted to the application of the embeddings to a pan-cancer analysis of TCGA data, and the quantitative and qualitative analysis of the results. We end this section with a discussion of our results and an outlook into future work.

8.1. Network-based moment propagation of mutation scores

8.1.1. Notation and problem statement

As in Chapters 4 and 5, we consider a molecular network, such as a protein-protein interaction network, that describes m interactions between d genes. Each node in the network corresponds to a gene, and each interaction between two genes is represented by an edge. Mathematically, an interaction network can be represented as a graph $\mathcal{G} = (V, E, \omega)$, where V corresponds to the set of nodes, E corresponds to the set of edges, i.e. $E = \{(u, v) \mid u, v \in V\}$, and $\omega : V \times V \rightarrow [0, 1]$ is a weighting function that assigns a weight in the range between 0 and 1 to each pair of nodes. The weighting function takes on a value of 0 if there is no edge between two nodes, and a value greater than 0 if an edge is present, i.e. $\omega(u, v) > 0$ iff $(u, v) \in E$. In protein-protein interaction networks, edge weights can for example indicate the confidence of an interaction: edges that have been observed experimentally might have higher confidence than edges that were predicted with machine learning tools. In the case of networks without edge weights, ω is a binary function that indicates the presence of an edge, i.e. $\omega(u, v) = 1$ iff $(u, v) \in E$. In addition to the network we assume that each gene has a g -dimensional feature vector assigned to it that describes g properties of each gene. We denote this feature vector as x_v for every vertex $v \in V$, or as $\mathbf{X} \in \mathbb{R}^{d \times g}$ in matrix notation. We furthermore assume that the set of nodes is partially labelled, i.e. there exists a subset of nodes $V_l \subset V$ that are assigned a class label l . We denote the label assignment of those nodes $v \in V_l$ as $y_v = l$.

Problem statement

Let us assume that we are given the partially labelled data set consisting of the network $\mathcal{G} = (V, E, \omega)$, the vertex feature matrix \mathbf{X} , as well as the class assignment $y_v = l$ for vertices $v \in V_l$ described above. Our goal is to develop a node embedding $\gamma_{\mathcal{G}}(\cdot)$ based on the feature representation \mathbf{X} and the network \mathcal{G} for each vertex $v \in V$ that, used in conjunction with a binary classifier C , predicts the probability with which each *unlabelled*

8. Network-based classification of cancer driver genes

vertex $u \in V \setminus V_l$ belongs to the class l , i.e.

$$C(\gamma_{\mathcal{G}}(x_u)) = \mathbb{P}(y_u = l). \quad (8.1)$$

So in essence, our goal is the creation of node embeddings that serve as input for the supervised, binary classification task of identifying cancer driver genes. To achieve this, we integrate protein-protein interaction networks with vertex scores x_v , that indicate the marginal association of the corresponding gene with cancer. Those scores are superimposed on the nodes in the network, and we create node embeddings from the vertex-, and potentially edge weighted network.

8.1.2. Generation of node embeddings for the prediction of cancer driver genes

The node embeddings proposed in this chapter consist of two different embeddings that are combined. The first one is based on the representation of each node in the network by its local neighbours. To be precise, we describe each node by the distribution across its neighbours' feature vectors. This is inspired by the success of the **NetSig** method [32] that identified novel cancer drivers based on the 1-hop neighbourhood of each gene in a protein-protein network. In our approach, we extend this idea in three different ways: (i) we do not restrict ourselves to 1-hop neighbourhoods, (ii) we allow for the incorporation of edge weights into the model, and (iii) while **NetSig** applies a sum aggregation across neighbourhood scores, we condense the distribution of neighbour-features by computing the *moments* of the distributions. This last step is essential to our approach, and brings two main benefits: first of all, it is computationally efficient, and second, it addresses the problem of knowledge contamination in networks, as moments of a distribution are theoretically independent of the sample size.

The second part of our node-embeddings is a Weisfeiler-Lehmann [187] like aggregation of features in a node's local neighbourhood. This concept has been widely exploited in different tasks, such as network propagation approaches [65] for the detection of cancer driver genes [143–146]. However, it is also central to a emerging branch of deep learning, that is graph convolutional networks [188, 255]. There exists a variety of how local neighbourhoods can be aggregated, and this question still is a topic of active research.

Before we give a detailed description of our node embeddings, we start by formalising the concept of a neighbourhood in a graph \mathcal{G} : we define a k -hop neighbourhood of a vertex v as the subset of all genes that can be reached from v with *exactly* k edges¹. More formally, we define a k -hop neighbourhood recursively as

$$\mathcal{N}_v^k = \left\{ u \mid (w, u) \in E \ \forall w \in \mathcal{N}_v^{k-1}, u \notin \mathcal{N}_v^l \ \forall l \in \{0, \dots, k-1\} \right\}, \quad (8.2)$$

where $\mathcal{N}_v^0 = v$. In other words, if a gene has already appeared in any l -hop neighbourhood, where $l \in \mathbb{N}$ and $l < k$, it cannot be part of the k -hop neighbourhood. Hence, we are defining neighbourhoods *incrementally*.

¹Note that this definition is different from the one introduced in Chapter 4.

8. Network-based classification of cancer driver genes

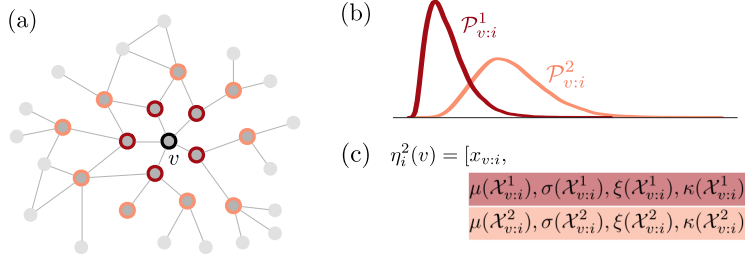


FIGURE 8.1.: Schematic representation of moment embeddings. (a) Illustration of an anchor vertex v , and its 1-hop (dark red) and 2-hop (light red) neighbourhood. (b) The distributions of the i^{th} feature in the 1-hop and 2-hop neighbourhoods. (c) Moment propagation of the i^{th} feature. (This figure is adapted from Figure 1 in Gumpinger et al. [256]).

Remark 8.1.1 (Incremental neighbourhoods). *We would like to point out that the definition of a k -hop neighbourhood in Equation 8.2 is different from the one used previously in this thesis (see Equation 4.1).*

8.1.2.1. Embedding genes using moments of local neighbourhood distributions

We start by defining the moment embeddings for every node $v \in V$. Each node $v \in V$ in the network has a g dimensional feature vector x_v assigned to it. We denote the i^{th} entry in this feature vector as $x_{v:i}$, and compute the moment embeddings with respect to each feature separately. The resulting moments for each feature are eventually stacked together. First, we assume that there exists a distribution $\mathcal{P}_{v:i}^k$ that generates the observed values of feature i in the k -hop neighbourhood of vertex v . That is, there exists a random variable $X_{v:i}^k$ that follows the distribution $\mathcal{P}_{v:i}^k$, i.e. $X_{v:i}^k \sim \mathcal{P}_{v:i}^k$, and the observed values of feature i in node v 's k -hop neighbourhood $x_{u:i}$, where $u \in \mathcal{N}_v^k$, correspond to realisations of this random variable. This is illustrated schematically in Figures 8.1a, and 8.1 b.

We create the moment embeddings as concise descriptions of the distributions $\mathcal{P}_{v:i}^k$ for $i = 1, \dots, g$, and hyperparameter $k \in \mathbb{N}_+$. To achieve this, we initially define a function $\bar{\nu}(X)$ that maps a scalar random variable $X \sim \mathcal{P}$ to its first four moments.

$$\bar{\nu}(X) = [\mathbb{E}_{\mathcal{P}}[X], \mathbb{E}_{\mathcal{P}}[X^2], \mathbb{E}_{\mathcal{P}}[X^3], \mathbb{E}_{\mathcal{P}}[X^4]], \quad (8.3)$$

where $\mathbb{E}_{\mathcal{P}}$ denotes the expectation under the distribution of random variable X . In practice, we replace the expectations with the sample mean $\mu(\cdot)$, the variance $\sigma(\cdot)$, the skewness $\xi(\cdot)$ and the kurtosis $\kappa(\cdot)$. This results in the function

$$\nu(\mathbf{x}) = [\mu(\mathbf{x}), \sigma(\mathbf{x}), \xi(\mathbf{x}), \kappa(\mathbf{x})], \quad (8.4)$$

where \mathbf{x} denotes a set of realisations of the random variable X . We call the function $\nu(\cdot)$ a *moment embedding function*. For a given vertex $v \in V$, we denote with $\mathcal{X}_{v:i}^k$ the values

8. Network-based classification of cancer driver genes

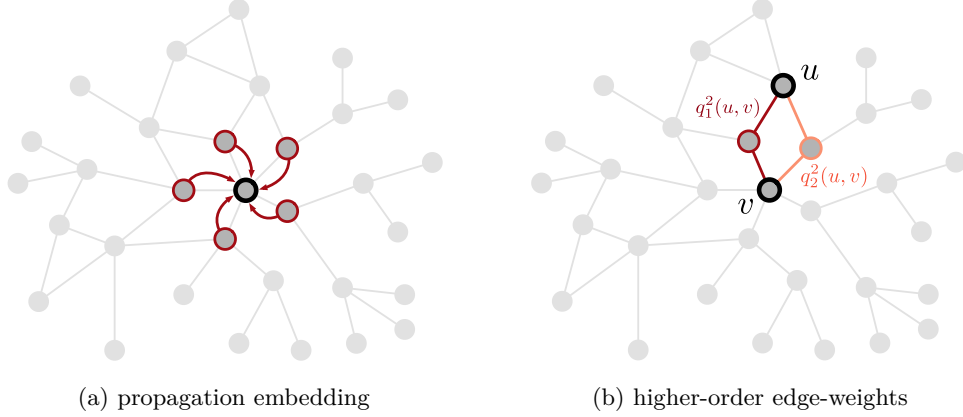


FIGURE 8.2.: (a) Schematic representation of the propagation embedding: an anchor node, highlighted in black, is updated based on its 1-hop neighbours. (b) Computation of higher-order edge weights $\omega^2(u, v)$ between the nodes u and v . There are two paths of length 2 between them, highlighted in dark red ($q_1^2(u, v)$) and in light red ($q_2^2(u, v)$). To obtain the weights of the paths, the edge weights pertaining to each path are multiplied, giving rise to $\omega_1^2(u, v)$ and $\omega_2^2(u, v)$, and the final weight is computed by combining those two weights. (This figure is adapted from Figure 2 in Gumpinger et al. [256]).

of the i^{th} feature of vertices u in the k -hop neighbourhood of v , i.e.

$$\mathcal{X}_{v:i}^k = \left\{ x_{u:i} \mid u \in \mathcal{N}_v^k \right\}. \quad (8.5)$$

Those values constitute draws from the distribution $\mathcal{P}_{v:i}^k$, and we define the moment embedding of vertex v with respect to feature i by applying the function $\nu(\cdot)$ to its 1-hop through k -hop neighbourhoods, that is

$$\eta_i^k(v) = \left[x_{v:i}, \nu(\mathcal{X}_{v:i}^1), \dots, \nu(\mathcal{X}_{v:i}^k) \right]. \quad (8.6)$$

This embedding function creates an $(1 + 4k)$ -dimensional representation of each vertex/gene in the network, that is $\eta_i^k : V \rightarrow \mathbb{R}^{1+4k}$, and is illustrated by means of an example in Figure 8.1c. It depends on one hyperparameter, that is the value of k indicating the order of the neighbourhoods that should be considered. By applying it to all g features separately, and successively stacking the resulting representations, we obtain the final moment embedding function as

$$\eta^k(v) = \left[\eta_1^k(v), \dots, \eta_d^k(v) \right]^T. \quad (8.7)$$

This results in a $g(1 + 4k)$ -dimensional representation for each gene in the network, i.e. $\eta^k(v) : V \rightarrow \mathbb{R}^{g(1+4k)}$, which constitutes the first building block of our node embeddings.

8.1.2.2. Embeddings using network propagation

The second part of our proposed node embeddings rely on a Weisfeiler-Lehman like aggregation of vertex scores in the local neighbourhood [187], which we refer to as a *propagation embedding*. Those propagation embeddings are reminiscent of approaches commonly used in network propagation [65], hence the name. In this type of embedding, we assume again that each node $v \in V$ is represented by its g -dimensional feature vector x_v . The propagation embeddings depend on a hyperparameter t that indicates the number of Weisfeiler-Lehman propagation steps. In each propagation step, the vertex scores x_v are updated simultaneously for all nodes in the network, according to the following equation:

$$x_v^t = \frac{1}{|\mathcal{N}_v^1|} \sum_{u \in \mathcal{N}_v^1} x_u^{t-1}, \quad (8.8)$$

where $x_v^0 = x_v$ corresponds to the initial vertex feature vector. This is illustrated schematically in Figure 8.2a. Hence, the propagation embedding corresponds to an average across all scores at the previous iteration $t-1$ in the 1-hop neighbourhood. Note that the average is taken for each of the g dimensions separately, such that the feature vector x_v^t is again a g -dimensional vector. In order to aggregate multiple such iterations, we stack them across the t iterations, such that the final propagation embedding becomes

$$\rho^t(x_v) = [x_v^0, x_v^1, \dots, x_v^t], \quad (8.9)$$

and $\rho^t : \mathbb{R}^g \rightarrow \mathbb{R}^{(t+1) \times g}$.

8.1.2.3. Combining moment and propagation embeddings to represent genes in a network

In order to arrive at our final node embeddings, we combine the moment embeddings introduced in Section 8.1.2.1, and the propagation embeddings proposed in Section 8.1.2.2, giving rise to the *moment propagation embeddings*, short MoPro embeddings. They correspond to the composition

$$\gamma_{t,k}(v) = (\rho^t \circ \eta^k)(v). \quad (8.10)$$

This function first represents each node in the network by means of the feature-distributions in its leading k -hop neighbourhoods by applying the function η^k . To be precise, the first four moments of those distributions are used, giving a concise and computationally efficient representation of the underlying distributions. Those node representations are then aggregated across local neighbourhoods for each node, by applying the function ρ^t . Hence, the composition function $\gamma_{t,k} : V \rightarrow \mathbb{R}^{(1+t)(1+4k) \times g}$ maps the vertices in the network to a higher-dimensional feature space. The function contains two hyperparameters, k and t , both coming from one of the node embeddings. Those can be tuned during training.

8.1.2.4. Extension to edge weighted networks

In Section 8.1.1, we mentioned the potential existence of a weighting function $\omega : V \times V \rightarrow [0, 1]$, that assigns a weight to each of the edges in the network, and that evaluates to 0 if there does not exist a direct link between two nodes. This edge weight can be incorporated into the MoPro embeddings, thereby distributing *importance* of neighbours in the local neighbourhood. Intuitively, if two nodes $u, v \in V$ are connected by a low-confidence edge, i.e. $\omega(u, v) \ll 1$ we assume that the node u should contribute less in the moment propagation $\eta^k(v)$ of node v , and vice versa. We can achieve this by rescaling the vertex-score of node u with the edge weight. This rescaling is done for each of the g features identically and independently, and the set of realisations from the neighbourhood distribution introduced in Equation 8.5 becomes

$$\mathcal{X}_{v:i}^k = \left\{ f(x_{u:i}, \omega(u, v)) \mid u \in \mathcal{N}_v^k \right\}, \quad (8.11)$$

where $f(\cdot, \cdot)$ is a rescaling function, such as a multiplication, that takes the scalar value of the i^{th} feature of each k -hop neighbour of vertex v , and rescales it with the corresponding edge weight.

Remark 8.1.2 (Choice of the weighting function). *It is important to note that the weighting function used here should respect the nature and directionality of the node features. That is, if the node features correspond to log-transformed p-values, the features can be rescaled by a simple multiplication, as $0 \leq \omega(\cdot, \cdot) \leq 1$. However, in case that node features correspond to p-values, a multiplication with a weight smaller than 1 would result in stronger p-values, which is the opposite of the desired effect.*

For k -hop neighbourhoods with $k \geq 2$, this rescaling becomes more involved. Given a root node v , and its, e.g., 2-hop neighbour u , i.e. $u \in \mathcal{N}_v^2$, it holds that $(v, v) \notin E$, and hence $\omega(u, v) = 0$. In order to still rescale those higher-order neighbours, we have to define the concept of a higher-order weight between two nodes that do not share a direct edge. We denote the corresponding weight as $\omega^k(u, v)$, where $u \in \mathcal{N}_v^k$, and its computation is a multi-step procedure:

- i. Enumeration of all paths of length k between the nodes u and v . We denote the i^{th} such path as $q_i^k(u, v)$. It corresponds to a set of edges that form a part in the path, such that $q_i^k(u, v) \subset E$, and store them in the set $\mathcal{Q}_{u,v}^k$. This is illustrated in Figure 8.2b.
- ii. To compute a weight for each of the paths $q_i^k(u, v)$, we multiply the weights of all edges assigned to them, i.e.

$$\widehat{\omega}_i^k(u, v) = \prod_{(x,y) \in q_i^k(u,v)} \omega(x, y),$$

and store those weights in the set $\mathcal{W}_{(u,v)}^k$.

- iii. To compute the final weight $\omega^k(u, v)$, we use an aggregation function $g : \mathcal{S} \rightarrow \mathbb{R}$ that maps an arbitrary set of scalars \mathcal{S} to a scalar weight. It summarises the weights

8. Network-based classification of cancer driver genes

across all paths in the set $\mathcal{W}_{u,v}^k$, that is

$$\omega_g^k(u, v) = g \left(\left\{ \widehat{\omega}_i^k(u, v) \mid \widehat{\omega}_i^k(u, v) \in \mathcal{W}_{(u,v)}^k \right\} \right).$$

Remark 8.1.3 (Choice of the aggregation function). *There exist various ways to combine multiple weights into the final k -hop weight $\omega^k(u, v)$, $u \in \mathcal{N}_v^k$. Here, we either apply the maximum or the mean function, i.e.*

$$\omega_{\max}^k(u, v) = \max \left(\left\{ \widehat{\omega}_i^k(u, v) \mid \widehat{\omega}_i^k(u, v) \in \mathcal{W}_{(u,v)}^k \right\} \right), \quad \text{or} \quad (8.12)$$

$$\omega_{\text{mean}}^k(u, v) = \text{mean} \left(\left\{ \widehat{\omega}_i^k(u, v) \mid \widehat{\omega}_i^k(u, v) \in \mathcal{W}_{(u,v)}^k \right\} \right). \quad (8.13)$$

The choice of this function is treated as a hyperparameter in our approach, and is hence tuned during training.

8.2. Prediction of cancer driver genes for a TCGA pan-cancer study

After the introduction of our MoPro embeddings in the last section, this section describes the application of the node embeddings to the task of a network-based, supervised classification of cancer driver genes from mutation scores of TCGA tumour profiles. This requires the existence of three different types of data, namely (i) the gene-wise mutation scores, (ii) a high-quality molecular network, and (iii) a set of well-established cancer driver genes. We will first elaborate those data sets, before explaining how they can be combined to fulfil the task of supervised cancer driver gene classification. We will see that a supervised classification requires a sophisticated cross-validation procedure, that takes two types of biases into account, that is (i) the highly imbalanced data set caused by the small number of cancer driver genes, and (ii) the non-existence of a high-quality negative class. We will continue to compare the combination of our proposed cross-validation procedure with the MoPro embeddings against a variety of comparison partners, and evaluate the robustness of our method. This section is concluded with a qualitative analysis of the newly discovered cancer driver genes.

8.2.1. Data set description

As mentioned above, three different types of data are required to be readily available, namely mutation scores of genes, a high-quality network, and a set of well-established cancer driver genes.

For the first type of data, we used MutSig mutation scores [233] that were derived from a TCGA pan-cancer study that comprises 9'423 tumour exomes, spanning 33 different cancer types. We combine this genetic data with the well-established InBio Map protein-protein interaction network [25, 257] to represent interactions between genes. The task of supervised prediction requires the notion of a class label. We derive this type of information from the *Cancer Gene Census* data, i.e. from the *Catalogue of Somatic Mutations In*

8. Network-based classification of cancer driver genes

Cancer, short COSMIC, data base [236]. All three data sources are described in greater detail below.

MutSig— mutation scores

The MutSig scores are p -values that assess the significance of three different aspects of mutational patterns within genes between tumour samples and matched normals: (i) the frequency of mutation (MutSigCV), (ii) the clustering of mutations (MutSigCL), and (iii) the functional impact of mutations (MutSigFN). Each of the three parts result in a p -value themselves, and those p -values are combined to yield the final MutSig p -value. The MutSigCV statistic [230] evaluates the mutational frequency of non-silent mutations within a gene, while estimating the background mutation rate from silent and non-coding mutations within the same gene, and neighbouring genes. MutSigCL and MutSigFN [both introduced in 258] are permutation-based methods, and computed from random permutations of the positions of non-silent mutations in a gene, while preserving the mutational category. For each random permutation, two different statistics are computed, one that assess the positional clustering, and one that assesses the functional impact. The permutations generate the null-distribution, and the p -value is computed as the empirical p -value based on this null-distribution. A joint p -value of MutSigCL and MutSigFN is computed from the joint probability distribution estimated from the permutations. The resulting p -value is then combined with the MutSigCV p -value either by using Fisher’s method, or the truncated product method.

From the TCGA exomes, a total of 18’154 genes could be attributed with their MutSig p -value, and we applied a $-\log_{10}$ transformation to each of them. We refer to those as MutSig scores in the following.

InBio Map — interaction between genes

Our newly proposed MoPro embeddings require a notion of interaction between genes. To represent this, we use a well-established protein-protein interaction network, called InBio Map [25, 257]². This network will constitute our view of interactions between genes on a protein level. The network contains in its raw form 390’424 interactions between 12’369 genes, the average degree is 63.13 (± 136.12). For each interaction in the network a score in the range (0, 1] is provided. It quantifies the confidence of the interaction. Interactions derived from pathway databases are assigned the highest confidence score of 1, while interactions between genes that are, e.g., inferred from orthology have scores < 1 . For a detailed description of the inference of interaction confidence score, we refer to Li et al. [25].

COSMIC — A set of well-established cancer-driver genes

The supervised classification of cancer driver genes requires access to labelled data in order to train a classifier. The CGC initiated the COSMIC data base, a well-established data resource for genes that are implicated in cancer [236], see also Section 7.2. At the time

²We are working on the same version of the network as in [32] to allow for a one-to-one comparison with NetSig.

8. Network-based classification of cancer driver genes

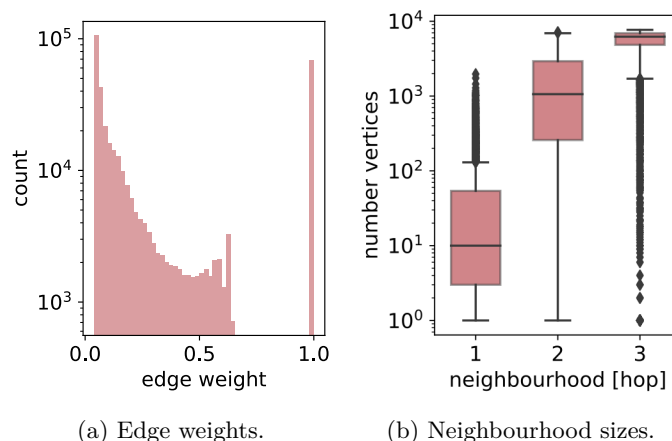


FIGURE 8.3.: Description of the InBio Map network after mapping the MutSig scores. (a) Distribution of edge weights in the network, (b) sizes of the 1-hop, 2-hop and 3-hop neighbourhoods. (This figure is adapted from Figure 2 in Gumpinger et al. [256]).

of publication, the COSMIC data base contained 723 genes that were separated into two tiers, indicating the evidence of the gene’s implication in cancer. Genes in Tier 1 show a documented and reproducible activity relevant to cancer (576 genes), while genes in Tier 2 are considered to be newly emerging cancer genes that possess strong evidence to play a role in cancer (147 genes). For the purpose of our supervised classification, we treat both tiers equally, and assign the cancer driver gene class label to those genes that are either in Tier 1 and Tier 2.

Bringing everything together — The final data set

Our goal is to derive a data set in the form of a graph, where nodes correspond to genes and edges correspond to interactions. Furthermore, each node is equipped with a mutation score and a label that identifies the gene as either a known cancer driver, or an unlabelled gene. For this purpose we integrate the three data sources mentioned above, that is the MutSig scores, the InBio Map network, and the COSMIC genes. We achieve this by first removing all nodes from the network that cannot be represented with a MutSig score, as well as removing all isolated nodes (i.e. those nodes with degree 0). This results in a substantially reduced data set containing 11’449 genes and 349’311 interactions among those. The average degree of the network, i.e. the size of the 1-hop neighbourhoods, is reduced to 61.02 (± 128.33). The sizes of the 1 to 3-hop neighbourhoods are illustrated in Figure 8.3b. Out of the 349’311 remaining edges, 67’894 have a confidence score of 1 assigned to them (see Figure 8.3a for the distribution), and out of the 723 known cancer driver genes, 635 are represented in the network.

Using the CGC genes as set of cancer drivers, we can observe a knowledge bias in the InBio Map protein-protein interaction network: genes that are classified as cancer genes tend to have higher degrees than unlabelled genes (see Figure 8.4a). Furthermore, genes that have low MutSig p -values exhibit a slightly higher correlation with degree (Pearson correlation: 0.17) compared to the set of unlabelled genes (Pearson correlation: 0.10), as illustrated

8. Network-based classification of cancer driver genes

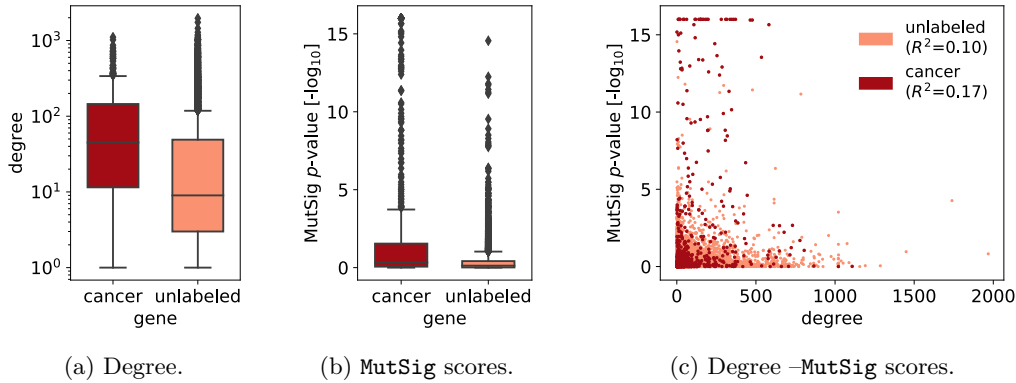


FIGURE 8.4.: Description of the TCGA data set after mapping MutSig p -values to the InBio Map network. (a) degree distributions of nodes classified as cancer drivers (CGC), and unlabelled nodes. (b) MutSig p -values of gene that are classified as cancer drivers (CGC) and unlabelled genes; (c) correlation between degree and MutSig scores, for cancer driver genes (CGC, in dark red), and unlabelled genes (light red). The R^2 value corresponds to the Pearson correlation between the degree and the MutSig scores. (This figure is adapted from Figure 2 in Gumpinger et al. [256]).

in Figure 8.4c. While CGC genes tend to exhibit lower MutSig p -values compared to the unlabelled genes, it is noteworthy that there exist CGC genes with high MutSig p -values, hence those genes could not have been detected with the MutSig approach alone (see Figure 8.4b).

The data set created as described above raises three challenges that have to be addressed in order to enable a supervised prediction of cancer driver genes: (i) there is no well-established set of non-cancer driver genes, i.e. a lack of a high-quality negative class; (ii) the resulting data set is imbalanced, with approximately 5.9% of genes falling into the ‘positive’ cancer driver gene class, and the remaining 94.1% falling into the class of unlabelled genes; (iii) the data set shows some effect of knowledge contamination, i.e. well established cancer driver genes exhibit on average higher degrees, such that the degree could function as a confounder in the prediction.

To address the first and second challenge, we develop a sophisticated cross-validation procedure that takes both aspects, i.e. the class imbalance and the lack of a negative class, into account. This cross-validation procedure will be discussed in the following section. The third challenge, i.e. the knowledge contamination is addressed by the moment component in the proposed MoPro embeddings.

8.2.2. Experimental setup

8.2.2.1. Cross-validation procedure

To address the challenges imposed by the class imbalance and the lack of a negative class, we develop a cross-validation procedure that is based on the repeated *undersampling* of the majority class. This procedure has been shown effective in the absence of a negative class label [259]. A schematic representation of the cross-validation procedure can be found in

8. Network-based classification of cancer driver genes

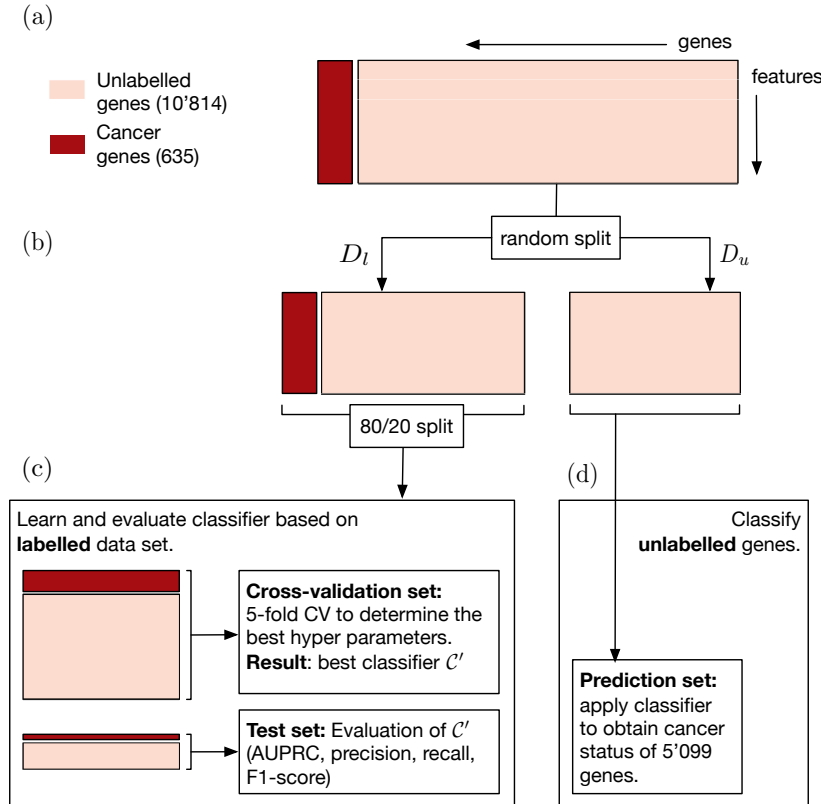


FIGURE 8.5.: Illustration of the cross-validation procedure for one random split of the data set. (a) The data set consists of 11'449 genes, each one is described by a fixed set of features. 635 genes have a cancer gene label assigned to them (dark red), while the remaining 10'814 genes are unlabelled. (b) The data set is randomly split into two sets, a labelled set D_l and an unlabelled set D_u . The 635 cancer genes form part of the labelled data set D_l , together with 5'715 unlabelled genes. For the sake of training a classifier, those genes are assigned a non-cancer gene label. (c) The data set D_l is used to train a classifier in a supervised fashion. For this purpose, we do a stratified split into a cross-validation set containing 80.0% of D_l data, and a 20.0% test split for the evaluation of the classifier. Hyperparameters are selected on the cross-validation set using 5-fold cross-validation, and the best hyperparameters give rise to the best classifier C' , whose performance is evaluated on the hold-out test split. (d) Eventually, C' is used to predict the cancer status of the genes in the unlabelled data set D_u . (This figure is adapted from Figure 3 in Gumpinger et al. [256]).

8. Network-based classification of cancer driver genes

Figure 8.5 for one cycle of undersampling. The starting point is a data matrix containing the 11'449 genes, where each of the genes is represented by a fixed set of features, generated with the MoPro embeddings. That is, we have a data matrix $D^{11449 \times p}$, where p indicates the number of features. Furthermore we know that 635 of those genes are well-established cancer driver genes (see Figure 8.5a). The cross-validation procedure consists of three main steps:

Step 1 — Data splits: We start by splitting the data set D into two disjoint sets, a labelled data set D_l and an unlabelled data set D_u (see Figure 8.5b). The labelled set D_l contains all 635 cancer genes, and a random sample of the 10'814 unlabelled genes. To achieve this, we subsample the unlabelled genes such that the 635 cancer genes make up 10% of the data, and the remaining 90% (corresponding to 5'715 genes) correspond to genes that are not implicated in cancer. Since we are training a binary classifier, we assign the cancer driver genes to the positive, and the remaining genes in D_l to the negative class. We assume this to be the *true label* for the current random split of the data. The data set D_l will be used in the next steps to train and evaluate a classifier, and we use the classifier trained on D_l to predict the cancer status of the genes in the data set D_u .

Step 2 — Training and evaluating the classifier: Next, the data set D_l will be used to train and evaluate a classifier. At this stage, the data set D_l contains samples that are assigned to one of two labels, and the positive class makes up 10% of the samples in D_l . We generate a stratified 80/20 split of the data. We refer to the split containing 80% of samples as the *cross-validation set*. We use it to select hyperparameters of the classifier with a 5-fold cross-validation scheme, and eventually train the classifier using the best hyperparameters on the complete cross-validation set. We denote the resulting best classifier as \mathcal{C}' . The split containing 20% of the samples is referred to as the hold-out *test set*. It is exclusively used to evaluate the performance of the classifier \mathcal{C}' with respect to the area under the precision recall curve (AUPRC), precision, recall and F1-score. Importantly, this hold-out test set is neither seen during training, nor used to select hyperparameters, i.e. the cross-validation and test-set are kept separate at all times, such that there is no information leakage.

Step 3 — Prediction of cancer driver genes: In this last step, we use the classifier \mathcal{C}' obtained in the previous step, and predict the cancer status of the genes in the unlabelled data set D_u (see Figure 8.5d). By design of the cross-validation procedure, those samples have not been seen during the hyperparameter selection process, during training, or during evaluation.

The goal of our approach is to obtain a prediction for each of the 10'814 genes that are part in the network, but are not contained in the set of CGC cancer driver genes. However, with one iteration of the aforementioned procedure, only genes that are in the set D_u obtain a prediction. For this, and another reason that will become apparent in the next paragraph, we repeat the cross-validation procedure multiple times for different random splits of the data, and we denote this number of repetitions as n_r . As a consequence, (i) n_r classifiers will be trained on different splits of the data set, (ii) n_r classifiers will

8. Network-based classification of cancer driver genes

be evaluated on the respective test sets, allowing us to compute the mean and standard deviation of the performance metrics, and (iii) each unlabelled gene will be classified with a subset of those classifiers, corresponding to those iterations for which it was in the D_u set. We require at least five predictions for each gene, which leads to a minimum number of $n_r = 11$ iterations of the cross-validation scheme. In order to ultimately classify a gene as a cancer driver, we take a majority vote across the predictions from the set of classifiers for which the gene was in the prediction set. In the case of ties we resort to the conservative prediction of no cancer gene. Importantly, as there exist no known labels for those genes, we analyse them qualitatively.

With the above described cross-validation procedure, we set out to address two major challenges inherent to the supervised prediction of cancer driver genes, that is (i) the class imbalance, and (ii) the lack of a high quality negative class. The proposed undersampling scheme results in a data set D_l for training and testing of a classifier, in which the class imbalance is less severe than in the original data set, i.e. the positive class has a prevalence of 10%. The n_r repetitions of this random undersampling ensure that the results are not confounded by sampling biases caused by the splitting of the data. The second challenge is addressed by the assignment of the negative class to a random sample of the unlabelled genes for each split (i.e. the unlabelled genes in the D_l set). Hence, in each split, the classifier learns to distinguish cancer genes from this random set of unlabelled genes. This is, however, a possibly incorrect assumption, as there might be yet-to-discover cancer drivers among those genes. By repeating this cross-validation procedure n_r times, we address this incorrect assumption: due to the random splitting of the data, the distribution of the ‘negative’ genes varies from split to split, such that the classifier C'_i trained on the i^{th} split of the data learns the modalities of the negative genes in this *current* split. Intuitively, the classifier focusses on those properties that distinguish cancer genes from unlabelled genes in the current split, and those properties vary from split to split. Since every gene is predicted with every classifier³, it might be classified as a cancer gene by some of the classifiers, but not by others. Hence, the gene might be more similar to a cancer gene in some aspects, and dissimilar in other aspects. The majority vote across multiple classifiers consolidates the predictions across multiple splits.

8.2.2.2. Classification details

Using the data described in Section 8.2.1, we represent each node in the network using the MoPro embeddings on the $-\log_{10}$ -transformed MutSig p -values, referred to as MutSig scores in the following. We apply four state-of-the-art classification algorithms for the prediction of the cancer driver status of the genes, that is **logistic regression**, **random forests**, **support vector machine**, and **gradient boosting**, implemented in the python-based SCIKIT-LEARN library [260]. For all methods we optimise over a grid of standard hyperparameters, as well as the following set of data and method-specific hyperparameters: (i) whether or not to include a scaling step into the classification pipeline, (ii) whether to include the edge weights in the network for the MoPro embeddings (see Section 8.1.2.4) and how to compute the higher-order weights $\omega^k(u, v)$ between nodes u and v , i.e. whether to use the mean or the maximum aggregation (see Section 8.1.2.4),

³Excluding those that were trained on the gene, i.e. whenever the gene was in the D_l set

8. Network-based classification of cancer driver genes

(iii) the number of propagation steps t as well as (iv) the number of k -hop neighbourhoods to explore in the moment embeddings. For the last two parameters, we explore the following ranges: $t \in \{1, \dots, 6\}$, $k \in \{1, 2\}$. We restrict the value of k to a maximum of 2, as the in the InBio Map network, neighbourhoods of size 3 already span large parts of the network (see Figure 8.3b). In case edge weights are used, we rescale the node-features (see function $f(\cdot, \cdot)$ in Equation 8.11) by multiplying the MutSig score with the corresponding edge weights. Since the MutSig scores correspond to $-\log_{10}$ -transformed p -values, multiplying them with values in the range $[0, 1]$ results in a reduction of the significance signal. We would like to note that all the above parameters are treated as hyperparameters that are optimised during training, and the best-performing hyperparameters are chosen on the cross-validation set.

In order to evaluate the predictive performance of the classification we use the area under the precision-recall curve (AUPRC), precision, recall, and the F1-score. Although the area under the receiver operator characteristic curve (AUROC) is often used to evaluate the performance of a binary classifier, it is not an appropriate metric in our setting. This is due to the high class imbalance, and our primary interest in detecting the minority class, i.e. the class of cancer driver genes. As a consequence, we report this metric, but would like to note its difficult interpretation. Since the cross-validation procedure yields a set of classifiers, one for each of the n_r splits of the data, we also obtain performance metrics on the test-sets in each split, and we report the mean and standard deviation across those metrics. We furthermore report the number of novel cancer driver genes, i.e. those genes that were predicted as cancer drivers, but are not contained in the set of CGC genes.

8.2.2.3. Comparison partners

We compare our proposed MoPro embeddings combined with the cross-validation procedure to a variety of baseline methods. They can be broadly categorised into *degree-based* baselines, *MutSig-based* baselines, and *NetSig-based* baselines.

8.2.2.4. Degree-based baselines

The degree-based baselines use only the degree as a feature for each gene, and aim at predicting the cancer status of a gene based on this information alone. Those baselines are supposed to show the knowledge contamination in the network. We use the node-degree in two different settings, which we refer to as **ranking** and **LogReg**. In the ranking setting, genes are ranked based on the degree, and the performance metrics are computed for the ranking. Since there is only one ranking, no standard deviations are reported. The second setting, **LogReg**, applies the proposed cross-validation procedure for a logistic regression classifier. Hence, standard deviations can be reported.

8.2.2.5. MutSig-based baselines

The second class of baseline methods uses the MutSig statistic [233] as the key concept. As for the degree-based baselines, the two settings **ranking** and **LogReg** are used (analogously to the degree-based methods). Additionally, we use a **Benjamini-Hochberg** correction at a false-discovery rate of 10.0% to determine significantly mutated genes, and predict those

8. Network-based classification of cancer driver genes

significant gene as cancer drivers. We compute evaluation metrics for this criterion, and since there is only one resulting classification, there are no standard deviations reported. Additionally, we apply two different methods based on network propagation to identify novel cancer driver genes from the `MutSig` statistic, that is `hierarchical HotNet` [146], as well as `uKIN`⁴ [148]. While the `hierarchical HotNet` method is a fully unsupervised approach, `uKIN` leverages labels of cancer genes during the propagation of the signal through the network. In the case of `hierarchical HotNet`, we consider all genes in the largest reported cluster as predicted cancer driver genes, hence we cannot report AUPRC. In the case of `uKIN`, we implement the same cross-validation procedure as suggested in the original publication, using 10 repetitions [for details on implementation, see supplementary material in 256].

8.2.2.6. NetSig-based baselines

The last class of baselines is based on the `NetSig` statistic [32]. `NetSig` is a computational method for the identification of cancer driver genes based on aggregating the `MutSig` scores of a genes local neighbourhood in a network. As such, it is the most direct comparison partner of our proposed `MoPro` embeddings. The `NetSig` method returns an empirical p -value for each gene in the network, obtained from a permutation procedure of the node-scores that corrects for knowledge contamination. We evaluate the `NetSig` p -values in three different settings, that is `ranking`, `LogReg` and `Benjamini-Hochberg`. The evaluation is identical as in the `MutSig` case.

8.2.3. Results

After establishing all theory and the experimental design, this section is devoted to the quantitative and qualitative analysis of the `MoPro` embeddings combined with our cross-validation scheme. We start with the quantitative analysis by reporting the performance metrics obtained with our proposed approach and the baselines, and continue to evaluate the robustness of our methods for different settings, and address the problem of knowledge contamination. The last part of this section is devoted to the qualitative analyses of the novel cancer driver genes that were discovered with our proposed approach.

8.2.3.1. Cancer gene classification with MoPro embeddings

The classification performances obtained are listed in Table 8.1, where Table 8.2a contains the baseline results, and Table 8.2b contains the results of the classification with `MoPro` embeddings. We evaluate the four classifiers on a range of hyperparameters as described in Section 8.2.2.2, and report the best setting in Table 8.2b. We observe that all four classifiers show similar performances with respect to AUPRC, with the exception of the random forest classifier, which stays approximately three percent points behind logistic regression, support vector machines and gradient boosting. Most importantly, we observe that the combination of `MoPro` embeddings with the proposed cross-validation procedure consistently outperforms all baseline methods. When focussing on AUPRC, the best baseline

⁴At the time of publication, the `uKIN` code was not yet available, such that we implemented the method ourselves.

8. Network-based classification of cancer driver genes

TABLE 8.1.: Results of cancer gene classification for (a) the baselines and (b) the MoPro embeddings. AUROC is the area under the receiver operating characteristic, AUPRC is the area under the precision recall curve. The last column with title *novel* indicates the number of *de novo* cancer genes, i.e. those genes that are not contained in the set of cancer genes. The we highlight the best-performing method with respect to AUPRC and AUROC in red. (This table is adapted from Table 1 in Gumpinger et al. [256]).

(a) Results of baseline methods. The first column indicates the feature that was used to represent each gene during classification, the second column indicates the method that was used for classification. In case of LogReg, we used the cross-validation procedure described in Section 8.2.2.1 and fixed the recall at 23.5%.

feature	method	AUROC	AUPRC	precision	recall	F1	novel
degree	ranking	0.683	0.096	0.105	0.436	0.169	2368
degree	LogReg	0.700 (0.006)	0.199 (0.007)	0.243 (0.012)	0.236 (0.000)	0.239 (0.006)	905
MutSig	ranking	0.643	0.248	0.474	0.202	0.283	142
MutSig	LogReg	0.620 (0.005)	0.312 (0.007)	0.552 (0.060)	0.236 (0.000)	0.330 (0.011)	243
MutSig	BH	0.643	0.248	0.490	0.191	0.274	126
MutSig	Hi. HotNet	-	-	0.137	0.111	0.123	444
MutSig	uKIN	0.732 (0.016)	0.296 (0.037)	0.428 (0.097)	0.233 (0.004)	0.298 (0.027)	187
NetSig	ranking	0.657	0.158	0.219	0.235	0.226	532
NetSig	LogReg	0.674 (0.006)	0.275 (0.012)	0.278 (0.019)	0.228 (0.000)	0.250 (0.008)	704
NetSig	BH	0.657	0.158	0.263	0.169	0.205	300

(b) Classification results for different classifiers using the proposed MoPro embeddings. The columns 2-5 indicate the hyperparameters that gave the best classification performance for each set of classifiers. t and k are the hyperparameters of the MoPro embeddings, namely the number of propagation steps and the neighbourhood degree up to which moments are computed, respectively.

method	scale	path	t	k	AUROC	AUPRC	precision	recall	F1	novel
LogReg	True	-	3	2	0.799 (0.008)	0.434 (0.014)	0.572 (0.046)	0.236 (0.000)	0.334 (0.009)	202
SVM	False	-	6	2	0.793 (0.005)	0.431 (0.012)	0.584 (0.058)	0.236 (0.000)	0.336 (0.010)	198
RandFor	True	mean	3	1	0.781 (0.009)	0.396 (0.021)	0.560 (0.057)	0.234 (0.004)	0.330 (0.011)	193
GradBoost	True	max	3	2	0.796 (0.008)	0.437 (0.020)	0.636 (0.088)	0.236 (0.000)	0.343 (0.012)	150

comparison partner is the logistic regression of the MutSig statistics (AUPRC=31.2%), followed by the uKIN approach (AUPRC=29.6%). The best baseline with respect to AUROC is logistic regression on the node degree (AUROC=70.0%). The fact that predicting only on the degree achieves such a performance hints towards the presence of knowledge contamination in the network, and we devote Section 8.2.3.4 to the discussion of this bias in the light of MoPro embeddings. With the MoPro embeddings, AUPRC values of up to 43.7% can be achieved with gradient boosting, and we observe similar trends for the AUROC values. This constitutes a major improvement compared to the baselines. A general trend that becomes apparent when looking at the results is that incorporating the knowledge about established cancer driver genes results in an improvement of performance. This is the case for the classical supervised machine learning setting that is at the heart of our cross-validation scheme, as well as for the uKIN method, that exploits this information in a semi-supervised way. Furthermore, our results indicate that networks contain information that is helpful when predicting the cancer driver status of genes. We can see this by the performance gain of network-driven methods compared to methods that do not consider

8. Network-based classification of cancer driver genes

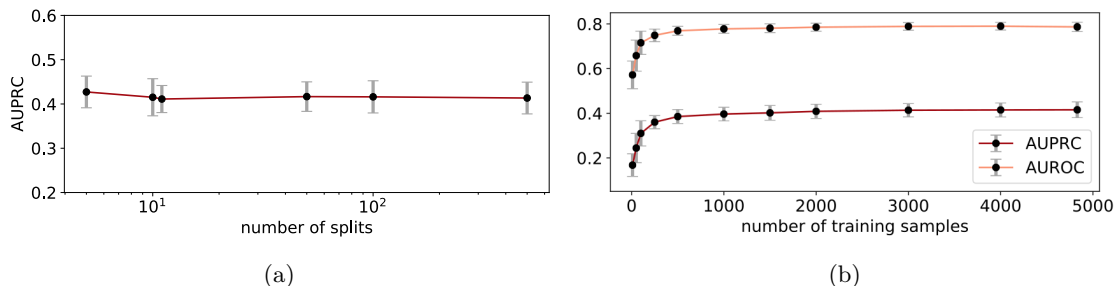


FIGURE 8.6.: Dependence of classification performance on cross-validation hyperparameters. We evaluate a set of logistic regression classifiers with hyperparameters as in Table 8.2b. (a) AUPRC upon varying the number of random data splits n_r in the cross-validation procedure, and therefore also the number of classifiers. (b) Evaluation metrics as a function of the training size. (This figure is adapted from Figure 4 in Gumpinger et al. [256]).

network information, such as logistic regression of the **MutSig** p -values.

For all the analyses in Table 8.1 the classification threshold was chosen such that the recall matched the recall obtained by ranking the **NetSig** scores, i.e. at a value of approximately 23.5%. Compared to the **NetSig** approach, **MoPro** embeddings achieve an up to three-fold improvement in precision at the same recall (**NetSig**: 21.6%, **GradBoost**: 63.6%). The best performing baseline with respect to precision is logistic regression of the **MutSig** scores, and achieves values of 55.2%, i.e. approximately eight percent points lower than what we observe with the **MoPro** embeddings.

The optimisation of data-specific hyperparameters yields that at least three propagation steps enables the best classification. All methods with exception of the random forest classifiers worked best when exploring moments of the 2-hop neighbourhoods. Edge weights do not result in an improved classification performance for logistic regression and support vector machines, but are useful in the tree-based approaches, i.e. random forests and gradient boosting. While random forests achieve better performance when averaging path weights, in the case of gradient boosting it appears beneficial to aggregate path weights using the maximum function (see Equation 8.12), such that we cannot determine a clear pattern of which aggregation is superior.

8.2.3.2. Dependence on cross-validation parameters

Next, we set out to analyse the dependence of the results on two hyperparameters of the cross-validation scheme, that is the number of data splits n_r , as well as the size of the training data set. In the results presented in Table 8.1, all results are obtained from $n_r = 11$ splits of the data into the labelled set D_l and the unlabelled set D_u . We evaluate the performance of logistic regression using the **MoPro** embeddings, and vary the number of data splits n_r in the range from [5, 500]. All other data specific parameters, such as the number of propagation steps t or the neighbourhood degree k remain fixed to the values in Table 8.2b. We evaluate the resulting performance with respect to AUPRC, and visualise the results in Figure 8.6a. We observe that the classification performance is not affected by changes in the parameter n_r . We would like to note that the slight decrease

8. Network-based classification of cancer driver genes

TABLE 8.3.: Results of the ablation study for the set of logistic regression classifiers. In *propagation only*, the node feature is propagated, but no moment embedding is computed. In *moments only*, moments are computed, but no propagation embedding is computed. The first row repeats the baseline results (MoPro embeddings) for comparability reasons. (This table is adapted from Table 2 in Gumpinger et al. [256]).

setting	method	AUROC	AUPRC
MoPro embeddings	LogReg	0.799 (0.008)	0.434 (0.014)
propagation only	LogReg	0.717 (0.006)	0.348 (0.010)
moments only	LogReg	0.772 (0.007)	0.406 (0.011)

in performance of approximately 2% is due to fixing the data hyperparameters.

The results of the second analysis, that is the effect of the training set size on the classification performance, is visualised in Figure 8.6b. In the proposed cross-validation scheme, and the results in Table 8.1, the training set contains 4'064 samples. This number is defined by the 5-fold cross-validation (the cross-validation set contains 5'080 genes, such that during 5-fold cross-validation 4'064 genes are used for training, and 1'016 for validation). In this experiment, we reduced the number of training samples to values in the range [10, 4'000], and evaluate the classification performance with respect to AUROC and AUPRC. We observe a steep increase in the performance metrics for up to 1'000 training samples, and a saturation in performance when using more than a 1'000 samples. Those results indicate that at least 1'000 samples are necessary to represent the data distribution, and successfully classify the genes.

8.2.3.3. Ablation study

The proposed MoPro embeddings are a composition of two node embeddings, that is the moment, and the propagation embeddings. We conduct an ablation study in which we remove one of the two components, and evaluate the classification performance. The goal of this is to understand to what extent the two embeddings contribute to the improved performance. The results of this ablation study can be found in Table 8.3. Upon removing moments, and only propagating the MutSig scores through the network (setting 'propagation only'), we observe a drop in classification performance with respect to AUPRC of approximately eight percent points. When generating node features via moments, but not propagating those through the network (setting 'moments only'), the observed drop in performance is less severe, with only about three percent points. This leads us to the following conclusions two conclusions: (i) describing genes by their neighbours in the network using moments of the feature distribution results in a significant gain of classification performance, and (ii) while the propagation alone only leads to a minor improvement of classification performance compared to the best baseline, it is the combination of both, the computation of the moments followed by propagation, that gives the best classification performance.

8. Network-based classification of cancer driver genes

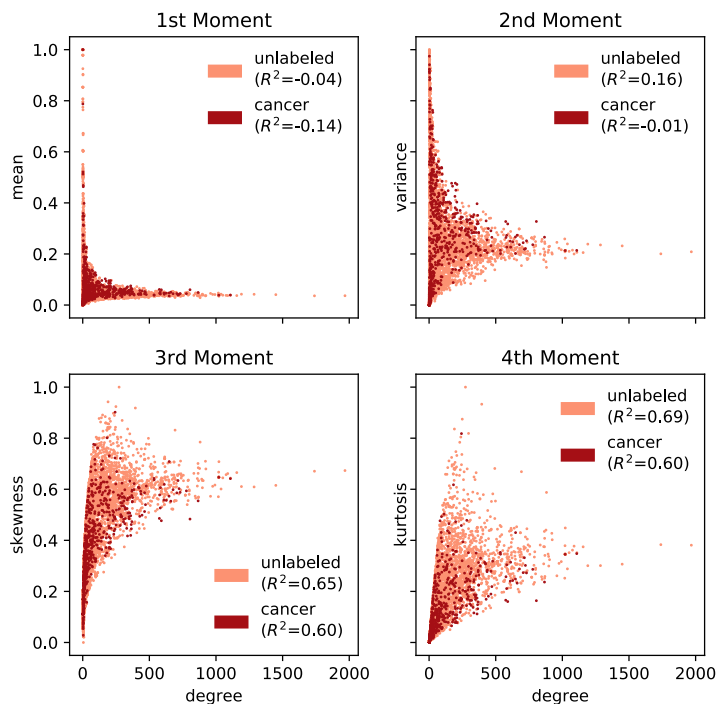


FIGURE 8.7.: The first four moments in the 1-hop neighbourhood for all nodes in the network, plotted against the respective node degree. Dark red markers correspond to the 635 cancer genes, light red markers to the unlabelled genes. The R^2 -value corresponds to the Pearson correlation between the respective moment and the degree. All moments were normalised to the range $[0, 1]$ for the ease of visualisation. (This figure is adapted from supplementary Figure S1 in Gumpinger et al. [256]).

8.2.3.4. Addressing knowledge bias in networks

As a last analysis of our results we are interested in how knowledge contamination is addressed. We stated earlier that the moment embeddings address this issue, as they describe the distribution of node features. However, empirically we find that especially the third and fourth moments are correlated with the degree of a node (see Figure 8.7), as the estimation of skewness and kurtosis becomes more accurate with increasing sample sizes. As a result, the degree could potentially work as a confounder. Importantly, we observe that cancer genes and unlabelled genes cannot be distinguished based on the moments, i.e. the positive correlations are present for both labels equally. We furthermore analyse those genes that were predicted as cancer driver genes with the logistic regression classifier, and observe that the degree distribution follows the degree distribution of the CGC cancer genes ($p = 0.06$, with a Kolmogorov-Smirnov test that assesses whether the degree distributions are significantly different), i.e. cancer genes are found within high- and low-degree nodes (see Figure 8.8). This result leads to the conclusion that our proposed approach empirically addresses the knowledge contamination present in the InBio Map network.

8. Network-based classification of cancer driver genes

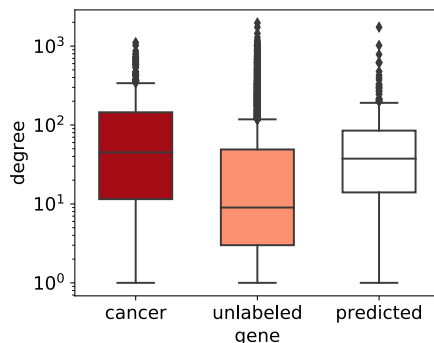


FIGURE 8.8.: The degree distribution of the 202 novel cancer driver genes predicted with logistic regression using the MoPro embeddings. The boxplots show the degree distributions of the CGC cancer genes (dark red), the unlabelled genes (light red), and the predicted genes (white). (This figure is adapted from supplementary Figure S2 in Gumpinger et al. [256]).

8.2.3.5. Qualitative evaluation of novel genes

Following the quantitative analysis of the results, we now focus on the qualitative analysis of the genes that were detected with the MoPro embeddings. To this end, we generate a set of consensus genes to focus on. The consensus set consists of those genes that were predicted as novel cancer drivers by any of the four classifiers logistic regression, support vector machine, random forest and gradient boosting⁵. The resulting consensus set contains a total of 50 candidate genes. Out of those 50 genes, 31 were significant according to **MutSig** ($p = 8.04 \times 10^{-42}$), 10 were significant according to **NetSig** ($p = 1.07 \times 10^{-6}$), and 12 were detected with **hierarchical HotNet** ($p = 2.04 \times 10^{-7}$). The p -values are computed with a hypergeometric test, and measure whether the consensus set is significantly enriched with **MutSig**, **NetSig**, and **hierarchical HotNet** genes. We remove the genes detected with those baselines from the consensus set. As a result, the final consensus set containing our suggested candidate cancer driver genes consists of 14 novel genes, for which we conducted a literature review for evidence of links between the genes and cancer.

Three genes, namely GATA4, ID2, and FOS, were found to have a direct link to tumourigenesis in humans. GATA4 is a transcription factor that negatively regulates the normal astrocyte⁶ proliferation. It appears to be a driver in glioma formation and fulfils the hallmarks of a tumour suppressor gene in Glioblastoma Multiforme [261]. Kijewska et al. [262] identified ID2 as a key regulator of breast cancer metastasis to the brain. It has been found up-regulated in brain metastasis, and elevated ID2 levels in breast cancer patients are linked to an increased risk of developing metastatic relapses in the brain. Lastly, the transcription factor FOS is implicated in the pathogenesis of bone tumours [263]. It has been shown to exhibit recurrent rearrangements in Osteoblastoma.

Five other genes (ACVR1B, CASP10, RAP1A, MYLK, CSNK1A1) exhibit strong links to

⁵The sizes of the set of novel cancer drivers for each individual method can be found in Table 8.2b, column ‘novel’.

⁶An astrocyte is a star-shaped cell type that can be found in the brain and spinal cord.

8. Network-based classification of cancer driver genes

tumours relevant behaviour in cells, and pathways involved in tumourigenesis. The gene ACVR1B (also known as ALK4) interacts with activin-A, and is associated to tumourigenesis via this interaction [264, 265]. The gene has furthermore been shown to exhibit somatic mutations in pancreatic cancers [266]. The CASP10 gene initiates cell apoptosis, such that inhibition of CASP10 effects apoptosis [267], which constitutes one of the hallmarks of cancer. The RAP1A gene is a member of the RAS oncogene family, and promotes cell migration an invasion, and has been linked to metastasis in oesophageal cancer [268]. MYLK (also known as MLCK) plays important roles in cell migration and tumour metastasis in breast [269] and colon cancers [270]. The CSNK1A1 gene is a member of the CK1 kinase family and regulates the autophagic pathway in RAS-driven cancers. Experiments that knocked-out the gene showed cell death in Multiple Myeloma [271, 272].

We analysed the remaining six genes CASP1, CASP14, RBL1, HNF4A, RALA, and DLGAP2, and found weaker links to cancer for all but DLGAP2, for example via expression or pathway membership [273–277]. However, we could not find a clear experimental evidence of their implication in cancer. For the DLGAP2 gene we could not find any evidence.

9. Discussion and outlook

Discussion

The focus of this chapter was the development of a novel approach for the network-guided detection of cancer driver genes from genetic mutation scores. While there exists a plethora of methods for the identification of those so-called cancer driver genes with the help of molecular networks, most approach this problem from an unsupervised perspective [32, 143–146], e.g. by exploiting principles of network propagation [65]. In this project, we aimed to identify cancer driver genes by adopting a *supervised* classification perspective. We achieved this by integrating the MutSig scores [233] with a protein-protein interaction network [25, 257] and with a set of well-established cancer driver genes, namely the genes collected by the *Cancer Gene Census* [236]. This set of cancer genes can be leveraged by reformulating the task of identifying cancer driver genes as a supervised classification task. In that sense, we are *learning from what we already know*: by incorporating information about the feature distributions of cancer driver genes, we can learn from those distributions to perform the prediction task.

To achieve the supervised classification, each gene has to be represented by a set of features. The main contribution in this project is a novel node-embedding which we call *moment propagation embedding*, short MoPro embeddings, that creates representations of each gene based on somatic mutation scores and the topology of the molecular network representing interaction effects between genes. Specifically, the MoPro embeddings consist of two components. The first component is the representation of each gene by the distribution of mutation scores of its k -hop neighbourhood in the network. Instead of working on the distributions directly, we represent the core-properties of each distribution by its first four moments, that is the mean, variance, skewness and kurtosis of the feature distributions. This constitutes a computationally efficient way of representing distributions for the classification task. The second component is inspired by procedures such as the Weisfeiler-Lehman aggregation [187] of features in a network, or network propagation [65], which inspired the name *propagation embedding*. The idea is that across a fixed number of propagation steps, the feature vector of each node in the network is updated based on the feature vectors in its local neighbourhood. The composition of those two components yields the MoPro embeddings, and we use those as features for the supervised classification of the genes in the network.

The reformulation as a supervised classification task bears two major challenges: (i) due to the small number of known cancer driver genes compared to the entirety of genes in protein-protein interaction networks, the data is highly imbalanced, and (ii) while the CGC genes provide us with a high-quality ‘positive’ class, we do not have a high-quality ‘negative’ class. In theory, each gene in the network is a potential cancer gene candidate. To address both challenges, we developed a sophisticated cross-validation scheme that

9. Discussion and outlook

relies on a repeated undersampling of the majority class, i.e. those genes that have not been identified as cancer driver genes by the CGC. This results in a different data set for each repetition. As we train a separate classifier for each of those data sets, we obtain a set of classifiers, each focussing on properties of the cancer genes of the current data set. This undersampling has the advantage of (i) reducing the class imbalance, as we can influence this by the undersampling rate, and (ii) for each repetition and resulting data set, the classifier learns to distinguish cancer driver genes from unlabelled genes.

We applied the MoPro embeddings combined with the proposed cross-validation using four well-established classification algorithms scheme to discover novel cancer driver genes in a TCGA cancer data set. We compared our proposed approach to a variety of baseline methods, some of which identify cancer driver genes in an unsupervised setting, while others exploit the knowledge about the cancer driver status of genes. Those analysis indicated that methods which exploit the known cancer status of genes lead to an improved performance. We also compared our approach against two well-established network-guided methods, namely *hierarchical HotNet* [146] and *NetSig* [32], as well as a recently published network-propagation approach called *uKIN*. *uKIN*, in addition to propagating mutation scores, also propagates class labels through the network [148]. We observed that *uKIN* is superior to the other two network-based methods, hence corroborating our above conclusion that including prior-knowledge in the form of known cancer driver genes improves performance.

We furthermore analysed different aspects of the model, such as the dependence on cross-validation hyperparameters. Those analyses showed the robustness of the predictive performance to the number of undersampling iterations in the cross-validation procedure. Furthermore, we evaluated the dependence of the performance metrics on the number of samples in the training set, and found that a training set size of 1'000 samples constitutes the lower bound to guarantee a satisfactory predictive performance. An ablation study showed that the main improvement in predictive performance is presumably caused by the representation of genes based on moments of feature-distributions in its neighbourhoods, and to a lesser extent by applying the network propagation step. One important aspect when working with molecular networks in the context of cancer is the so-called *knowledge contamination*, i.e. the phenomenon that well-studied cancer genes tend to have more neighbours in those networks. In the light of classification, this implies that the degree potentially constitutes a confounder. We observed that this is indeed the case in the InBio Map network, as predicting based on the degree already achieved a high performance with respect to the area under the receiver operating characteristic. We empirically found that the proposed moment embeddings address the knowledge contamination in the network, and that the degree-distribution of the predicted genes follows the degree-distribution of the known cancer driver genes.

We concluded our analyses with a qualitative evaluation of a set of consensus genes that were detected using the MoPro embeddings. This set consisted of 50 genes, and contained both genes that could be discovered with established methods such as *MutSig*, *NetSig*, or *hierarchical HotNet*, as well as 14 'novel' cancer driver genes that none of those methods were able to discover. Through a literature review of those 14 novel genes, we found evidence for their implication in cancer for all but one gene, and for three of those

9. Discussion and outlook

genes, we found strong evidence. This shows the power of our proposed approach to discover biologically meaningful novel cancer driver genes, and the set of proposed cancer drivers constitutes a promising target for future biological validation.

In conclusion, we proposed a framework to generate network-derived features that enable the supervised classification of cancer driver genes by applying a sophisticated cross-validation procedure. This cross-validation scheme addresses the class imbalance and lack of a high-quality negative class, induced by the small number of cancer drivers in comparison to the body of all genes. The proposed node embeddings address the problem of knowledge contamination in the network, and enable in combination with the cross-validation procedure a robust classification of cancer driver genes. Combining both, we were able to identify novel, and most importantly biologically meaningful, cancer driver genes, that constitute promising targets for follow-up experiments.

Outlook

Cancer is a widespread disease and among the leading causes for death [215] across most age groups. Due to its prevalence in the human population and the global health challenges it imposes, combating cancer has become one of the biomedical goals of the century. To achieve this goal, collecting and sharing of data, knowledge and information is indispensable. To this end, various large-scale collaborative projects, such as *The Cancer Genome Atlas*, the *International Cancer Genome Consortium* [278] and the *Cancer Gene Census* [236] aim at sharing a variety of cancer-related data with the research community.

Cancer is a genetic disease that it is predominantly caused by the aggregation of genetic aberrations during an individual’s lifespan. Hence, one major mode of analysis in cancer related research focuses on the analysis of tumour genomes compared to matched normal tissues, with the goal to identify genes that are, upon alteration, linked to the development and progression of cancer. Those genes constitute promising candidates for diagnostic and therapeutic development. A plethora of computational and statistical tasks for this purpose has been developed over the past 20 years, and lead to the discovery of novel genes implicated in cancer. Recently, methods that incorporate interaction information between genes derived from molecular networks into this process have shown great successes [32, 146, 148].

This part of the thesis was devoted to the description of an approach that integrates genetic mutation scores with molecular networks for the *supervised* prediction of cancer driver genes and as such differs from most established methods in that it leverages information on well-known cancer driver genes. There exist various directions for further research, focussing either on technical aspects of the method, or on biological ones. We discuss those directions in the following.

Representation of distributions

The main contribution of this project was the derivation of novel node embeddings, which we gave the name *moment propagation embeddings* (MoPro embeddings). They constitute a two-step approach that first describes the gene by the distribution of features in its k -hop

9. Discussion and outlook

neighbourhoods. We described those distributions via their first four moments. The second step is inspired by a network propagation, i.e. the moments are successively propagated through the network. An ablation study showed that the main improvement of predictive power compared to baselines can be attributed to the computation of the moments, rather than the propagation step. Hence, information that is helpful during prediction appears to be contained in the neighbourhood distributions. While the representation via moments is computationally appealing, there exist other options to achieve the classification of two genes based on their neighbourhood distributions. One compelling concept is that of optimal transport [279]. Togninalli et al. [280] developed a family of graph kernels called Wasserstein Weisfeiler-Lehman graph kernels. They are based on Wasserstein distances for the classification of graphs, and their idea can be readily extended to the task of node classification.

The intuition underlying the Wasserstein distance is the computation of a distance between two probability distributions by finding the cheapest way to move all the probability mass of one distribution to match the second distribution. Intuitively, this distance will be small for distributions that exhibit similar probability masses, and larger for distributions whose probability masses are very dissimilar.

In our setting, each gene can be represented by feature values in its k -hop neighbourhood in the molecular network, constituting a random sample from the underlying probability distribution. The Wasserstein distance for discrete distributions, proposed in [280, Equation (2)], can hence be used to compute pairwise distances between the genes, resulting in a kernel matrix that can be subjected e.g. to a support vector machine for the classification of the genes. This classification is compatible with the cross-validation procedure described in this project, hence challenges caused by class imbalance and lack of a negative class can be addressed in the same way. One major impairment of the Wasserstein Weisfeiler-Lehman kernels proposed in [280] is the computational complexity of the analysis. However, in case of our cross-validation procedure all pairwise distances would have to be precomputed only once, as the cross-validation does not affect the distances.

Graph convolutional networks

The proposed MoPro embeddings constitute a way of representing each gene in a molecular network by a set of features. The computation of those features is defined by the propagation of moments that describe the k -hop neighbourhoods of nodes through the network. While there are some hyperparameters to the generation of the features, such as the number of k -hop neighbourhoods to explore or the number of propagation steps, the feature generation process is completely deterministic given those parameters.

An exiting direction of further research is to deviate from this deterministic approach, and move towards a setting in which the embeddings are *learned* in an end-to-end manner to improve the prediction of nodes. This idea is at the core of graph neural networks [reviewed in 281], an area of deep learning that gained a lot of traction over the last couple of years. The gist of those methods is the classification of graphs, or of nodes within one single graph, by learning graph (or node) representations that incorporate the network topology as well as node-level information. One prominent member of this group of methods are

9. Discussion and outlook

graph convolutional networks (GCN) that generalise the concept of convolution underlying convolutional neural networks (CNN) to the graph domain. The main idea is to represent each node in a network by aggregating its own feature with features from its neighbours in the network, and there exists a rich body of literature how this aggregation can be achieved [see 281, Table 3].

Especially the semi-supervised approach to train graph neural networks proposed by Kipf & Welling [188] appears promising in our setting. The proposed GCN aims at the classification of nodes in a network, where labels are only available for a small subset of nodes, which they refer to as *semi-supervised* classification. The resemblance of this task to our problem statement in supervised cancer gene classification is conspicuous. While Kipf & Welling [188] proposed a GCN with layers as first-order approximations of spectral graph convolutions [282], the idea of training a GCN in a semi-supervised fashion can readily be translated to any other GCN architecture.

In the context of cancer driver gene prediction, we envision the representation of nodes in a network by a features, such as mutation scores. In case other node-level features, such as structural properties of genes, or deleteriousness scores, are available, those could be readily integrated. Next, those features can be superimposed to nodes in a molecular network and be used as input to a GCN. Training the GCN in the semi-supervised manner proposed by Kipf & Welling [188], thereby *learning* the node representations instead of *defining* them, appears as a compelling direction of future research.

Representation of genes

In this project, we initially represented each gene in a network by its **MutSig** statistic [233] which measures frequency, clustering and functional impact of somatic mutations in the gene. When superimposing those scores on the network, the resulting weighted network represents the distribution of mutational patterns across human gene-gene-interactions. This vertex-weighted network is then subjected to the proposed moment propagation embeddings, such that the resulting embedded node features represent this distribution of mutational patterns across the network, without explicitly requiring the network topology, thereby enabling the classification with off-the-shelf machine learning algorithms.

There are other statistics and methods to measure the functional or structural impact of somatic mutations, and we reviewed some of those methods in the introduction (Section 7). By the initial representation of a gene with a score, we determine our prior hypothesis about the emergence of cancer, and by changing the score, we thus also change our prior assumptions. Hence, while the **MutSig** scores are clearly a well-established tool for the representation of the degree to which a gene is involved in cancer, it might well be the case that genes which drive tumourigenesis via different processes, such as structural modifications of the protein, are missed. Applying the moment propagation embeddings with a different initial score might hence shed light on genes that drive cancer by different mechanisms. Along those lines, a single gene can also be represented by more than one feature, and the can still be computed in this setting. The combination of multiple somatic mutation scores might lead to a more holistic description of the mechanisms underlying cancer.

9. Discussion and outlook

One could take this even one step further: while somatic point mutations unarguably harbour potential to drive cancer development, other types of genetic aberrations have been linked to the malignant transformation of cells as well. Examples are translocations, deletions, copy number variations, or the fusion of genes [254]. Including this type of information into the generation of the moment propagation embeddings of genes constitutes a promising line of future research.

Cancer subtype analysis

In this chapter, we focussed on the pan-cancer analysis of TCGA tumour profiles. This implies that *all* tumour profiles were combined and analysed jointly, irrespective of the tumour subtype. However, the TCGA does not only provide the tumour profiles, but also a classification of each tumour into 33 tumour subtypes. Similarly, the *Cancer Gene Census* annotations of known cancer drivers contain not only well-established cancer genes, but also the tumour type the genes were implicated in, such that the data for a tumour-subtype-specific analysis is readily available.

While a pan-cancer analysis sheds light on the genetic causes that are shared by many tumours, it might gloss over details for each of the subtypes. It is a well-established fact that different tumour types are driven by alterations of different genes, and while a gene might be found to be heavily mutated in one cancer type, the same gene might not be significantly mutated for another type [254]. Furthermore, different tumour subtypes exhibit highly heterogeneous mutation rates, for example melanomas and lung tumours typically exhibit high numbers of non-synonymous mutations¹, while paediatric tumours and leukaemia are less frequently mutated [254].

Especially from a diagnostic and therapeutic viewpoint, disentangling the genetic causes underlying specific tumour types is of utmost importance. It allows for the administration of so-called targeted cancer therapies [283] that inhibit cancer growth by interfering with specific molecular targets. The development of such a targeted anticancer treatment is at the heart of personalised cancer treatment: based on the mutational patterns of an individual's tumour, driver genes can be identified and targeted with specific drugs. For example, the cancer drug *Vemurafenib* targets a mutation of the BRAF protein that is often present in melanomas, and is administered to patients with inoperable or metastatic melanomas with this mutation [283]. As opposed to standard generic therapies such as chemotherapy that exhibit cytotoxic effects on fast-growing and dividing cells, those targeted therapies are often cytostatic, i.e. they block the proliferation of tumour cells. Those drugs can be developed by identifying the molecular targets (i.e. cancer driver genes), and developing a drug that interferes with the target's potential to drive tumourigenesis [283]. Hence, understanding the genetic causes underlying individual tumour subtypes can spark the development of anticancer drugs that target the respective cancer driver, and hold great potential for the personalised treatment for all forms of cancer.

¹A mutation that alters the amino acid sequence of the protein product.

Part IV.

Discussion and outlook – Learning from structured data

Context and overview

This concluding part of the thesis is devoted to the discussion of (graph-) structured data. It starts with a short digression into another form of structure that often underlies data, namely time-structure, and we briefly describe a contribution of the thesis' author to this topic in the field of synthetic biology.

While the general theme of this dissertation was the implication of molecular networks to decipher the genetics underlying complex traits, contributions were made to two particular areas, that is the implication of molecular networks in genetic association studies in Part II, and in the discovery of cancer driver genes in Part III. We discussed each of those topics in detail, and provided a specific outlook at the end of those two parts (see Chapter 6 and Chapter 9). Here, we would like to briefly summarise the main gist of the contributions in the light of network-biology.

The remainder of this thesis describes recent trends and challenges in learning from graph-structured data, and we conclude with a brief statement about their inclusion into the analysis genetics underlying complex traits and diseases.

10. Learning from structured data

10.1. Time-structured data

The focus of this dissertation was to leverage molecular networks to unravel the genetics underlying complex traits. Consequently, the thesis was centred around the concept of graphs. However, there is another important form of structure that can underlie data, namely a time-resolution [284]. This is the case in various disciplines, including economy, physics, medicine, and biology. In the following, we will focus on the latter two groups, medicine and biology.

An important field of research is the analysis of medical time series, as they allow for the early prediction of events in patients. Those time series commonly track the development of physiological variables of patients over time, which includes measurements such as heart rate, blood pressure and temperature. Hyland et al. [285] have shown the value of this type of data for the early prediction of circulatory failure in patients in the intensive care unit, and algorithms that aim to discover patterns in medical time series are an active field of research [186, 286].

Biological time series commonly measure cellular properties, such as cell morphology or fluorescence¹ over time. This data can be obtained from optical measurements, for example from microscopy images, both for single cells and whole cell cultures [287]. This constitutes a compelling approach, as most biological processes are governed by defined spatio-temporal dynamics. In consequence, end-point measurements of cellular properties are not always sufficient and are prone to losing important information.

One interesting application is to measure gene expression over time as a response to DNA alterations, thereby addressing the question to what extent a modified genetic sequence leads to a change in gene expression over time. Answering this question requires a data set that links genetic sequences to a time-resolved curve that measures the gene expression given the genetic sequence. This type of data opens the door to an exciting machine learning task, namely the prediction of function from sequence. However, such an approach also raises a challenge with respect to experimental design: the choice of the time points at which measurements should be taken, such that the dynamics are still represented without losing important information. Our collaborators Dr. Markus Jeschek, Prof. Dr. Kobi Benenson and Simon Höllner developed a novel approach to enable these kinds of measurements in *Escherichia coli*, short *E. coli*, and the authors of this thesis contributed to the experimental design regarding the time series. We briefly outline this contribution in the following section, and refer to the original publication for an in-depth description of the whole project:

¹Where fluorescence could be linked to the expression of a specific gene.

10. Learning from structured data

Höllerer, S., Papaxanthos, L, **Gumpinger, A.C.**, Fischer, K., Beisel, C., Borgwardt, K, Benenson, Y., and Jeschek, M.. *Large-scale DNA-based phenotypic recording and deep learning enable highly accurate sequence-function mapping*. In press at *Nature communications* (2020), bioArxiv preprint: [288].

Optimising sampling times for sequence to function mapping in synthetic biology

A major challenge in biology is to understand the effect of genetic modifications of gene-regulatory elements (**GREs**) on gene expression. Existing protocols focus on finding genetic variants that exhibit desired properties, such as expression of a specific gene in a target range, rather than systematically exploring the vast space of variants. This is mostly achieved by expensive multi-step screening experiments that (i) generate libraries containing hundreds of thousands of modified GREs at random, (ii) transform or transfect those modified GREs back into cells, (iii) select cells that show the desired behaviour, e.g. based on fluorescence, and (iv) sequence the genetic code of those variants. This commonly results in data sets containing the genetic sequences of tens to hundreds of cells, combined with their respective functional readout. While those experiments might result in genetic variants that exhibit the desired properties, they are insufficient to foster a comprehensive understanding of the genetic landscape underlying gene expression, as they are limited to a few hundred variants. Höllerer et al. [288] developed an innovative genetic construct to address those limitations. It is based on recording the phenotype on the same DNA molecule as the genotype, such that genotype and phenotype can be determined in a single sequencing read. Using next-generation sequencing **NGS**, this approach enables the generation of large-scale sequence-function data sets. These can be leveraged in combination with deep learning, not only to predict function from sequence, but also to enhance our understanding of the genetic mechanisms that give rise to the function.

The core of the approach is a three-component genetic construct, consisting of a site-specific DNA recombinase (*modifier*), a GRE that controls the expression of the recombinase (*diversifier*), and the recombinase substrate (*discriminator*). All three components are all located on the same DNA molecule. The modifier element alters the genetic sequence of the discriminator, such that the discriminator can only exist in one of two mutually exclusive states, native or modified, corresponding to the *phenotype*. The diversifier, which in our context is the genotype under investigation, regulates the expression of the modifier. As a consequence, the genotype and the phenotype (i.e. the discriminator in its *innate* or *flipped* state) can be read off the same DNA molecule. This genetic construct was named uASPIre (ultra-deep Acquisition of Sequence-Phenotype Interrelations). Importantly, the modification of the discriminator results in a binary phenotype on the level of single DNA molecules. However, by repeating this for many DNA molecules that share the same diversifier, we can compute the fraction of flipped discriminators for the given sequence at a specific time point. A time component can be added by taking measurements at various time points after initiating the experiment. As a result, for one diversifier, or genotype, we can obtain a time-series like curve that represents the fraction of flipped diversifiers over time, which we refer to as the modifier's *flipping profile*. Intuitively, 'strong' genotypes will result in curves that quickly reach 100.0% flipping, while 'weaker' phenotypes result in less steep curves.

10. Learning from structured data

The DNA molecules that contain the genotype and phenotypes obtained with uASPIre can be sequenced leveraging high-throughput next-generation sequencing techniques. However, since the number of DNA molecules that can be analysed in one next-generation sequencing run is limited, various design choices have to be made to guarantee a high quality of the resulting genotypes and flipping profiles, while maximising the number of investigated sequences. One of the most important design principles is the optimal selection of sampling points, which determines the granularity of the resulting time series. While high numbers of samples increase the dynamic resolution, they use up more capacities during sequencing. Hence, our goal was to reduce the number of samples, while preserving the flipping dynamics.

This optimisation was done in a first proof-of-principle experiment in *E. coli*. The serine recombinase Bxb1 was chosen to be the modifier, and the GRE was a library of ribosome binding sites (RBS) which control the expression of Bxb1. The proof-of-principle library contained roughly 10'000 sequences and was subjected to the uASPIre workflow [288]. Flipping profiles of all 10'000 genetic sequences were recorded over 24 hours with 18 sampling points. To further increase the throughput of uASPIre, a reduction of sampling time points was necessary. Hence, we optimised the sampling regime to reduce the number of sampling times by applying a greedy search strategy. It consists of a multi-step procedure that first approximates each flipping profile by a logistic function to account for measurement noise, and fixes the first time sample at the time of induction, and the last sample at 720 minutes after induction. Subsequently, the following steps were repeated iteratively, until 18 time points were chosen: (i) find the next time point that, when added to the sampling regime, reduces the average error between the fit to each profile, and the linear approximation through the currently chosen sampling times, the most, and (ii) add that time point to the sampling regime, and return to (i). This led to a set of nine time points, that were used for the next large-scale experiment. This sampling optimisation² was performed in second large-scale RBS experiment, that resulted in a data set of 303'503 RBSs and their respective flipping profiles, constituting over 2.7 million sequence-function pairs. Notably, this increase in throughput was achieved without changing the NGS device or protocol.

This large-scale data set of unprecedented size was subsequently exploited to train a deep neural network, named SAPIENs, for the quantitative prediction of function from sequence. SAPIENs was able to confidently predict RBS activity, and consistently outperformed off-the-shelf machine learning classifiers. Analysing the model in greater detail shed light on different properties of RBSs, such as the translations- promoting effect of Guanine nucleotides, and the translation-reducing effect of Cytosine nucleotides, as well as the presence of Shine-Dalgarno motifs in sufficient distance to the start codon in strong RBSs [289].

This joint project emphasises the potential of combining machine learning, deep learning, and synthetic biology to enhance our understanding underlying the sequence-function landscape. An integral part of this project is to dynamically resolve sequence-function behaviours as opposed to end-point measurements, as is often the case in high-throughput experiments.

²Other optimisations were performed by other parts of the collaboration, see Höllerer et al. [288].

10.2. Summary and Discussion

The dissertation at hand focussed on the development of novel machine learning techniques that aim at unravelling the genetics underlying complex traits by integrating genetic data with molecular networks. While the first part of this thesis contained an introduction to network biology in general, we focussed in particular on molecular protein-protein interaction networks that describe physical interactions between genetic products in the form of graphs. Those networks offer a holistic view of molecular processes taking place on various scales and promise to shed light on complex mechanisms that potentially lead to the development of complex traits.

Network-guided association studies

In the second part we described two contributions that extend classic genome-wide association studies with network information to address the problem of *missing heritability* which is assumed to be partially attributed to the negligence of non-linear interactions between genetic variants. We contributed to this task by developing two methods which are capable of integrating molecular networks into genetic association studies to serve as biological prior knowledge. This is achieved by only analysing interactions between genetic variants that are harboured within genes that interact according to a network. The benefit of biological networks in this context is twofold: firstly, it reduces the vast search space containing all possible interactions between all genetic variants. This search space is immense, posing a statistical challenge in the form of a multiple hypothesis testing problem, and a computational challenge, as those interactions have to be enumerated and tested. By focussing only on interactions that are supported by the network, these challenges are not completely resolved, but alleviated. Additionally combining those approaches with concepts from significant pattern mining that have proven especially successful in further reducing the statistical and computational challenges, eventually makes a previously-infeasible problem tangible, so that the methods can be applied in practice. Secondly, deriving interactions from networks implies that the combined variants can be interpreted biologically by means of the edge type, as edges in the network commonly imply either an experimentally validated or theoretically predicted interaction³.

Some protein-protein interaction networks, such as String [26] or the InBio Map [25], contain edge weights that indicate the confidence of an interaction. Our proposed methods currently do not account for edge weights, and this surely constitutes an interesting direction of future work. However, including low-confidence interactions into analyses is highly promising, as in those cases, our methods allow to corroborate low-confidence edges, if a significant hit is found within the edge. We did observe this for a study of migraine patients, where a significant interaction was found within a low-confidence edge. This further confirms the existence of the edge in the network.

Our models investigate network-guided interactions between genetic variants under a model of genetic heterogeneity. It assumes that multiple genetic variants influence a phenotype in a similar way. The results obtained with our approaches indicated that this

³Of course this might vary from network to network.

type of non-linear interaction takes place on the scale of only few genetic variants, such as single nucleotide polymorphisms or rare variants. In other words, we found interactions in the form of edges and segments within those to be the most informative parts of a network when it comes to genetic heterogeneity.

Classification of cancer driver genes

The third part of the thesis focussed on a different task, namely the identification of cancer driver genes, i.e. genes that are, upon mutation, implicated in the development and progression of cancer in humans. To achieve this, we developed a novel node embedding that is based on the representation of each gene by a mutation score. For each gene, this score measures to what degree the frequency, clustering and functional impact of mutations differs between a tumour and a matched normal sample. Combining this information with network information is inspired by the observation that network and pathway information is central to cancers, and that cancer genes are commonly involved in pathways that regulate cell growth and cell fate [254]. The underlying hypothesis is that cancer is a consequence of the disruption of a pathway, rather than a specific gene within a pathway.

To achieve this, we developed a node embedding of mutation scores that combines two major concepts. The first one is the representation of each gene by the distribution of its neighbours' mutation scores in the network. To obtain a vectorial representation of those distributions, we describe the distributions by their moments. The second concept are network propagations that have proven successful in a variety of tasks in computational biology, described by Cowen et al. [65], including the identification of cancer driver genes [32, 143–146]. The so created node embeddings, which we named *moment propagation* (MoPro) embeddings, can then readily be used as an input to any binary classifier, where well-established cancer driver genes constituted the positive class which we aimed to predict. To this end, a specific cross-validation scheme was implemented to address the resulting high class-imbalance as well as the lack of a high-quality negative class.

Interestingly we observed that the major improvement of classification performance compared to the baselines can be attributed to the moment embedding, as opposed to the propagation embeddings, indicating that *describing* a node's neighbourhood is more meaningful than *aggregating* a node's neighbourhood in the network. One important property of protein-protein interaction networks is the so-called *knowledge contamination*, i.e. that well-studied cancer genes tend to have higher degrees in the network. This phenomenon results in a bias when incorporating network-information into the prediction task, as the degree of a gene potentially acts as a confounder. We observed empirically that our MoPro embeddings enable the prediction of both, low-degree and high-degree cancer driver genes, which is a desirable outcome. As opposed to existing methods that mostly approach the task of cancer gene prediction from an unsupervised perspective [32, 146, 233], we set out to classify cancer driver genes in a supervised setting. We found that including the knowledge of well-established cancer driver genes greatly helps the discovery of previously unknown cancer drivers.

10.3. Trends and challenges in learning from graph-structured data

We are living in a world full of networks, may they be maps, telecommunication networks, social networks, citation networks, cells, or proteins. Most complex systems can be represented by means of networks in one way or the other. The ever-accelerating technological advances result in a growth of those networks at remarkable rates. They grow with respect to both the number of nodes as well as the number edges contained in them, and their analysis becomes an increasingly important topic across various research fields [290]. The past decades have seen an increase in methods and tools that are developed for exactly this purpose, that is the extraction of knowledge from graphs across different disciplines. Not only do those contributions span various research areas, but also different theoretical concepts, ranging from graph kernels to node embeddings and graph convolutional networks. Here, we would like to outline the current *hot topics* evolving around graph-structures, and their implications in biomedical research.

Graph kernels are among the most prominent and well-established approaches for classification tasks related to graphs [291]. They constitute a versatile approach to either classify nodes within a single graph, or to consider a graph itself as an object that is to be classified, as could for example be the case for the task of protein function prediction, where each protein is represented as a graph. The underlying idea is to compute a kernel that represents similarities between two objects, i.e. between two nodes or graphs, in a usually high-dimensional, or even infinite-dimensional feature space, without having to define the function that maps the objects to this space (*kernel trick*). The kernel should be able to capture the essence of the graph required when subjected to a kernel machine (such as support vector machines) for a successful classification. The development of novel graph kernels has been an active field of research for almost two decades [290] with many famous examples, such as Weisfeiler-Lehman kernels [187], shortest path kernels [292], or hash graph kernels [293], as well as fairly new approaches [280, 294]. Graph kernels have been successful across various tasks in biology, such as prediction of protein function [17, 280, 295] and the identification of disease genes [296] as well as in chemo-informatics, where they were, e.g., leveraged to predict molecule properties [280, 295].

A recently emerging field of research are node embeddings, often also referred to as *graph representation learning* or *graph embeddings* (for a detailed review, we refer the interested reader e.g. to Hamilton et al. [297] and Zhang et al. [298]). At the core of those methods lies the learning of low-dimensional representations of vertices in networks that account for the local and global network structure. In other words, the often notoriously complex network structure is translated into a vectorial representation of nodes. Similarly to graph kernels, such embeddings can be leveraged for both node and graph classification. Other than graph kernels that do not rely on an explicit representation in the feature space, node embeddings create such an explicit representation, sometimes called the latent representation. To be more precise, they *learn* this representation in a way that facilitates downstream supervised or unsupervised machine learning tasks, such as node or graph classification, link prediction or clustering. Those embeddings can either be trained for a specific classification task end-to-end, or they can be trained in an unsupervised manner

10. Learning from structured data

which requires the definition of a meaningful objective function. Prominent graph embeddings include the DeepWalk algorithm [299], node2vec [300], or the LINE method [301]. Those graph embeddings are agnostic to the underlying network type, such that they can be leveraged across various fields, including social networks, citation networks or molecular networks. Yue et al. [302] provide a systematic evaluation of different node embedding methods in the realm of biomedical networks, with a focus on link prediction in drug-disease association networks and drug-drug interaction networks, as well as node classification for the prediction of protein functions. They find that node embeddings are a highly promising concept that is competitive with or superior to traditional methods that rely e.g. on matrix factorization.

Furthermore, a deep-learning inspired approach that gained a lot of attention in the recent years are graph convolutional networks (GCNs). The central idea underlying graph convolutional networks is to translate the concept of convolution, which has proven highly successful in the field of image classification [303], to graph-structured data. This is mostly achieved by iteratively aggregating the signal of individual nodes based on their local neighbourhoods in networks, an idea reminiscent of the Weisfeiler-Lehman graph kernel. A plethora of different types of graph convolutional networks have been proposed over the last five years that present different architectures and aggregation functions, and they have been useful for many different applications, ranging from computer vision to natural language processing and chemistry. Graph neural networks, such as *graph autoencoders* have been used to generate embeddings of nodes, emphasising the relatedness to the aforementioned node embeddings. We refer to Wu et al. [281] for a comprehensive review of different neural network architectures on graphs, including graph convolutional networks. While the application of GCNs to biomedicine is a recently-emerging field, initial research shows the potential of marrying biological networks with graph convolutions. Zitnik et al. [304] applied GCNs to the task of predicting polypharmacy side effects⁴ from networks containing three different types of interactions, that is protein-protein interactions, protein-drug interactions, and drug-drug side-effect interactions. Another promising application was presented by Fout et al. [305] that successfully leveraged GCNs for the prediction of protein interface interactions based on the 3D structures of proteins.

Those success stories of graph-based learning across various disciplines and questions in biomedical research emphasise the benefit of including graph-structured data into the processes we use to generate new knowledge. Not only do they enable a holistic view of whole systems by taking interactions into account, but they also improve our capabilities to predict and understand molecular mechanisms that, for example, lead to complex traits and diseases. The fact that methods evolving around the graph-structures underlying many data sets continuously gain attention is highly promising, although there still lie challenges ahead.

Interpretability of learning-based approaches

One long standing challenge in deep learning based approaches is the interpretability of the mechanisms that give rise to predictions. While machine learning methods have

⁴Side effects arising due to administering multiple drugs at the same time.

10. Learning from structured data

reached near-human performances across various tasks, many algorithms continue to be treated as black boxes. Digressing from this notion to an interpretability-driven evaluation of predictions is a critical challenge to all learning-based methods, including the ones mentioned in the preceding paragraphs.

Especially in the biomedical field, where predictions potentially form targets for novel diagnostic or therapeutic tools, it is indispensable to fully understand the mechanisms that give rise to the predicted values [306]. It is this concept of interpretability that allows humans to assign a level of confidence to the predictions [307]. Taking an example from Pope et al. [308] that analyses a data set of molecules with the goal to predict whether or not the molecule permeates the blood brain barrier, a task that is of specific interest in drug development. While it is without a doubt a successful outcome being able to predict whether or not the molecule possesses this ability, it is even more important to understand *which* properties of the molecule are the decisive factor for this ability. In the same publication Pope et al. [308] describe steps to enable such interpretations, by extending interpretability methods from the domain of convolutional neural networks to the graph convolutional network domain.

Time-varying graphs

Another important challenge for future work is to account for graph structures that vary over time. Methods such as graph convolutional networks or node embeddings, commonly assume graphs to be static, which means that neither nodes nor edges nor their attributes or labels vary over time [309]. However, for many graph-based problems this is not necessarily the case. Thinking of social networks, nodes might be added or removed upon individuals joining or leaving a social network, new edges might arise due to new friendships being made, and so on. Similarly, biological processes underlie time-dependent behaviours, might this be the up- or down-regulation of the expression of genes in response to environmental or cell-cycle related stimuli, or the rewiring of a protein-protein interaction network as a consequence of the mutation of specific genes [57]. Including the concept of time-varying graphs into recently emerging graph-based learning procedures constitutes a highly-promising and relevant direction of further research.

10.4. Closing remarks

The ever-increasing speed of technological advancement leads to the collection and availability of equally growing data sets. Parallel to and in line with this data revolution increases the need for scalable algorithms to mine the knowledge that is hidden in those large-scale data sets. Machine and especially deep learning techniques that are capable of leveraging, and even more so *require* those massive amounts of data, are gaining more and more attention, especially in fields such as image recognition or natural language processing [310].

Deep learning techniques have also proven successful for various biological tasks, including analyses of the genome, such as prediction of the methylation status, gene expression or

10. Learning from structured data

the control of splicing, as described in Zou et al. [311]. However their application to discover genetic causes underlying complex traits is still in its infancy.

Despite large-scale bio-bank efforts, such as the ones described in Bycroft et al. [83], Nagai et al. [84], and Chen et al. [85], that aim to sequence hundreds of thousands of individuals, most genomic data sets are affected by the $n \ll p$ problem⁵, hindering the training of the highly parametrised models used in deep learning approaches. Related to this, those data sets are inherently imbalanced and noisy, meaning that the number of non-trait-causing variants exceeds the number of trait-causing variants by far. In other words, most features are *uninformative* for the trait of interest, further complicating the application of deep learning approaches. On the contrary, classical methods that analyse the impact of mutations on a phenotype allow for an easier interpretation of the findings, albeit suffering from other limitations, such as the multiple hypothesis testing burden, or the negligence of non-linear interactions between mutated positions that might lead to downstream effects on the phenotype. This led to the development of a vast number of methods that incorporate network information to overcome those limitations in different ways. For example methods based on network propagation [65, 143, 144, 146, 148], the greedy exploration of sub-modules within networks [63, 64], or the local aggregation of features [32] have shown to successfully identify novel genes presumably implicated in the traits of interest. Similarly, the methods proposed in this thesis live at the intersection between machine learning and genetics, and allow by design an interpretation of results, as the analysed patterns were described explicitly by leveraging the underlying graph structures. We have shown that they constitute valuable tools to decipher the genetics underlying various complex traits that yield salient patterns for further biological validation.

However, jointly exploring genomic data and molecular networks in a deep learning setting appears a compelling direction for future work. It requires addressing the different challenges inherent to genomic data, molecular networks as well as deep learning approaches, and to bring those different concepts in line. This entails addressing problems induced by (i) the data dimensionality in the form of the $n \ll p$ problem, (ii) the large number of uninformative features, (iii) the noisy and incomplete network structures, as well as (iv) the interpretability of the output.

Given the traction the various types of methods for the analysis of graph-structured data have gained over the past decades, we look curiously into the future, with the hope that existing and emerging methods will continue to transform our understanding of complex traits, molecular dynamics, and diseases. We envision that methods which adopt holistic views by incorporating network-based information will thrive to become a central pillar of various research disciplines and be firmly integrated into the toolbox of every data scientist.

⁵The number of features exceeds the number of samples by far

Part V.
Appendix

A. Data-simulation for significant pattern mining approaches

Simulation of binary data

Disclaimer: the core of this simulation procedure follows the one proposed by Llinares-López et al. [165]. We first start by describing the generation of the binary phenotype, as well as the pattern indicator for the truly associated pattern, and the confounded pattern. We create them as n -dimensional binary vectors, where n is the number of samples in the data set, and denote them as \mathbf{y} , \mathbf{g}_t and \mathbf{g}_c , respectively. To achieve this, we draw the three vectors from a multivariate Binomial distribution with mean vector μ and covariance matrix Σ , using the R-package *binsim*, where

$$\mu = [0.5 \quad 0.5 \quad 0.5], \quad (\text{A.1})$$

$$\Sigma = \begin{bmatrix} 1 & p_s/2 & p_{con}/2 \\ p_s/2 & 1 & 0 \\ p_{con}/2 & 0 & 1 \end{bmatrix} \quad (\text{A.2})$$

The parameters p_s and p_{con} can be used to regulate the strength of the association signal between the phenotype and the significant and confounded subgraph, respectively, and we chose $p_s = p_{con}$. Following Llinares-López et al. [165], we add a small disturbance to the vector \mathbf{g}_c by flipping each of its values with low probability $p_\epsilon = 0.05$, resulting in the covariate vector \mathbf{c} .

The pattern indicator vectors \mathbf{g}_t and \mathbf{g}_c will be used to derive a contingency table, from which the statistical association is derived. The process of generating the data that results in the pattern indicator of the truly associated and confounded pattern is identical, hence we will explain this procedure only for the truly associated pattern. Importantly, for all simulations it is ensured that the truly associated and the confounded pattern do not overlap in the network.

Simulation of data for the tNeAT methods

To simulate data for the tNeAT method, we generate artificial data for the KEGG pathway with identifier hsa05205, ‘Proteoglycans in cancer’. The pathway contains $d = 224$ nodes and $m = 1'324$ edges. In each simulation, we generate a data set with 224 features, corresponding to the 224 nodes in the pathway, and $n = 2'000$ samples. All simulated patterns comprise five genes, and form different types of subgraphs in the network, that is either (i) a star-shaped subgraph, i.e. a full neighbourhood, (ii) an incomplete subgraph, in which only a specific fraction of genes within a neighbourhood are associated, or (iii) a

A. Data-simulation for significant pattern mining approaches

random subgraph of size five in the network. Simulating the data is a three-step procedure: in the first step, we simulate the background data, i.e. data for those genes that are not part of the induced pattern, in the second step, we expand the pattern indicator \mathbf{g}_t to obtain the data for features in the pattern, and in the third step, we plant the pattern in the network.

Simulation of data matrix

We simulate a data matrix $\mathbf{D} \in [0, 1]^{224 \times 2000}$ in the following way. For every gene $j = 1, \dots, d$ we simulate the maximal number of minor alleles uniformly from the range $[2, 40]$, and store the value in the j^{th} entry of a d -dimensional vector \mathbf{n}_{\max} . Next, for every sample i and every gene j , we draw x from $\mathcal{U}(0, \mathbf{n}_{\max}[j])$ and set $\mathbf{D}[i, j] = x/\mathbf{n}_{\max}[j]$.

Generation of features participating in pattern

We need to expand the encoding vector \mathbf{g}_t , to result in a $5 \times 2'000$ matrix $\mathbf{S} \in [0, 1]^{5 \times 2'000}$ with values between 0 and 1 that will be incorporated into the data matrix \mathbf{D} . In order to do so, we fix the threshold t at which the subgraph should be significant to 0.70. This means that, if \mathbf{S} is binarised at threshold t as described in Equation 4.2, and the encoding of the pattern is computed according to Equation 4.3, this results in the vector \mathbf{g}_t . In order to do so, we proceed as follows: First, we again randomly generate the maximal number of minor alleles that overlap with each of the five genes in the true subgraph uniformly from the range $[2, 40]$, resulting in the vector \mathbf{n}_{\max} . Next, we simulate the matrix \mathbf{S} such that, at threshold $t = 0.7$ the following holds: $\mathbf{g}_t[i] = 0, \forall i = 1, \dots, 2'000$. We do so by repeating the following two steps for every sample i , and every one of the five genes $j = 1, \dots, 5$:

1. simulate a random value x such that $x < \mathbf{n}_{\max}[j] \times 0.70$, and
2. set $\mathbf{S}[i, j] = \frac{x}{\mathbf{n}_{\max}[j]}$

In the end, we guarantee that for every sample i with $\mathbf{y}[i] = 1$, exactly one of the five genes exceeds the threshold $t = 0.70$. We do this to emphasise the genetic heterogeneity model, i.e. that there is one gene within the pattern that contains at least 70% minor alleles, but it is not important which one of those genes this is the case for. To achieve this, we repeat the following three steps for every sample i with $\mathbf{g}_t[i] = 1$:

1. chose the index j of the gene at random from $\{1, \dots, 5\}$, then
2. simulate a random value x such that $x \geq \mathbf{n}_{\max}[j] \times 0.70$, and
3. set $\mathbf{S}[i, j] = \frac{x}{\mathbf{n}_{\max}[j]}$

Remark A.1 (Count based data). *For the burden tests that use the count of minor alleles in a gene, as opposed to the ratio, the bare counts of minor alleles per gene are used, i.e. the values x in the above steps.*

Implantation into the network

In the graph $\mathcal{G} = (V, E)$, every node has a label ranging from 1 to 244, and the data matrix \mathbf{D} contains the measurements corresponding to the j^{th} node in the j^{th} row. When

A. Data-simulation for significant pattern mining approaches

implanting the data corresponding to a pattern, \mathbf{S} , into the data matrix, we sample nodes from the graph \mathcal{G} that obey the desired subgraph configuration, and replace the rows in \mathbf{D} corresponding to the subgraph with \mathbf{S} .

Simulation of data for the SiNIMin methods

This description of the simulation closely follows the one described in the supplement of our original publication, that is Gumpinger et al. [124].

We simulate data for a randomly generated network consisting of 75 nodes, that represent the genes, and 100 edges that represent interactions between genes. In each simulation, we generate data for 500 samples. For each of the 75 genes, which we index with $j = 1, \dots, 75$, we simulate a random number n_j of SNPs that are mapped to gene j , by drawing n_j from a uniform distribution over the range $[1, 10]$. Furthermore, we allow two genes to overlap with a probability of 30%, which implies that SNPs between the two genes are shared. The extend of overlap is limited to at most half of the SNPs per gene. This results in a varying number of features g per data set, where $g = \sum_{j=1}^{75} n_j$, where $75 \leq g \leq 750$, since $1 \leq n_j \leq 10$.

Simulation of data matrix

We simulate a binary data matrix $\mathbf{D} \in [0, 1]^{g \times 500}$ at random, where the probability of observing a 1 at any position equals 0.3, to roughly approximate the density in a genetic data set under a dominant encoding of minor alleles.

Generation of features participating in pattern

We simulate patterns as gene-segment interactions between two segments, such that their combined length equals six variants, and each individual segment consists of at least two, and at most four variants. Hence, we need to expand the pattern indicator vector \mathbf{g}_t to a 6×500 -dimensional binary matrix $\mathbf{S} \in [0, 1]^{6 \times 500}$ that will be inserted into the data matrix \mathbf{D} at the position of the SNPs that are part of the segment interaction. When expanding the encoding vector, we guarantee that for every sample i with $\mathbf{g}_t[i] = 1$ exactly one of the six SNPs gets assigned a 1. In this way, we create an association of genetic heterogeneity, that at the same time minimises the association signal of individual variants in the interaction.

Implantation into the network

To insert the pattern into the network, we randomly draw an edge from the network, and chose the lengths and starting positions of the SNPs in the interacting segments. We randomly split the matrix \mathbf{S} according to the randomly chosen segment lengths, and insert the resulting two parts into the data matrix.

B. Software packages

SiNIMin

The SiNIMin-software, together with instructions for installation and examples, is available at <https://github.com/BorgwardtLab/SiNIMin>. SiNIMin is implemented in C++, and we provide a command-line tool for its execution. Below, we show an example of how to invoke the method from the command line.

LISTING B.1: Calling SiNIMin method

```
#!/bin/bash
# Input flags of \sinimin method:
# -i: (string) file containing binary data with g rows (SNPs), n columns.
# -l: (string) file containing n binary labels.
# -c: (string) file containing n rows, each row indicates covariate class.
# -m: (string) file indicating mapping of SNPs to genes (nodes in graph).
# -e: (string) file containing one row per gene in the network.
# -s: (string) name of features, ordered as in data_file.
# -f: (double) target family-wise error rate.
# -o: (string) prefix of output files.
# -d: (integer) maximum length of segments to explore. If not specified, all
#     possible lengths are explored.
# -n: (integer) number of threads, default=1.
# -p: (integer) number of permutations. If this is set, Westfall-Young
#     permutations are invoked, default=1.

# Example of running SiNIMin approach.
./sinimin \
-i "${data_file}" \
-l "${labels_file}" \
-c "${covariate_file}" \
-m "${mapping_file}" \
-e "${edge_file}" \
-s "${feature_file}" \
-f 0.05 \
-o "${output_prefix}" \
```

MoPro embeddings

The MoPro embeddings are implemented in python, and source-code for their generation can be downloaded from <https://github.com/BorgwardtLab/MoProEmbeddings>. We provide two ipython notebooks detailing (i) the pre-processing of the data, as well as (ii) the

B. Software packages

generation of MoPro embeddings from the pre-processed data.

LISTING B.2: Computation of MoPro embeddings

```
import numpy as np
import pandas as pd

from MoProEmbeddings import basegraph
from MoProEmbeddings import utils
from MoProEmbeddings import path_weights
from MoProEmbeddings import features

# file containing the network and vertex features.
network_file = "./data/network.txt"
score_file = "./data/gene_features.txt"
labels_file = "./data/labels.txt"

# load the data.
edges = np.loadtxt(network_file, dtype=str)
scores = pd.read_csv(score_file, delim_whitespace=True, index_col=0)
labels = np.loadtxt(labels_file, dtype=int)

# create the basegraph object, log-transform node-features and add labels.
bg = basegraph.create(edges, scores)
bg = features.log_transform(bg, 'pvalue')
bg.vs['class_label'] = labels

# compute the higher-order weights.
weights = path_weights.ShortestPathWeights(
    bg, max_k=2, mode='max', weight='std')

# create the moment embeddings.
for m in ['mean', 'std', 'skew', 'kurt']:
    for k in [1, 2]:
        bg, _ = features.weighted_attr_kx(
            bg, 'log_pvalue', stat=m,
            k=k, weight='std', weight_dict=weights.shortest_path_weights
        )

# save.
moment_embeddings_pkl = './data/moment_embeddings.pkl'
utils.write_pickle(bg, moment_embeddings_pkl)

# propagation embeddings.
mp_data = data.MomProp(
    moment_embeddings_pkl, 'log_pvalue', 2, n_hops=2,
    moments=['mean', 'std', 'skew', 'kurt'], edge_weight='std',
    path_weight='max')
```

Acronyms

- CMH** Cochran-Mantel-Haenszel [test]. 49
- CNN** convolutional neural network. 157
- FDR** false discovery rate. 17
- FWER** family-wise error rate. 16, 36
- GCN** graph convolutional network. 157, 167
- GRE** gene-regulatory element. 162
- GWAS** genome-wide association study. 13
- LD** linkage disequilibrium. 14
- LMM** linear mixed model. 18
- MAF** minor allele frequency. 15
- NGS** next-generation sequencing. 14, 162
- PPI** protein-protein interaction. 3
- RBS** ribosome binding site. 163
- RWR** random walk with restart. 24
- SNP** single nucleotide polymorphism. 14, 15, 82

Glossary

Bonferroni correction Method to restrict the number of false positives during simultaneous testing of many statistical hypotheses. 16

common disease/common variant Hypothesis that common diseases arise due to multiple genetic factors that are common in the population. 14

epistasis Analysis of non-linear effects between two genetic variants on phenotype. 20

false discovery rate Expectation of the proportion of false discoveries among all discoveries. 17

family-wise error rate Probability to observe at least one false-positive association during statistical testing. 16, 36

gene segment A set of subsequent variants along the sequenced genome within a gene (where gene is defined as the entirety of its introns and exons). 85

gene size The number of genetic variants that map to a gene. 76

genetic heterogeneity Model of non-linear interaction between genetic variants (e.g. SNP). The same phenotype might be caused by different genetic variants for different individuals. 27, 60, 82

genetic segment A set of subsequent variants (e.g. SNPs, rare variants) along the sequenced genome. 85

hypothesis space The set of all possible patterns of specific type (synonym: search space). 31

linkage disequilibrium The non-random correlation between close-by SNPs caused by recombination patterns. 14

major allele The more prevalent allele at a genetic locus. 15

minor allele The less prevalent allele at a genetic locus. 15

missing heritability Phenomenon that SNPs found in GWAS only partially explain the phenotypic heritability. 20, 60

multiple hypothesis testing problem The creation of a large amount of false positive associations caused by the simultaneous testing of many hypotheses. 16, 36

Glossary

- oncogenes** An altered form of a gene that is involved in cell proliferation and apoptosis. An oncogene develops its tumourigenic potential upon structural alteration of the so-called proto-oncogenes. 124, 130
- passenger mutations** Mutations that have no effect on the tumourigenic potential of a cell. 126
- pattern** A finite set of discrete features. 30
- pattern indicator function** A function that indicates whether a pattern is present in a sample or not. 30
- rare variant** A genetic variant with minor allele frequencies below a threshold γ , where commonly $\gamma \leq 0.01$. 20
- search space** The set of all possible patterns of specific type (synonym: hypothesis space). 31
- single nucleotide polymorphism** A locus of common genetic variation in a population, abbreviated as SNP. 14
- support** Count of occurrence of a pattern in a database. 30
- tag SNP** A single nucleotide polymorphism (SNP) that represents a region of high LD in the genome. 14
- Tarone's procedure** Improvement of Bonferroni correction, based on identifying testable hypotheses in a data set. 37
- testability** The concept underlying Tarone's correction. A hypothesis tested with a discrete test is said to be testable, if, based on its marginal probabilities, the strongest possible association p -value is larger than a pre-defined threshold. 39
- transaction** Description of a sample as a set of features in pattern mining.. 30
- transaction database** Data structure that represents data in pattern mining. 30
- tumour suppressor genes** Genes that have a protective function by suppressing tumourigenicity. 124, 130
- type-I error** Error caused by falsely rejecting a true null hypothesis. 36
- type-II error** Error caused by falsely accepting the null hypothesis. 37

List of Figures

2.1	Representation of single nucleotide polymorphisms in a GWAS	15
2.2	Family-wise error rate for increasing number of tests	17
2.3	Example of a genome-wide association study	18
2.4	Integration of genetic data with biological networks.	22
3.1	Example of significant itemset mining	32
3.2	Minimum p -values for Pearson's χ^2 test and Fisher's exact test.	40
3.3	Illustration of testability at threshold δ	41
3.4	Example of pattern enumeration tree in itemset mining	45
4.1	Example of data used in tNeAT approach.	56
4.2	Binarisation of neighbourhood pattern at various thresholds.	59
4.3	Pattern enumeration for tNeAT approach	62
4.4	Example of subgraph configurations	69
4.5	tNeAT/tNeAT-WY: family-wise error rate estimates	72
4.6	tNeAT/tNeAT-WY: power analysis	73
4.7	Distribution of k -hop neighbourhood sizes in TAIR <i>A. thaliana</i> network	76
5.1	Overview of concepts for gene-segment interaction detection	84
5.2	Runtimes of methods for finding gene segment interactions	99
5.3	SiNIMin/SiNIMin-WY: family-wise error rate estimates	101
5.4	SiNIMin/SiNIMin-WY: power analysis	102
5.5	Effect of network modification on performance of SiNIMin	103
5.6	Number of SNPs mapping to <i>A. thaliana</i> genes	104
8.1	Schematic representation of moment embeddings	134
8.2	Propagation embedding and higher-order edge weights	135
8.3	Description of the InBio Map network	140
8.4	Description TCGA data sets	141
8.5	Illustration of cross-validation procedure for one random split of data	142
8.6	Dependence of results on cross-validation hyperparameters	148
8.7	Correlation between degree and moments	150
8.8	Degree of predicted cancer genes with MoPro embeddings	151

List of Tables

2.1	Contingency table counting minor alleles	19
3.1	Contingency table for CMH test	50
4.1	Contingency table for a pattern $\mathcal{N}_v^{k,t}$	60
4.2	<i>A. thaliana</i> phenotypes and their genomic inflation (tNeAT-WY)	75
4.3	Number of significant hits for <i>A. thaliana</i> with various methods	77
4.4	Analysis of significant tNeAT-WY hits in <i>A. thaliana</i>	79
5.1	Contingency table for a pattern $\mathcal{E}_{\tau_1 \tau_2}$	87
5.2	<i>A. thaliana</i> phenotypes and their genomic inflation (SiNIMin-WY)	105
5.3	Number of significant hits for <i>A. thaliana</i> with various methods	107
5.4	Novel hits for <i>A. thaliana</i> detected with SiNIMin-WY	108
5.5	Dimensions of migraine cohorts	109
5.6	Genomic inflation λ_{gc} for different covariates in migraine data	110
5.8	Number of significant hits for migraine data with various methods	112
5.9	Novel hits for migraine cohorts detected with SiNIMin-WY/edgeEpi-WY	112
8.1	Results of cancer gene classification	147
8.3	Ablation study for MoPro embeddings	149

Bibliography

1. Lohr, S. The age of big data. *New York Times* **11** (2012).
2. Reichlin, L. et al. Big data in economics: evolution or revolution? (2017).
3. Oswald, F. L. & Putka, D. J. Big data methods in the social sciences. *Current Opinion in Behavioral Sciences* **18**, 103–106 (2017).
4. Gagneur, J., Friedel, C., Heun, V., Zimmer, R. & Rost, B. *Bioinformatics advances biology and medicine by turning big data troves into knowledge in 50 Jahre Universitäts-Informatik in München* 33–45 (Springer, 2017).
5. Welch, T. F. & Widita, A. Big data in public transportation: a review of sources and methods. *Transport Reviews* **39**, 795–818 (2019).
6. Wang, Z., Wei, G., Zhan, Y. & Sun, Y. Big data in telecommunication operators: data, platform and practices. *Journal of Communications and Information Networks* **2**, 78–91 (2017).
7. Beam, A. L. & Kohane, I. S. Big data and machine learning in health care. *Jama* **319**, 1317–1318 (2018).
8. Leonelli, S. Philosophy of Biology: The challenges of big data biology. *Elife* **8**, e47381 (2019).
9. Jones, N. How machine learning could help to improve climate forecasts. *Nature* **548** (2017).
10. Bansal, S., Chowell, G., Simonsen, L., Vespignani, A. & Viboud, C. Big data for infectious disease surveillance and modeling. *The Journal of Infectious Diseases* **214**, S375–S379 (2016).
11. Flood, M. D., Jagadish, H., Raschid, L., et al. Big data challenges and opportunities in financial stability monitoring. *Banque de France, Financial Stability Review* **20**, 129–42 (2016).
12. Nyman, R. et al. News and narratives in financial systems: exploiting big data for systemic risk assessment (2018).
13. Lee, I. Big data: Dimensions, evolution, impacts, and challenges. *Business Horizons* **60**, 293–303 (2017).
14. Cirillo, D. & Valencia, A. Big data analytics for personalized medicine. *Current Opinion in Biotechnology* **58**, 161–167 (2019).
15. Tsai, C. J., Riaz, N. & Gomez, S. L. *Big Data in Cancer Research: Real-World Resources for Precision Oncology to Improve Cancer Care Delivery in Seminars in Radiation Oncology* **29** (2019), 306–310.

Bibliography

16. Yan, Y., Zhang, S. & Wu, F.-X. *Applications of graph theory in protein structure identification* in *Proteome Science* **9** (2011), S17.
17. Borgwardt, K. M. et al. Protein function prediction via graph kernels. *Bioinformatics* **21**, i47–i56 (2005).
18. Diss, G. Towards attaining a quantitative and mechanistic model of a cell. *Nature Reviews Molecular Cell Biology* (Feb. 2020).
19. The European Bioinformatics Institute (EMBL-EBI). *Protein-protein interaction networks* <https://www.ebi.ac.uk/training/online/course/network-analysis-protein-interaction-data-introduction/protein-protein-interaction-networks>.
20. Huang, J. K. et al. Systematic evaluation of molecular networks for discovery of disease genes. *Cell Systems* **6**, 484–495 (2018).
21. Oughtred, R. et al. The BioGRID interaction database: 2019 update. *Nucleic Acids Research* **47**, D529–D541 (2019).
22. BioGRID. *Statistics: BioGRID version 2.0.60* https://wiki.thebiogrid.org/doku.php/build_2.0.60.
23. BioGRID. *Statistics: BioGRID version 3.5.181* <https://wiki.thebiogrid.org/doku.php/statistics>.
24. Liu, C. et al. Computational network biology: Data, model, and applications. *Physics Reports* (2019).
25. Li, T. et al. A scored human protein–protein interaction network to catalyze genomic interpretation. *Nature Methods* **14**, 61 (2017).
26. Szklarczyk, D. et al. The STRING database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic Acids Research*, gkw937 (2016).
27. The European Bioinformatics Institute (EMBL-EBI). *Network Biology* <https://www.ebi.ac.uk/training/online/course/network-analysis-protein-interaction-data-introduction/networks-cell-biology-summary-0>.
28. McDonald, M.-L. N. et al. Beyond GWAS in COPD: probing the landscape between gene-set associations, genome-wide associations and protein-protein interaction networks. *Human Heredity* **78**, 131–139 (2014).
29. Fuchsberger, C. et al. The genetic architecture of type 2 diabetes. *Nature* **536**, 41–47 (2016).
30. Baranzini, S. E. et al. Network-based multiple sclerosis pathway analysis with GWAS data from 15,000 cases and 30,000 controls. *The American Journal of Human Genetics* **92**, 854–865 (2013).
31. Reyna, M. A. et al. Pathway and network analysis of more than 2500 whole cancer genomes. *Nature Communications* **11**, 1–17 (2020).
32. Horn, H. et al. NetSig: network-based discovery from cancer genomes. *Nature Methods* **15**, 61 (2018).

Bibliography

33. Kanehisa, M. & Goto, S. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research* **28**, 27–30 (2000).
34. Wang, J., Liu, Y. & Chen, T. Identification of key genes and pathways in Parkinson’s disease through integrated analysis. *Molecular Medicine Reports* **16**, 3769–3776 (2017).
35. Dinu, V., Miller, P. L. & Zhao, H. Evidence for association between multiple complement pathway genes and AMD. *Genetic Epidemiology: The Official Publication of the International Genetic Epidemiology Society* **31**, 224–237 (2007).
36. Yang, X. et al. Widespread expansion of protein interaction capabilities by alternative splicing. *Cell* **164**, 805–817 (2016).
37. Kelemen, O. et al. Function of alternative splicing. *Gene* **514**, 1–30 (2013).
38. Tseng, Y.-T. et al. *IIIDB: a database for isoform-isoform interactions and isoform network modules* in *BMC Genomics* **16** (2015), S10.
39. Pan, Q., Shai, O., Lee, L. J., Frey, B. J. & Blencowe, B. J. Deep surveying of alternative splicing complexity in the human transcriptome by high-throughput sequencing. *Nature Genetics* **40**, 1413 (2008).
40. Corominas, R. et al. Protein interaction network of alternatively spliced isoforms from brain links genetic risk factors for autism. *Nature Communications* **5**, 1–12 (2014).
41. Mani, R., Onge, R. P. S., Hartman, J. L., Giaever, G. & Roth, F. P. Defining genetic interaction. *Proceedings of the National Academy of Sciences* **105**, 3461–3466 (2008).
42. O’Neil, N. J., Bailey, M. L. & Hieter, P. Synthetic lethality and cancer. *Nature Reviews Genetics* **18**, 613–623 (2017).
43. Emmert-Streib, F., Dehmer, M. & Haihe-Kains, B. Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in Cell and Developmental Biology* **2**, 38 (2014).
44. Lee, W.-P. & Tzou, W.-S. Computational methods for discovering gene networks from expression data. *Briefings in Bioinformatics* **10**, 408–423 (2009).
45. Wedeen, V. J. et al. The geometric structure of the brain fiber pathways. *Science* **335**, 1628–1634 (2012).
46. Zalesky, A., Fornito, A. & Bullmore, E. On the use of correlation as a measure of network connectivity. *Neuroimage* **60**, 2096–2106 (2012).
47. Cheng, F. et al. Prediction of drug-target interactions and drug repositioning via network-based inference. *PLoS Computational Biology* **8** (2012).
48. Garca, E. D. V. et al. Disease networks and their contribution to disease understanding: A systematized review of their evolution, techniques and data sources. *Journal of Biomedical Informatics*, 103206–103206 (2019).
49. Nickel, M., Murphy, K., Tresp, V. & Gabrilovich, E. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE* **104**, 11–33 (2015).

Bibliography

50. Mohamed, S. K., Nounu, A. & Nováek, V. Biological applications of knowledge graph embedding models. *Briefings in Bioinformatics* (2020).
51. Muñoz, E., Nováek, V. & Vandenbussche, P.-Y. Facilitating prediction of adverse drug reactions by using knowledge graphs and multi-label learning models. *Briefings in Bioinformatics* **20**, 190–202 (2019).
52. Mohamed, S. K., Nounu, A. & Nováek, V. *Drug target discovery using knowledge graph embeddings* in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (2019), 11–18.
53. Alshahrani, M. et al. Neuro-symbolic representation learning on biological knowledge graphs. *Bioinformatics* **33**, 2723–2730 (2017).
54. Ideker, T. & Nussinov, R. Network approaches and applications in biology. *PLoS Computational Biology* **13** (2017).
55. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).
56. Barabasi, A.-L. & Oltvai, Z. N. Network biology: understanding the cell’s functional organization. *Nature Reviews Genetics* **5**, 101–113 (2004).
57. Carter, H., Hofree, M. & Ideker, T. Genotype to phenotype via network analysis. *Current Opinion in Genetics & Development* **23**, 611–621 (2013).
58. Goh, K.-I. & Choi, I.-G. Exploring the human diseasome: the human disease network. *Briefings in Functional Genomics* **11**, 533–542 (2012).
59. Ideker, T. & Krogan, N. J. Differential network biology. *Molecular Systems Biology* **8** (2012).
60. Subramanian, A. et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences* **102**, 15545–15550 (2005).
61. Cline, M. S. et al. Integration of biological networks and gene expression data using Cytoscape. *Nature Protocols* **2**, 2366 (2007).
62. Goldovsky, L., Cases, I., Enright, A. J. & Ouzounis, C. A. BioLayout Java. *Applied Bioinformatics* **4**, 71–74 (2005).
63. Jia, P., Zheng, S., Long, J., Zheng, W. & Zhao, Z. **dmGWAS**: dense module searching for genome-wide association studies in protein–protein interaction networks. *Bioinformatics* **27**, 95–102 (2011).
64. Wang, Q., Yu, H., Zhao, Z. & Jia, P. **EW_dmGWAS**: edge-weighted dense module search for genome-wide association studies and gene expression profiles. *Bioinformatics* **31**, 2591–2594 (2015).
65. Cowen, L., Ideker, T., Raphael, B. J. & Sharan, R. Network propagation: a universal amplifier of genetic associations. *Nature Reviews Genetics* **18**, 551 (2017).
66. Nelson, M. R. et al. The support of human genetic evidence for approved drug indications. *Nature Genetics* **47**, 856 (2015).

Bibliography

67. Claussnitzer, M. et al. A brief history of human disease genetics. *Nature* **577**, 179–189 (2020).
68. MacDonald, M. E. et al. The Huntington’s disease candidate region exhibits many different haplotypes. *Nature genetics* **1**, 99–103 (1992).
69. Kerem, B.-s. et al. Identification of the cystic fibrosis gene: genetic analysis. *Science* **245**, 1073–1080 (1989).
70. Hunter, D. J. Gene–environment interactions in human diseases. *Nature Reviews Genetics* **6**, 287–298 (2005).
71. International Human Genome Sequencing Consortium et al. Initial sequencing and analysis of the human genome. *Nature* **409**, 860–921 (2001).
72. International HapMap Consortium et al. The international HapMap project. *Nature* **426**, 789 (2003).
73. 1000 Genomes Project Consortium et al. A global reference for human genetic variation. *Nature* **526**, 68–74 (2015).
74. Klein, R. J. et al. Complement factor H polymorphism in age-related macular degeneration. *Science* **308**, 385–389 (2005).
75. Duerr, R. H. et al. A genome-wide association study identifies IL23R as an inflammatory bowel disease gene. *Science* **314**, 1461–1463 (2006).
76. Wellcome Trust Case Control Consortium et al. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature* **447**, 661 (2007).
77. Nelson, C. P. et al. Association analyses based on false discovery rate implicate new loci for coronary artery disease. *Nature Genetics* **49**, 1385 (2017).
78. Scott, R. A. et al. An expanded genome-wide association study of type 2 diabetes in Europeans. *Diabetes* **66**, 2888–2902 (2017).
79. Langefeld, C. D. et al. Transancestral mapping and genetic load in systemic lupus erythematosus. *Nature Communications* **8**, 16021 (2017).
80. Zengini, E. et al. Genome-wide analyses using UK Biobank data provide insights into the genetic architecture of osteoarthritis. *Nature Genetics* **50**, 549–558 (2018).
81. NHGRI. *DNA-Sequencing cost* <https://www.genome.gov/about-genomics/fact-sheets/DNA-Sequencing-Costs-Data>.
82. Papageorgiou, L. et al. Genomic big data hitting the storage bottleneck. *EMBnet journal* **24** (2018).
83. Bycroft, C. et al. The UK Biobank resource with deep phenotyping and genomic data. *Nature* **562**, 203–209 (2018).
84. Nagai, A. et al. Overview of the BioBank Japan Project: study design and profile. *Journal of Epidemiology* **27**, S2–S8 (2017).
85. Chen, Z. et al. China Kadoorie Biobank of 0.5 million people: survey methods, baseline characteristics and long-term follow-up. *International Journal of Epidemiology* **40**, 1652–1666 (2011).

Bibliography

86. Reich, D. E. & Lander, E. S. On the allelic spectrum of human disease. *TRENDS in Genetics* **17**, 502–510 (2001).
87. Bush, W. S. & Moore, J. H. Genome-wide association studies. *PLoS Computational Biology* **8** (2012).
88. Davey, J. W. et al. Genome-wide genetic marker discovery and genotyping using next-generation sequencing. *Nature Reviews Genetics* **12**, 499–510 (2011).
89. Siu, H., Zhu, Y., Jin, L. & Xiong, M. Implication of next-generation sequencing on association studies. *BMC Genomics* **12**, 322 (2011).
90. DiStefano, J. K. & Taverna, D. M. *Technological issues and experimental design of gene association studies in Disease Gene Identification* 3–16 (Springer, 2011).
91. International HapMap Consortium et al. A haplotype map of the human genome. *Nature* **437**, 1299 (2005).
92. Gumpinger, A. C., Roqueiro, D., Grimm, D. G. & Borgwardt, K. M. *Methods and tools in genome-wide association studies in Computational Cell Biology* 93–136 (Springer, 2018).
93. Abdi, H. Bonferroni and Sidák corrections for multiple comparisons. *Encyclopedia of Measurement and Statistics* **3**, 103–107 (2007).
94. Benjamini, Y. & Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society* **57**, 289–300 (1995).
95. Benjamini, Y. & Yekutieli, D. The control of the false discovery rate in multiple testing under dependency. *Annals of Statistics*, 1165–1188 (2001).
96. Storey, J. D. A direct approach to false discovery rates. *Journal of the Royal Statistical Society* **64**, 479–498 (2002).
97. Storey, J. D. & Tibshirani, R. Statistical significance for genomewide studies. *Proceedings of the National Academy of Sciences* **100**, 9440–9445 (2003).
98. Price, A. L. et al. Principal components analysis corrects for stratification in genome-wide association studies. *Nature Genetics* **38**, 904–909 (2006).
99. Power, R. A., Parkhill, J. & de Oliveira, T. Microbial genome-wide association studies: lessons from human GWAS. *Nature Reviews Genetics* **18**, 41 (2017).
100. Sul, J. H., Martin, L. S. & Eskin, E. Population structure in genetic studies: Confounding factors and mixed models. *PLoS Genetics* **14** (2018).
101. Kang, H. M. et al. Variance component model to account for sample structure in genome-wide association studies. *Nature Genetics* **42**, 348 (2010).
102. Zhou, X. & Stephens, M. Genome-wide efficient mixed-model analysis for association studies. *Nature Genetics* **44**, 821–824 (2012).
103. Lippert, C. et al. FaST linear mixed models for genome-wide association studies. *Nature Methods* **8**, 833 (2011).
104. Listgarten, J. et al. A powerful and efficient set test for genetic markers that handles confounders. *Bioinformatics* **29**, 1526–1533 (2013).

Bibliography

105. Widmer, C. et al. Further improvements to linear mixed models for genome-wide association studies. *Scientific Reports* **4**, 1–13 (2014).
106. Fahrmeir, L., Kneib, T., Lang, S. & Marx, B. *Regression* (Springer, 2007).
107. Pearson, K. X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **50**, 157–175 (1900).
108. Cochran, W. G. Some methods for strengthening the common χ^2 tests. *Biometrics* **10**, 417–451 (1954).
109. Mantel, N. & Haenszel, W. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the National Cancer Institute* **22**, 719–748 (1959).
110. Buniello, A. et al. The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019. *Nucleic Acids Research* **47**, D1005–D1012 (2019).
111. Tam, V. et al. Benefits and limitations of genome-wide association studies. *Nature Reviews Genetics* **20**, 467–484 (2019).
112. Edwards, S. L., Beesley, J., French, J. D. & Dunning, A. M. Beyond GWASs: illuminating the dark road from association to function. *The American Journal of Human Genetics* **93**, 779–797 (2013).
113. Manolio, T. A. et al. Finding the missing heritability of complex diseases. *Nature* **461**, 747–753 (2009).
114. Zuk, O. et al. Searching for missing heritability: designing rare variant association studies. *Proceedings of the National Academy of Sciences* **111**, E455–E464 (2014).
115. Gibson, G. Rare and common variants: twenty arguments. *Nature Reviews Genetics* **13**, 135–145 (2012).
116. Zuk, O., Hechter, E., Sunyaev, S. R. & Lander, E. S. The mystery of missing heritability: Genetic interactions create phantom heritability. *Proceedings of the National Academy of Sciences* **109**, 1193–1198 (2012).
117. Bansal, V., Libiger, O., Torkamani, A. & Schork, N. J. Statistical analysis strategies for association studies involving rare variants. *Nature Reviews Genetics* **11**, 773–785 (2010).
118. Povysil, G. et al. Rare-variant collapsing analyses for complex traits: guidelines and applications. *Nature Reviews Genetics* **20**, 747–759 (2019).
119. Lee, S., Abecasis, G. R., Boehnke, M. & Lin, X. Rare-variant association analysis: study designs and statistical tests. *The American Journal of Human Genetics* **95**, 5–23 (2014).
120. Cordell, H. J. Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Human Molecular Genetics* **11**, 2463–2468 (2002).
121. Phillips, P. C. Epistasis: the essential role of gene interactions in the structure and evolution of genetic systems. *Nature Reviews Genetics* **9**, 855–867 (2008).

Bibliography

122. Mackay, T. F. & Moore, J. H. Why epistasis is important for tackling complex human disease genetics. *Genome Medicine* **6**, 42 (2014).
123. Cordell, H. J. Detecting gene–gene interactions that underlie human diseases. *Nature Reviews Genetics* **10**, 392–404 (2009).
124. Gumpinger, A. C., Rieck, B., Grimm, D. G., International Headache Genetics Consortium & Borgwardt, K. *Network-guided search for genetic heterogeneity between gene pairs* In press at OUP Bioinformatics.
125. Pedroso, I. & Breen, G. Gene set analysis and network analysis for genome-wide association studies. *Cold Spring Harbor protocols* **2011** (2011).
126. Liu, J. Z. et al. A versatile gene-based test for genome-wide association studies. *The American Journal of Human Genetics* **87**, 139–145 (2010).
127. Lamparter, D., Marbach, D., Rueedi, R., Kutalik, Z. & Bergmann, S. Fast and rigorous computation of gene and pathway scores from SNP-based summary statistics. *PLoS Computational Biology* **12**, e1004714 (2016).
128. Sedeño-Cortés, A. E. & Pavlidis, P. Pitfalls in the application of gene-set analysis to genetics studies. *Trends in Genetics* **30**, 513–514 (2014).
129. Ballard, D. H., Cho, J. & Zhao, H. Comparisons of multi-marker association methods to detect association between a candidate region and disease. *Genetic Epidemiology* **34**, 201–212 (2010).
130. Fisher, R. A. *Statistical methods for research workers* in *Breakthroughs in Statistics* 66–70 (Springer, 1992).
131. Wu, M. C. et al. Rare-variant association testing for sequencing data with the sequence kernel association test. *The American Journal of Human Genetics* **89**, 82–93 (2011).
132. Lee, S. et al. Optimal unified approach for rare-variant association testing with application to small-sample case-control whole-exome sequencing studies. *The American Journal of Human Genetics* **91**, 224–237 (2012).
133. Morgenthaler, S. & Thilly, W. G. A strategy to discover genes that carry multi-allelic or mono-allelic risk for common diseases: a cohort allelic sums test (CAST). *Mutation Research/Fundamental and Molecular Mechanisms of Mutagenesis* **615**, 28–56 (2007).
134. Li, B. & Leal, S. M. Methods for detecting associations with rare variants for common diseases: application to analysis of sequence data. *The American Journal of Human Genetics* **83**, 311–321 (2008).
135. Madsen, B. E. & Browning, S. R. A groupwise association test for rare mutations using a weighted sum statistic. *PLoS Genetics* **5** (2009).
136. Álvarez-Miranda, E., Ljubi, I. & Mutzel, P. *The maximum weight connected subgraph problem* in *Facets of Combinatorial Optimization* 245–270 (Springer, 2013).
137. Ideker, T., Ozier, O., Schwikowski, B. & Siegel, A. F. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics* **18**, S233–S240 (2002).

Bibliography

138. Dittrich, M. T., Klau, G. W., Rosenwald, A., Dandekar, T. & Müller, T. Identifying functional modules in protein–protein interaction networks: an integrated exact approach. *Bioinformatics* **24**, i223–i231 (2008).
139. Beisser, D., Klau, G. W., Dandekar, T., Müller, T. & Dittrich, M. T. BioNet: an R-Package for the functional analysis of biological networks. *Bioinformatics* **26**, 1129–1130 (2010).
140. Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C. *Introduction to algorithms* (MIT press, 2009).
141. Shannon, P. et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research* **13**, 2498–2504 (2003).
142. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. Optimization by simulated annealing. *Science* **220**, 671–680 (1983).
143. Vandin, F., Upfal, E. & Raphael, B. J. Algorithms for detecting significantly mutated pathways in cancer. *Journal of Computational Biology* **18**, 507–522 (2011).
144. Vandin, F., Clay, P., Upfal, E. & Raphael, B. J. *Discovery of mutated subnetworks associated with clinical data in cancer* in *Biocomputing 2012* 55–66 (World Scientific, 2012).
145. Leiserson, M. D. et al. Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. *Nature Genetics* **47**, 106 (2015).
146. Reyna, M. A., Leiserson, M. D. & Raphael, B. J. Hierarchical HotNet: identifying hierarchies of altered subnetworks. *Bioinformatics* **34**, i972–i980 (2018).
147. Hammerschlag, A. R. et al. Genome-wide association analysis of insomnia complaints identifies risk genes and genetic overlap with psychiatric and metabolic traits. *Nature Genetics* **49**, 1584 (2017).
148. Hristov, B. H., Chazelle, B. & Singh, M. A guided network propagation approach to identify disease genes that combines prior and new information. *arXiv preprint arXiv:2001.06135* (2020).
149. Ruffalo, M., Koyutürk, M. & Sharan, R. Network-based integration of disparate omic data to identify ‘silent players’ in cancer. *PLoS Computational Biology* **11** (2015).
150. Yuan, M. & Lin, Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society* **68**, 49–67 (2006).
151. Meier, L., Van De Geer, S. & Bühlmann, P. The group lasso for logistic regression. *Journal of the Royal Statistical Society* **70**, 53–71 (2008).
152. Jacob, L., Obozinski, G. & Vert, J.-P. *Group lasso with overlap and graph lasso* in *Proceedings of the 26th Annual International Conference on Machine Learning* (2009), 433–440.
153. Liu, J., Huang, J., Ma, S. & Wang, K. Incorporating group correlations in genome-wide association studies using smoothed group lasso. *Biostatistics* **14**, 205–219 (2013).

Bibliography

154. Li, C. & Li, H. Network-constrained regularization and variable selection for analysis of genomic data. *Bioinformatics* **24**, 1175–1182 (2008).
155. Azencott, C.-A., Grimm, D., Sugiyama, M., Kawahara, Y. & Borgwardt, K. M. Efficient network-guided multi-locus association mapping with graph cuts. *Bioinformatics* **29**, i171–i179 (2013).
156. McClellan, J. & King, M.-C. Genetic heterogeneity in human disease. *Cell* **141**, 210–217 (2010).
157. Burrell, R. A., McGranahan, N., Bartek, J. & Swanton, C. The causes and consequences of genetic heterogeneity in cancer evolution. *Nature* **501**, 338–345 (2013).
158. Terada, A., Okada-Hatakeyama, M., Tsuda, K. & Sese, J. Statistical significance of combinatorial regulations. *Proceedings of the National Academy of Sciences* **110**, 12996–13001 (2013).
159. Minato, S.-i., Uno, T., Tsuda, K., Terada, A. & Sese, J. *A fast method of statistical assessment for combinatorial hypotheses based on frequent itemset enumeration in Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2014), 422–436.
160. Terada, A., Tsuda, K. & Sese, J. *Fast Westfall-Young permutation procedure for combinatorial regulation discovery in 2013 IEEE International Conference on Bioinformatics and Biomedicine* (2013), 153–158.
161. Llinares-López, F., Sugiyama, M., Papaxanthos, L. & Borgwardt, K. *Fast and memory-efficient significant pattern mining via permutation testing in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), 725–734.
162. Sugiyama, M., López, F. L., Kasenburg, N. & Borgwardt, K. M. *Significant subgraph mining with multiple testing correction in Proceedings of the 2015 SIAM International Conference on Data Mining* (2015), 37–45.
163. Papaxanthos, L., Llinares-López, F., Bodenham, D. & Borgwardt, K. *Finding significant combinations of features in the presence of categorical covariates in Advances in Neural Information Processing Systems* (2016), 2279–2287.
164. Llinares-López, F. et al. Genome-wide detection of intervals of genetic heterogeneity associated with complex traits. *Bioinformatics* **31**, i240–i249 (2015).
165. Llinares-López, F. et al. Genome-wide genetic heterogeneity discovery with categorical covariates. *Bioinformatics* **33**, 1820–1828 (2017).
166. Llinares-López, F. & Borgwardt, K. Machine Learning for Biomarker Discovery: Significant Pattern Mining. *Analyzing Network Data in Biology and Medicine: An Interdisciplinary Textbook for Biological, Medical and Computational Scientists*, 313 (2019).
167. Borgelt, C. Frequent item set mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **2**, 437–456 (2012).
168. Llinares López, F. *Significant Pattern Mining for Biomarker Discovery* PhD thesis (ETH Zurich, 2018).

Bibliography

169. Agrawal, R., Srikant, R., et al. *Fast algorithms for mining association rules in Proceedings of the 20th International Conference of Very Large Data Bases, VLDB* **1215** (1994), 487–499.
170. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. I., et al. Fast discovery of association rules. *Advances in Knowledge Discovery and Data Mining* **12**, 307–328 (1996).
171. Ogihara, Z. P., Zaki, M., Parthasarathy, S., Ogihara, M. & Li, W. *New algorithms for fast discovery of association rules in In 3rd International Conference on Knowledge Discovery and Data Mining* (1997).
172. Han, J., Pei, J. & Yin, Y. Mining frequent patterns without candidate generation. *ACM Sigmod Record* **29**, 1–12 (2000).
173. Grahne, G. & Zhu, J. *Efficiently using prefix-trees in mining frequent itemsets. in FIMI* **90** (2003), 65.
174. Zaki, M. J. & Gouda, K. *Fast vertical mining using diffsets in Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2003), 326–335.
175. Schmidt-Thieme, L. *Algorithmic Features of Eclat. in FIMI* (2004).
176. Uno, T., Asai, T., Uchida, Y. & Arimura, H. *LCM: An Efficient Algorithm for Enumerating Frequent Closed Item Sets. in Fimi* **90** (2003).
177. Uno, T., Kiyomi, M., Arimura, H., et al. *LCM ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets in Fimi* **126** (2004).
178. Uno, T., Kiyomi, M. & Arimura, H. *LCM ver. 3: collaboration of array, bitmap and prefix tree for frequent itemset mining in Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations* (2005), 77–86.
179. Camilli, G. The relationship between Fisher’s exact test and Pearson’s chi-square test: A Bayesian perspective. *Psychometrika* **60**, 305–312 (1995).
180. Fisher, R. A. On the interpretation of χ^2 from contingency tables, and the calculation of P. *Journal of the Royal Statistical Society* **85**, 87–94 (1922).
181. Webb, G. I. Discovering significant patterns. *Machine Learning* **68**, 1–33 (2007).
182. Bonferroni, C. E., Bonferroni, C. & Bonferroni, C. Teoria statistica delle classi e calcolo delle probabilità. (1936).
183. Tarone, R. E. A modified Bonferroni method for discrete data. *Biometrics*, 515–522 (1990).
184. Meinshausen, N., Maathuis, M. H., Bühlmann, P., et al. Asymptotic optimality of the Westfall–Young permutation procedure for multiple testing under dependence. *The Annals of Statistics* **39**, 3369–3391 (2011).
185. Westfall, P. H., Young, S. S., et al. *Resampling-based multiple testing: Examples and methods for p-value adjustment* (John Wiley & Sons, 1993).

Bibliography

186. Bock, C. et al. Association mapping in biomedical time series via statistically significant shapelet mining. *Bioinformatics* **34**, i438–i446 (2018).
187. Shervashidze, N., Schweitzer, P., Leeuwen, E. J. v., Mehlhorn, K. & Borgwardt, K. M. Weisfeiler-Lehman graph kernels. *Journal of Machine Learning Research* **12**, 2539–2561 (2011).
188. Kipf, T. N. & Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
189. Kanehisa, M., Goto, S., Furumichi, M., Tanabe, M. & Hirakawa, M. KEGG for representation and analysis of molecular networks involving diseases and drugs. *Nucleic Acids Research* **38**, D355–D360 (2010).
190. Kanehisa, M., Furumichi, M., Tanabe, M., Sato, Y. & Morishima, K. KEGG: new perspectives on genomes, pathways, diseases and drugs. *Nucleic Acids Research* **45**, D353–D361 (2016).
191. Atwell, S. et al. Genome-wide association study of 107 phenotypes in Arabidopsis thaliana inbred lines. *Nature* **465**, 627–631 (2010).
192. Grimm, D. G. et al. easyGWAS: A cloud-based platform for comparing the results of genome-wide association studies. *The Plant Cell* **29**, 5–19 (2017).
193. Seren, Ü. et al. AraPheno: a public database for Arabidopsis thaliana phenotypes. *Nucleic Acids Research*, gkw986 (2016).
194. Togninalli, M. et al. AraPheno and the AraGWAS Catalog 2020: a major database update including RNA-Seq and knockout mutation data for Arabidopsis thaliana. *Nucleic Acids Research* **48**, D1063–D1068 (2020).
195. Berardini, T. Z. et al. The Arabidopsis information resource: making and mining the gold standard annotated reference plant genome. *Genesis* **53**, 474–485 (2015).
196. The Arabidopsis Information Resource (TAIR). *Protein-protein interaction network* <https://www.arabidopsis.org/portals/proteome/proteinInteract.jsp>.
197. Krishnakumar, V. et al. Araport: the Arabidopsis information portal. *Nucleic Acids Research* **43**, D1003–D1009 (2015).
198. The Arabidopsis Information Resource (TAIR). *Araport11 genome annotation* https://www.arabidopsis.org/download/index-auto.jsp?dir=%2Fdownload_files%2FGenes%2FAraport11_genome_release.
199. Interactome, Arabidopsis Interactome Mapping Consortium. Evidence for network evolution in an Arabidopsis interactome map. *Science* **333**, 601–607 (2011).
200. Delker, C. et al. Jasmonate biosynthesis in Arabidopsis thaliana—enzymes, products, regulation. *Plant Biology* **8**, 297–306 (2006).
201. Lawit, S. J., OGrady, K., Gurley, W. B. & Czarnecka-Verner, E. Yeast two-hybrid map of Arabidopsis TFIID. *Plant Molecular Biology* **64**, 73–87 (2007).

Bibliography

202. Guyuron, B. et al. Electron microscopic and proteomic comparison of terminal branches of the trigeminal nerve in patients with and without migraine headaches. *Plastic and Reconstructive Surgery* **134**, 796e (2014).
203. Bond, A. M., Bhalala, O. G. & Kessler, J. A. The dynamic role of bone morphogenetic proteins in neural stem cell fate and maturation. *Developmental Neurobiology* **72**, 1068–1084 (2012).
204. Fowkes, J. & Sutton, C. *A Bayesian network model for interesting itemsets in Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2016), 410–425.
205. Mampaey, M., Tatti, N. & Vreeken, J. *Tell me what I need to know: succinctly summarizing data with itemsets in Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2011), 573–581.
206. Vreeken, J., Van Leeuwen, M. & Siebes, A. Krimp: mining itemsets that compress. *Data Mining and Knowledge Discovery* **23**, 169–214 (2011).
207. Yoshizoe, K., Terada, A. & Tsuda, K. MP-LAMP: parallel detection of statistically significant multi-loci markers on cloud platforms. *Bioinformatics* **34**, 3047–3049 (2018).
208. Ponte-Fernández, C., González-Domnguez, J. & Martn, M. J. Fast search of third-order epistatic interactions on cpu and gpu clusters. *The International Journal of High Performance Computing Applications* **34**, 20–29 (2020).
209. Terada, A., Kim, H. & Sese, J. *High-speed Westfall-Young permutation procedure for genome-wide association studies in Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics* (2015), 17–26.
210. Sugiyama, M. & Borgwardt, K. *Finding statistically significant interactions between continuous features in Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence* (2019).
211. Tatti, N. *Itemsets for real-valued datasets in 2013 IEEE 13th International Conference on Data Mining* (2013), 717–726.
212. Woolf, B. The log likelihood ratio test (the G-test). *Annals of Human Genetics* **21**, 397–409 (1957).
213. World Health Organisation. *Cancer* <https://www.who.int/en/news-room/fact-sheets/detail/cancer>.
214. World Health Organisation. *Cancer* <https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death>.
215. Siegel, R. L., Miller, K. D. & Jemal, A. Cancer statistics, 2019. *CA: A Cancer Journal for Clinicians* **69**, 7–34 (2019).
216. NIH: National Cancer Institute. *What is Cancer* <https://www.cancer.gov/about-cancer/understanding/what-is-cancer>.
217. Croce, C. M. Oncogenes and cancer. *New England Journal of Medicine* **358**, 502–511 (2008).

Bibliography

218. Hanahan, D. & Weinberg, R. A. The hallmarks of cancer. *Cell* **100**, 57–70 (2000).
219. Sherr, C. J. Principles of tumor suppression. *Cell* **116**, 235–246 (2004).
220. Knudson, A. G. Two genetic hits (more or less) to cancer. *Nature Reviews Cancer* **1**, 157–162 (2001).
221. Varley, J. Germline TP53 mutations and Li-Fraumeni syndrome. *Human Mutation* **21**, 313–320 (2003).
222. Candido-dos-Reis, F. J. et al. Germline mutation in BRCA1 or BRCA2 and ten-year survival for women diagnosed with epithelial ovarian cancer. *Clinical Cancer Research* **21**, 652–657 (2015).
223. Anand, P. et al. Cancer is a preventable disease that requires major lifestyle changes. *Pharmaceutical Research* **25**, 2097–2116 (2008).
224. Zhang, Y.-B. et al. Combined lifestyle factors, incident cancer, and cancer mortality: a systematic review and meta-analysis of prospective cohort studies. *British Journal of Cancer* **122**, 1085–1093 (2020).
225. Fane, M. & Weeraratna, A. T. How the ageing microenvironment influences tumour progression. *Nature Reviews Cancer*, 1–18 (2019).
226. Aunan, J. R., Cho, W. C. & Søreide, K. The biology of aging and cancer: a brief overview of shared and divergent molecular hallmarks. *Aging and Disease* **8**, 628 (2017).
227. De Magalhães, J. P. How ageing processes influence cancer. *Nature Reviews Cancer* **13**, 357–365 (2013).
228. Network, C. G. A. R. et al. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature* **455**, 1061 (2008).
229. Hofree, M. et al. Challenges in identifying cancer genes by analysis of exome sequencing data. *Nature Communications* **7**, 1–9 (2016).
230. Lawrence, M. S. et al. Mutational heterogeneity in cancer and the search for new cancer-associated genes. *Nature* **499**, 214–218 (2013).
231. Pon, J. R. & Marra, M. A. Driver and passenger mutations in cancer. *Annual Review of Pathology: Mechanisms of Disease* **10**, 25–50 (2015).
232. Creixell, P. et al. Pathway and network analysis of cancer genomes. *Nature Methods* **12**, 615 (2015).
233. Lawrence, M. S. et al. Discovery and saturation analysis of cancer genes across 21 tumour types. *Nature* **505**, 495–501 (2014).
234. Gonzalez-Perez, A. & Lopez-Bigas, N. Functional impact bias reveals cancer drivers. *Nucleic Acids Research* **40**, e169–e169 (2012).
235. Taylor, B. S. et al. Functional copy-number alterations in cancer. *PloS One* **3** (2008).
236. Sondka, Z. et al. The COSMIC Cancer Gene Census: describing genetic dysfunction across all human cancers. *Nature Reviews Cancer* **18**, 696–705 (2018).
237. Paz-Yaacov, N. et al. Elevated RNA editing activity is a major contributor to transcriptomic diversity in tumors. *Cell Reports* **13**, 267–276 (2015).

Bibliography

238. Zack, T. I. et al. Pan-cancer patterns of somatic copy number alteration. *Nature Genetics* **45**, 1134–1140 (2013).
239. Bailey, M. H. et al. Comprehensive characterization of cancer driver genes and mutations. *Cell* **173**, 371–385 (2018).
240. Gao, Q. et al. Driver fusions and their implications in the development and treatment of human cancers. *Cell Reports* **23**, 227–238 (2018).
241. Hanahan, D. & Weinberg, R. A. Hallmarks of cancer: the next generation. *Cell* **144**, 646–674 (2011).
242. Cheng, F., Zhao, J. & Zhao, Z. Advances in computational approaches for prioritizing driver mutations and significantly mutated genes in cancer genomes. *Briefings in Bioinformatics* **17**, 642–656 (2016).
243. Dees, N. D. et al. MuSiC: identifying mutational significance in cancer genomes. *Genome Research* **22**, 1589–1598 (2012).
244. Tamborero, D., Gonzalez-Perez, A. & Lopez-Bigas, N. OncodriveCLUST: exploiting the positional clustering of somatic mutations to identify cancer genes. *Bioinformatics* **29**, 2238–2244 (2013).
245. Ng, P. C. & Henikoff, S. SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Research* **31**, 3812–3814 (2003).
246. Adzhubei, I. A. et al. A method and server for predicting damaging missense mutations. *Nature Methods* **7**, 248–249 (2010).
247. Berman, H. M. et al. The protein data bank. *Nucleic Acids Research* **28**, 235–242 (2000).
248. Reimand, J. & Bader, G. D. Systematic analysis of somatic mutations in phosphorylation signaling predicts novel cancer drivers. *Molecular Systems Biology* **9** (2013).
249. Ryslik, G. A., Cheng, Y., Cheung, K.-H., Modis, Y. & Zhao, H. Utilizing protein structure to identify non-random somatic mutations. *BMC Bioinformatics* **14**, 190 (2013).
250. Vuong, H., Cheng, F., Lin, C.-C. & Zhao, Z. Functional consequences of somatic mutations in cancer using protein pocket-based prioritization approach. *Genome Medicine* **6**, 81 (2014).
251. Sanchez-Garcia, F. et al. Integration of genomic data enables selective discovery of breast cancer drivers. *Cell* **159**, 1461–1475 (2014).
252. Tamborero, D. et al. Comprehensive identification of mutational cancer driver genes across 12 tumor types. *Scientific Reports* **3**, 2650 (2013).
253. Tokheim, C. J., Papadopoulos, N., Kinzler, K. W., Vogelstein, B. & Karchin, R. Evaluating the evaluation of cancer driver genes. *Proceedings of the National Academy of Sciences* **113**, 14330–14335 (2016).
254. Vogelstein, B. et al. Cancer genome landscapes. *Science* **339**, 1546–1558 (2013).

Bibliography

255. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. *Neural message passing for quantum chemistry* in *Proceedings of the 34th International Conference on Machine Learning-Volume 70* (2017), 1263–1272.
256. Gumpinger, A. C., Lage, K., Horn, H. & Borgwardt, K. *Prediction of cancer driver genes through network-based moment propagation of mutation scores* In press at OUP Bioinformatics.
257. Lage, K. et al. A human phenome-interactome network of protein complexes implicated in genetic disorders. *Nature Biotechnology* **25**, 309–316 (2007).
258. Lohr, J. G. et al. Discovery and prioritization of somatic mutations in diffuse large B-cell lymphoma (DLBCL) by whole-exome sequencing. *Proceedings of the National Academy of Sciences* **109**, 3879–3884 (2012).
259. Ben-Hur, A. & Noble, W. S. *Choosing negative examples for the prediction of protein-protein interactions* in *BMC Bioinformatics* **7** (2006), S2.
260. Pedregosa, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12**, 2825–2830 (2011).
261. Agnihotri, S. et al. A GATA4-regulated tumor suppressor network represses formation of malignant human astrocytomas. *Journal of Experimental Medicine* **208**, 689–702 (2011).
262. Kijewska, M. et al. Using an in-vivo syngeneic spontaneous metastasis model identifies ID2 as a promoter of breast cancer colonisation in the brain. *Breast Cancer Research* **21**, 4 (2019).
263. Fittall, M. W. et al. Recurrent rearrangements of FOS and FOSB define osteoblastoma. *Nature Communications* **9**, 1–6 (2018).
264. Rautela, J. et al. Therapeutic blockade of activin-A improves NK cell function and antitumor immunity. *Science Signaling* **12**, eaat7527 (2019).
265. Kalli, M. et al. Activin A signaling regulates IL13R α 2 expression to promote breast cancer metastasis. *Frontiers in Oncology* **9**, 32 (2019).
266. Su, G. H. et al. ACVR1B (ALK4, activin receptor type 1B) gene mutations in pancreatic carcinoma. *Proceedings of the National Academy of Sciences* **98**, 3254–3257 (2001).
267. Wang, J., Chun, H. J., Wong, W., Spencer, D. M. & Lenardo, M. J. Caspase-10 is an initiator caspase in death receptor signaling. *Proceedings of the National Academy of Sciences* **98**, 13884–13888 (2001).
268. Zhang, Y.-L., Wang, R.-C., Cheng, K., Ring, B. Z. & Su, L. Roles of Rap1 signaling in tumor cell migration and invasion. *Cancer Biology & Medicine* **14**, 90 (2017).
269. Zhou, X. et al. Myosin light-chain kinase contributes to the proliferation and migration of breast cancer cells through cross-talk with activated ERK1/2. *Cancer Letters* **270**, 312–327 (2008).
270. Avizienyte, E., Brunton, V. G., Fincham, V. J. & Frame, M. C. The SRC-induced mesenchymal state in late-stage colon cancer cells. *Cells Tissues Organs* **179**, 73–80 (2005).

Bibliography

271. Carrino, M. et al. Prosurvival autophagy is regulated by protein kinase CK1 alpha in multiple myeloma. *Cell Death Discovery* **5**, 1–14 (2019).
272. Cheong, J. K. et al. Casein kinase 1 α -dependent feedback loop controls autophagy in RAS-driven cancers. *The Journal of Clinical Investigation* **125**, 1401–1418 (2015).
273. Gouravani, M. et al. The NLRP3 inflammasome: a therapeutic target for inflammation-associated cancers. *Expert Review of Clinical Immunology* (2020).
274. Krajewska, M. et al. Tumor-associated alterations in caspase-14 expression in epithelial malignancies. *Clinical Cancer Research* **11**, 5462–5471 (2005).
275. Schade, A. E., Fischer, M. & DeCaprio, J. A. RB, p130 and p107 differentially repress G1/S and G2/M genes after p53 activation. *Nucleic Acids Research* **47**, 11197–11208 (2019).
276. Seibold, M. et al. RAL GTPases mediate multiple myeloma cell survival and are activated independently of oncogenic RAS. *Haematologica*, haematol-2019 (2019).
277. Wang, Z. et al. Nuclear receptor HNF4 α performs a tumor suppressor function in prostate cancer via its induction of p21-driven cellular senescence. *Oncogene* **39**, 1572–1589 (2020).
278. International Cancer Genome Consortium et al. International network of cancer genome projects. *Nature* **464**, 993 (2010).
279. Villani, C. *Optimal transport: old and new* (Springer Science & Business Media, 2008).
280. Togninalli, M., Ghisu, E., Llinares-López, F., Rieck, B. & Borgwardt, K. Wasserstein Weisfeiler-Lehman graph kernels in *Advances in Neural Information Processing Systems* (2019), 6436–6446.
281. Wu, Z. et al. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).
282. Hamilton, W., Ying, Z. & Leskovec, J. Inductive representation learning on large graphs in *Advances in Neural Information Processing Systems* (2017), 1024–1034.
283. National Cancer Institute. *Targeted cancer therapies* <https://www.cancer.gov/about-cancer/treatment/types/targeted-therapies/targeted-therapies-fact-sheet>.
284. Secrier, M. & Schneider, R. Visualizing time-related data in biology, a review. *Briefings in Bioinformatics* **15**, 771–782 (2014).
285. Hyland, S. L. et al. Early prediction of circulatory failure in the intensive care unit using machine learning. *Nature Medicine* **26**, 364–373 (2020).
286. Keogh, E., Lin, J., Fu, A. W. & VanHerle, H. Finding unusual medical time-series subsequences: Algorithms and applications. *IEEE Transactions on Information Technology in Biomedicine* **10**, 429–439 (2006).
287. Schroeder, T. Long-term single-cell imaging of mammalian stem cells. *Nature Methods* **8**, S30–S35 (2011).

Bibliography

288. Höllerer, S. et al. Large-scale DNA-based phenotypic recording and deep learning enable highly accurate sequence-function mapping. *bioRxiv* (2020).
289. Shine, J. & Dalgarno, L. The 3-terminal sequence of Escherichia coli 16S ribosomal RNA: complementarity to nonsense triplets and ribosome binding sites. *Proceedings of the National Academy of Sciences* **71**, 1342–1346 (1974).
290. Kriege, N. M., Johansson, F. D. & Morris, C. A survey on graph kernels. *Applied Network Science* **5**, 1–42 (2020).
291. Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R. & Borgwardt, K. M. Graph kernels. *Journal of Machine Learning Research* **11**, 1201–1242 (2010).
292. Borgwardt, K. M. & Kriegel, H.-P. *Shortest-path kernels on graphs* in *IEEE International Conference on Data Mining (ICDM)* (2005).
293. Morris, C., Kriege, N. M., Kersting, K. & Mutzel, P. *Faster kernels for graphs with continuous attributes via hashing* in *IEEE International Conference on Data Mining (ICDM)* (2016), 1095–1100.
294. Rieck, B., Bock, C. & Borgwardt, K. *A Persistent Weisfeiler–Lehman Procedure for Graph Classification* in *Proceedings of the 36th International Conference on Machine Learning* (eds Chaudhuri, K. & Salakhutdinov, R.) **97** (PMLR, Long Beach, California, USA, June 2019), 5448–5458.
295. Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K. & Borgwardt, K. *Efficient graphlet kernels for large graph comparison* in *Artificial Intelligence and Statistics* (2009), 488–495.
296. Chen, B., Li, M., Wang, J. & Wu, F.-X. Disease gene identification by using graph kernels and Markov random fields. *Science China Life Sciences* **57**, 1054–1063 (2014).
297. Hamilton, W. L., Ying, R. & Leskovec, J. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584* (2017).
298. Zhang, D., Yin, J., Zhu, X. & Zhang, C. Network representation learning: A survey. *IEEE Transactions on Big Data* (2018).
299. Perozzi, B., Al-Rfou, R. & Skiena, S. *Deepwalk: Online learning of social representations* in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2014), 701–710.
300. Grover, A. & Leskovec, J. *node2vec: Scalable feature learning for networks* in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), 855–864.
301. Tang, J. et al. *Line: Large-scale information network embedding* in *Proceedings of the 24th International Conference on World Wide Web* (2015), 1067–1077.
302. Yue, X. et al. Graph embedding on biomedical networks: methods, applications and evaluations. *Bioinformatics* **36**, 1241–1251 (2020).
303. Krizhevsky, A., Sutskever, I. & Hinton, G. E. *Imagenet classification with deep convolutional neural networks* in *Advances in Neural Information Processing Systems* (2012), 1097–1105.

Bibliography

304. Zitnik, M., Agrawal, M. & Leskovec, J. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics* **34**, i457–i466 (2018).
305. Fout, A., Byrd, J., Shariat, B. & Ben-Hur, A. *Protein interface prediction using graph convolutional networks* in *Advances in Neural Information Processing Systems* (2017), 6530–6539.
306. Camacho, D. M., Collins, K. M., Powers, R. K., Costello, J. C. & Collins, J. J. Next-generation machine learning for biological networks. *Cell* **173**, 1581–1592 (2018).
307. Chakraborty, S. et al. *Interpretability of deep learning models: a survey of results in 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation* (2017), 1–6.
308. Pope, P. E., Kolouri, S., Rostami, M., Martin, C. E. & Hoffmann, H. *Explainability methods for graph convolutional neural networks* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2019), 10772–10781.
309. Zhang, S., Tong, H., Xu, J. & Maciejewski, R. Graph convolutional networks: a comprehensive review. *Computational Social Networks* **6**, 11 (2019).
310. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
311. Zou, J. et al. A primer on deep learning in genomics. *Nature Genetics* **51**, 12–18 (2019).

