Interest-based Video Summarization via Subset Selection

A dissertation submitted to

ETH ZURICH

for the degree of

Doctor of Sciences (Dr. sc. ETH Zürich)

presented by

Michael Gygli
Master Sciences, Technologies, Santé
born May 21, 1987
citizen of Eriswil, Switzerland

accepted on the recommendation of

Prof. Dr. Luc Van Gool, examiner
Prof. Dr. Kristen Grauman, co-examiner
Dr. Helmut Grabner, co-examiner

2017
To my family, for their patience and love
Abstract

In the last decade, video capture has become cheap and omnipresent. Huge quantities of videos are recorded and shared every day, creating the pressing need for better and easier ways to categorize, search, preview and edit them. This thesis proposes interest-based video summarization in order to address some of these challenges. The goal of interest-based summarization is to efficiently reduce video data to its gist, in order to make it more accessible and improve viewing experience. Towards this goal, this thesis pushes the state-of-the-art in several ways, of which we highlight three core contributions.

First, we introduce a strong, generic model for detecting interesting segments in videos. The prediction model is a Siamese Convolutional Neural Network trained using a pairwise rank loss. To train the model, we collected on a large-scale dataset of videos with annotated highlights. By exploiting the wealth of websites for GIF creation from video, we were able to create a weakly annotated dataset two orders of magnitudes larger than previous datasets of this kind.

Second, we show how structured learning for subset selection can be adapted to video summarization. Our summarization framework learns a linear combination, or mixture, of submodular objectives, thus allowing to jointly optimize objectives such as interestingness, representativeness and diversity of a summary. We further combine this model with a textual-visual embedding to create summaries that can be adapted to natural language queries.

While the submodular mixture model is able to optimize multiple objectives, it is limited by the choice of these objectives and the linearity of the model. Thus, as our third contribution, this thesis explores a novel approach for deep structured prediction. We introduce Deep Convolutional Value Networks, which are able to model higher-order interactions in structured problems while remaining tractable in inference. This model can potentially be applied to summarization,
replacing linear models based on heuristically chosen objectives with a deep, non-linear model that can be trained end-to-end.

In summary, this thesis pushes the state of the art in three important tasks of summarization: Finding interesting events, context adaptation and function optimization.
Zusammenfassung


Acknowledgements

I would like to thank all the people who accompanied me through the journey of my PhD.

First, I would like to thank my supervisor Prof. Luc Van Gool for giving me this exceptional opportunity, for his foresight and guidance. In particular I would like him to thank for giving me the space to develop and helping create many exciting opportunity along the way. I would also like to thank Prof. Kristen Grauman for agreeing to examine my thesis. I feel truly honoured.

A big thank you is due for Dr. Helmut Grabner and Dr. Hayko Riemenschneider who helped me finding my way in the Computer Vision community and patiently spent hours on me for discussions, paper writing & correcting, and sitting at BQM for brainstorming. Similarly, many colleagues from the Computer Vision Laboratory have supported me on this journey and have become friends. In particular I would like to thank the Varcity team - Santiago, Andras, Dengxin, Nikolay, Julien, Till, Kenneth - as well as Anna, Alex, Valeria, Lukas and Matthias. They, and all the other members of CVL, have made the PhD pleasure. Dr. Xavier Boix and Dr. Gemma Roig also deserve a special thank you, for supervising my Master thesis and introducing me to research and the lab.

Most importantly I would also like to thanks my friends and family who I have not seen and visited enough during this period. May you forgive me my absence. The most special thanks go to my partner Verena, for a always being there for me and patiently enduring my absorption into the more obscure aspects of my work. I am truly thankful for your love, patience and support.
## Contents

### List of Figures

### List of Tables

### 1 Introduction
1.1 Contributions of this thesis ........................................ 3
1.2 Organization of the thesis ........................................... 4

### 2 The Interestingness of Images
2.1 Chapter overview ...................................................... 7
2.2 Introduction ......................................................... 7
2.3 What causes human interest? ......................................... 10
2.4 Computational approach for interestingness prediction ...... 11
   2.4.1 Unusualness .................................................. 12
   2.4.2 Aesthetics ...................................................... 13
   2.4.3 General preferences ........................................ 14
   2.4.4 Combination .................................................. 15
2.5 Experiments .......................................................... 15
   2.5.1 Strong context: Webcam dataset ............................. 16
   2.5.2 Weak context: Scene categories dataset .................... 20
   2.5.3 Arbitrary photos: Memorability dataset ................... 21
2.6 Conclusion .................................................................. 22

### 3 Automatic Generation of Animated GIFs from Video
3.1 Chapter overview ...................................................... 25
3.2 Introduction ............................................................ 25
3.3 Related Work ........................................................... 28
3.4 Video2GIF Dataset ..................................................... 30
3.5 Method ................................................................. 33
   3.5.1 Video Processing .............................................. 34
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Chapter overview</td>
<td>85</td>
</tr>
<tr>
<td>6.2 Introduction</td>
<td>86</td>
</tr>
<tr>
<td>6.3 Background</td>
<td>88</td>
</tr>
<tr>
<td>6.4 Learning a Deep Value Network</td>
<td>90</td>
</tr>
<tr>
<td>6.4.1 Gradient based inference</td>
<td>91</td>
</tr>
<tr>
<td>6.4.2 Optimization</td>
<td>91</td>
</tr>
<tr>
<td>6.4.3 Generating training tuples</td>
<td>92</td>
</tr>
<tr>
<td>6.5 Related work</td>
<td>94</td>
</tr>
<tr>
<td>6.6 Experimental evaluation</td>
<td>95</td>
</tr>
<tr>
<td>6.6.1 Multi-label classification</td>
<td>96</td>
</tr>
<tr>
<td>6.6.2 Weizmann horses</td>
<td>97</td>
</tr>
<tr>
<td>6.6.3 Labeled Faces in the Wild</td>
<td>99</td>
</tr>
<tr>
<td>6.6.4 Ablation experiments</td>
<td>99</td>
</tr>
<tr>
<td>6.6.5 Visualizing the learned correlations</td>
<td>101</td>
</tr>
<tr>
<td>6.7 Conclusion</td>
<td>102</td>
</tr>
<tr>
<td>7 Conclusion</td>
<td>105</td>
</tr>
<tr>
<td>7.1 Contributions</td>
<td>105</td>
</tr>
<tr>
<td>7.2 Perspectives</td>
<td>106</td>
</tr>
<tr>
<td>A Submodularity Proof of Diversity Objective</td>
<td>109</td>
</tr>
<tr>
<td>B Relation of Normalized Mutual Information and Variation of Information</td>
<td>111</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Upload statistics of YouTube distribution (Data from: [140]).  
1.2 Thesis overview. We show the inputs and outputs for the methods presented. Interestingness and Video highlight detection predict the relevance of a single element, while summarization models selects sets of elements.  

2.1 Interestingness compared to aesthetics and memorability.  
2.2 What aspects relate to interestingness?  
2.3 Correlation of scene categories and interest on the dataset of [129], interestingness scores obtained as described in Sec. 2.5.2.  
2.4 An example (Sequence 1) out of the 20 webcam sequences [52] (GT: ground truth, Est: the estimated scores from our method).  
2.5 The 8 scene category dataset (GT: ground truth, Est: the estimated scores from our method).  
2.6 The memorability dataset (GT: ground truth, Est: the estimated scores from our method).  
2.7 The normalized weights for the feature combinations. The importance of unusualness features decreases, as the context becomes weak.  

3.1 Our goal is to rank video segments according to their suitability as animated GIF. We collect a large-scale dataset of animated GIFs and the corresponding video sources. This allows us to train our Robust Deep RankNet using over 500K pairs of GIF and non-GIF segment pairs, learning subtle differences between video segments using our novel adaptive rank Huber loss.
3.2 Length distribution of the input videos. .......................... 32

3.3 Most frequent video tags on the used dataset. We can observe that not all tags are equally informative. While several describe a specific visual concept (e.g., cat or wrestling) others describe abstract concepts that cannot be expected to help the task at hand. .......................... 32

3.4 Distribution over video categories. Note how the categories are highly imbalanced and often not specific. e.g., Entertainment is an extremely broad category with strong visual and semantic variation. .......................... 32

3.5 The architecture of our Robust Deep RankNet. We train the green-colored layers from scratch. Each hidden layer is followed by a ReLu non-linearity [123]. The final scoring layer is a linear function of the last hidden layer. The rank loss acts on pairs of segments and is non-zero, unless $s^+$ scores higher than $s^-$ by a margin of 1. To emphasize that the loss acts on pairs of segments, we show the two passes separately, but we use a single network. .......................... 34

3.6 Rank loss comparison. Ours Huber rank loss combines the robustness w.r.t. to small margin violations of the $l_2$ loss with the robustness to outliers of the $l_1$ loss. .......................... 37

3.7 Qualitative results. Examples of top 3 and bottom 3 predicted segments. Our approach picks up aspects that are related to GIF suitability. For example, it learns that segments with people in motion are suitable for GIFs (e.g., (a) and (c)), while low contrast segments without any (main) objects are not (e.g., (a) the 4th image). It also scores segments showing the goal area of soccer games higher than the crowd in the stadium (b). We show a failure case (d): the network scores the segments with people on the ground higher than the landing plane (4th image). .......................... 42
4.1 **Overview.** Our method consists of two parts: A supervised learning stage (training) and inference (testing). Given pairs of videos and their user created summaries as training examples, we learn a combined objective. Then, when given a new video as input, our method creates summaries that are both interesting and representative.

4.2 **Learnt weights per objective:** We can observe how the learning algorithm adapts to the specific summary length: While interestingness, *i.e.* a local prediction of importance for each segment, is the most important objective for shorter lengths, having a representative and well distributed solution becomes more important, as the summaries get longer. Weights averaged over the whole dataset.

4.3 **Egocentric dataset, Video P1:** The selected segments of the interestingness objective and our method (shown in blue). Using multiple competing objectives helps to regularize the summarization. This leads to summaries that are more representative, while still laying focus on the most most “interesting” parts (Also see Fig. 4.2). Using the interestingness objective only leads to a redundant summary, where, in this example, 4 of 16 segments depict the visit in a shoe store. Thus, it misses other substantial event from the initial video.

4.4 **Egocentric dataset, Video P1:** Textual representation of a summary created with our method. Our method selects expressive segments from each of the main events and the travel between them. It tells the same story as the summary created by a human annotator. Please also be aware of the fact that the reference summaries are not extractive, *i.e.* they can contain formulations and sentences that are not in the annotation and can thus never be selected. One consequence of this is more repetitive sentences in the automatic summary. We give a keyframe visualization for the same video in Fig. 4.3.
4.5 **Egocentric dataset, Video P2:** Quantitative comparison of uniform sampling, [92] and our method. [92] is prone to include duplicates and misses the visit of the frozen yogurt shop. While uniform sampling includes this, it is hardly visible for the selected segments/frames: Uniform sampling more often selects frames that are non-informative. Instead, our method is diverse and selects segments that help understand the story.  

5.1 Our query-adaptive video summarization model picks frames that are relevant to the query while also giving a sense of entire video. We want to summarize a video of an iron-man competition, in which participants swim, bike and run. Query-adapted summaries are representative by showing all three sports, while placing more focus on the frames matching the query.  

5.2 A visualization of the latent semantic embedding space. Semantically similar concepts are projected to similar locations in the vector space.  

5.3 Qualitative Results of top ranked keyframes on RAD. a) Liu et al. b) Video2GIF c) Our new model (from left). Video titles are shown on the left. Ground truth relevance labels are shown in Blue. \(P^=\)Positive, \(N^=\)Negative.  

5.4 Each row represents a video summary created by Video2GIF, Relevance objective and Our model (rows: 1, 6 and 11 in Tab. 5.4). Green number on images depicts frame number. We plot the distribution of relevance scores and cluster annotations over the video and marked the selected frames for the methods.  

6.1 Segmentation results of DVN on Weizmann horses test samples. Our gradient based inference method iteratively refines segmentation masks to maximize the predicted scores of a deep value network. Starting from a black mask at step 0, the predictions converge within 30 steps yielding the output segmentation.
6.2 A deep value network with a feed-forward convolutional architecture, used for segmentation. The network takes an image and a segmentation mask as input and predicts a scalar evaluating the compatibility between the input pairs. .................................................. 97

6.3 Visualization of the learned shape priors (Weizmann horses). From left to right (a) The mean mask of the training set (b) mask generated when providing the mean image of the training set (c, d) Samples generated by our model by adding Gaussian random noise to the mean image with $\sigma = 10$. .......................................................... 100

6.4 Qualitative results on the Weizmann $32 \times 32$ dataset. In comparison to previous works, DVN is able to learn a strong shape prior and thus correctly detect the horse shapes including legs. Previous methods are often misled by other objects or low contrast, thus generating inferior masks. ................................................. 103

6.5 Qualitative results on the Labeled Faces in the Wild (LFW) 3-class segmentation. The last two rows show failure cases, where our model does not detect some of hair and moustache correctly. ........................................... 104
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The interestingness cues and their performance on the 3 datasets. We highlight best combination and best and second best single feature.</td>
<td>23</td>
</tr>
<tr>
<td>3.1</td>
<td>Statistics on the Video2GIF dataset. We show numbers for the complete dataset and for the one after discarding too short or too long videos (see text). We also show the number of videos that come with the Creative Commons license (CC-BY).</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>Experimental results. A lower nMSD and higher mAP represent better performance.</td>
<td>41</td>
</tr>
<tr>
<td>3.3</td>
<td>Cross-dataset results (mAP). We train on our dataset and test on the video highlight dataset of [157]. Our method outperforms rankSVM and [187], which learns an unsupervised model for each domain. Sun et al. [157] performs best, but it is directly trained on their dataset and learns multiple models, one per category. Instead, we learn a single global model for GIF suitability.</td>
<td>43</td>
</tr>
<tr>
<td>4.1</td>
<td>Taxonomy of the most recent and relevant video summarization methods. We differentiate in terms of objectives they optimize and and how they combine multiple objectives. Many methods score segment locally. Others combine multiple objectives, but do so based on a hand-defined sequential optimization. In opposition, we learn the importance of each objective from data and optimize them jointly.</td>
<td>49</td>
</tr>
</tbody>
</table>
4.2 **Egocentric dataset.** Performance of the individual objectives and previous methods vs. our approach for summaries of 2 minute length. .................................................. 56

4.3 **Egocentric dataset.** We use the same summary length as Lee et al. [92], in order allow a comparison (≈ 1 minute and 20 seconds) .................................................. 58

4.4 **User video dataset.** Performance of the individual objectives and previous methods vs. our approach. ........... 61

5.1 Comparison of model configurations trained on a subset of the Clickture dataset or our Video Thumbnail dataset (RAD). As can be seen, having a better text representation and training objective substantially improves performance compared to the model proposed by Liu et al. [107] 78

5.2 Performance of our relevance models on the RAD dataset in comparison with previous methods. ........... 80

5.3 Thumbnail selection performance on the MSR Evaluation dataset. Note that [107] uses queries for their method which are not publicly available. Thus the numbers are not directly comparable. ........... 82

5.4 Performance of summarization methods on the RAD dataset. Repr means Representativeness. ✓ and – depict whether an objective was used or not. MMR and ours learn their corresponding weights. Percentage in parentheses the normalized learnt weights. Upper bound refers to the best possible performance, obtained using the ground truth annotations of RAD. ........... 84

6.1 Tag prediction from text data. $F_1$ performance of Deep Value Networks compared to the state-of-the-art on multilabel classification. All results except ours are taken from [7, 104]. .................................................. 96

6.2 Test IOU on Weizmann-32×32 dataset. Our method outperforms all previous methods, despite using a much lower input resolution than [188] and [143]. ........... 98

6.3 Superpixel accuracy (SP Acc.) on Labeled Faces in the Wild test set. .................................................. 100
6.4 Test performance of different configurations on the Weizmann 32x32 dataset. 101
We live in a time in which abundant amounts of data are collected. This data comes in many forms and is collected for a variety of reasons. Data-hungry companies, for example, encourage users to create profiles, in order to understand their needs. This allows them to provide personalized advertisement or better recommendations. But data is also collected by users themselves and shared on social networks and photo and video sharing websites such as Facebook, Instagram, YouTube and Flickr. In particular for images and videos, we have seen a strong increase in the amount of data being collected and shared. To date, 80 million photos are uploaded to Instagram each day [138]. Similarly, the amount of videos on YouTube is growing at an increasing pace (cf. Fig. 1.1), with 400 hours uploaded every minute as of November 2015. These massive amounts of visual content have created the pressing need for ways to easily categorize, search and preview them. This, in turn, has spurred research in areas such automatic tagging, image description and action recognition. The goal of these approaches is to automatically augment videos and images with textual data so that they can be categorized or searched via text queries. An alternative way to reduce the amount of visual data is through automatic filtering based on criteria such as aesthetics [30, 33] or interestingness [52, 53]. In particular, many approaches have focused on making videos more accessible by automatically editing them and by generating overview summaries [3, 55, 58, 157]. Such summaries help users to quickly assess the relevance and interestingness of videos and improve the viewing experience [167]. This thesis also focuses on summarization and makes several contributions in that area.

Editing and summarizing videos is challenging even for experienced human editors. To automate this process, a system needs to obtain an understanding
of high-level semantics in the video, to be able to assess what content is interesting or relevant. It needs to recognize what objects are present, where a scene is taking place and what kind of interactions occur. To give an example: Unusual or funny things are often considered interesting [57], something that is notoriously hard to predict [20]. To complicate things further, what is considered interesting or relevant is, to some extent, subjective and context-dependent. Thus, an ideal method generates summaries that can be adapted to specific users and contexts.

While the problem is challenging, not all hope is lost. Various works have shown that there exists considerable agreement in what is considered interesting [53, 54, 152, 155]. This makes it possible to learn a generic interestingness or importance model, as has been done in various works, e.g., [92], including several approaches described in this thesis. Furthermore, videos often come with additional meta-data such as tags or the title. The title often provides a strong indication on what is the central and most important topic of a video. This has been exploited, for example, in [78, 106, 155]. When summaries are shown in a search scenario, another option is to rely on the query to predict relevance of frames or segments [107]. We describe a method relying on these ideas in Chapter 5, where we present a method that produces summaries adapted to a query or the video title.

Apart from showing relevant content, a great summary requires additional properties. Certainly, it should not omit important content of the initial video, i.e. it has to be representative of the video and show the full diversity of its content [55, 167]. Other desired properties might be temporal uniformity (cf. Sec-

Figure 1.1: Upload statistics of YouTube distribution (Data from: [140]).
1.1 Contributions of this thesis

Truong and Venkatesh [167] have identified three broad perspectives on video summarization: Coverage, Interesting events, and query context/personalization. This thesis makes contributions to all of these. In particular it pushes the state of the art for finding interesting events and personalization through text queries.
Additionally, we make contributions in structured prediction, useful for optimizing summaries. Specifically, this thesis contributes:

- An empirical analysis of visual interest and its relation to objects and scene content as well as affective attributes (Section 2.3).
- A new large-scale dataset for video highlight detection based on webly annotated data (Section 3.4). This dataset is the largest of its kind and one to two orders of magnitude larger than previously published datasets.
- A novel state-of-the-art model for predicting GIF suitability of video segments by using a Siamese neural network model (Section 3.5.2). Previous works was limited to simple, linear models by the size of available datasets (see Section 3.6.2). This model is also the first publicly available highlight detection model.
- The adaption of the submodular mixtures model of [102] to video summarization. This allows to jointly optimize for different objectives, as opposed to a traditional, sequential optimization (Chapter 4).
- A video summarization model which allows to adapt summaries to natural language queries (Chapter 5). This model advances the state of the art, which was previously limited to handling small vocabularies such as one or several single-word categories [145, 157].
- A novel approach for end-to-end structured learning. We introduce Deep Value Networks, which allow to model complex higher-order interactions in structured prediction problems while remaining tractable in inference (Chapter 6).

1.2 Organization of the thesis

This thesis is organized into Chapters that each correspond to a research paper on a specific topic. Each chapter starts with a brief overview of the contents and puts the work into context with the others. We provide a schematic categorization of the chapters that focuses on the analysis of visual data in Figure 1.2. The chapters are organized as follows:

1 Provided on https://github.com/gyglim/video2gif_dataset
2 Provided on https://github.com/gyglim/video2gif_code
3 Provided on https://github.com/gyglim/gm_submodular
Chapter 2 provides an empirical analysis on what images are considered *interesting*, based on their objects, scenes, aesthetic properties, etc. It further proposes a model to estimate the interestingness of an image based on low-level content features. This Chapter is based on [53], which has been published at ICCV 2013.

Chapter 3 introduces a new model for predicting *video highlights*. Specifically, it presents a model that estimates the GIF suitability of video segments, using a Siamese Convolutional Neural Network (CNN). To train the model, we collected a new large-scale dataset with webly annotated data (Section 3.4). This Chapter is based on [58], which has been published at CVPR 2016.

Chapter 4 proposes a general and versatile method for creating summaries that comply with multiple objectives. As discussed, summaries should, among others, be relevant and representative for the input video. We formulate the problem as a structured prediction problem and learn a linear combination of multiple submodular set functions using the method of [102]. With this formulation, the proposed model can create summaries that satisfy all of the desired objectives of a good summary. This Chapter is based on [55], which has been published at CVPR 2015.

Chapter 5 introduces a model for *query-adaptive video summarization*, where the summary is adapted to a text query. The method learns a textual-visual embedding space, where queries and frames can be compared using similarity measures such as the cosine similarity to assess relevance. This relevance score is incorporated into a submodular mixtures framework to create relevant, interesting and representative summaries. This Chapter is based on [175], which has been published at ACM Multimedia 2017.

Chapter 6 investigates a novel approach for structured prediction. Our method learns a novel Deep Value Network that quantifies the quality of different solution hypotheses for a given input. Once the value net is trained, at inference, a solution is obtained by iteratively updating an initial solution in order to maximize the predicted quality of the output. This Chapter is based on [56], which has been published at ICML 2017.
Chapter 7 concludes this thesis. It further gives a brief review of the biggest remaining challenges in the area and how they can potentially be tackled.
2 The Interestingness of Images

2.1 Chapter overview

In this chapter we investigate human interest in photos. Based on our own and others’ psychological experiments, we identify various cues for “interestingness”, namely aesthetics, unusualness and general preferences. For the ranking of retrieved images, interestingness is more appropriate than cues proposed earlier. Interestingness is, for example, correlated with what people believe they will remember. This is opposed to actual memorability, which is uncorrelated to both of them. We introduce a set of features computationally capturing the three main aspects of visual interestingness that we propose and build an predictor from them. Its performance is shown on three datasets with varying context, reflecting diverse levels of prior knowledge of the viewers.

2.2 Introduction

With content-based image retrieval on the rise, there is a parallel increase in the study of cues that could help in ranking the retrieved images. These include image quality [76], memorability [67] and aesthetics [30, 33]. Yet, a measure that would seem more relevant to automatically quantify is how interesting people find an image and this “interestingness” has hardly been studied so far. Apart from retrieval, other applications like video summarization or automated camera hand-over are also bound to benefit.

The most related work might be that by Dhar et al. [33], who used their high-level aesthetics features to train a classifier on Flickr’s interestingness. It can
predict the interestingness of these images well, but it is questionable that these results can be generalized to other image datasets. Flickr’s interestingness [17] is based on social behavior, i.e., according to the uploader’s reputation and a non-disclosed ratio between views, favorites and comments on the images. This measure has not been shown to relate to what people find interesting in images. For example, images where interest is caused through negative emotions (disgust, disturbance, threat, etc.) tend to get low Flickr interestingness. Users will hardly select such images as a favorite.

In our own series of psychological experiments we analyze “interestingness” and how it relates to measures such as aesthetics and memorability (Fig. 2.1). There exists indeed a strong correlation between aesthetics and interestingness (Fig. 2.1, top row). However, what is interesting does not necessarily need to be aesthetically pleasing, e.g., the image of skulls is interesting, even though it is not aesthetic. While one would also expect a high correlation of memorability and interestingness, our experiments indicate the contrary (Fig. 2.1, bottom row). More details are to follow.

In this chapter we (i) investigate what arouses human interest (Sec. 2.3) and show that it is fundamentally different from other properties such as memorability; (ii) propose a set of features able to computationally capture the most im-
2.2. INTRODUCTION

(a) Interestingness correlated with an extensive set of image attributes, based on the data of [66]. We compare the attributes to our interestingness score, collected as described in Sec. 2.5.3.

(b) Correlations of noteworthy attributes from above and interestingness.

Figure 2.2: What aspects relate to interestingness?
2.3 What causes human interest?

In his seminal work Berlyne [11] introduced four variables affecting interest: novelty, uncertainty, conflict and complexity. He showed that new, complex and unexpected events are a strong trigger of interest. Recent psychological research extends Berlyne’s theory, e.g., Silvia [147] who analyzes the effects of complexity and understandability on interest. The more computational approach in [144] concurs with these ideas. Biederman and Vessel [12] explain interest with perceptual pleasure, resulting from comprehensible information and newly activated synapses. They furthermore found that natural scenes with wide landscapes are preferred over man-made scenes. Other cognitive work by Chen et al. [22] identifies novelty, challenge, instant enjoyment, and demand for attention as sources of interestingness. While Smith and Ellsworth [150] found that high pleasantness is a major aspect of interestingness, recent studies [171] indicate otherwise for images with polygons and paintings.

Given the lack of clear-cut and quantifiable psychological findings, we investigate the correlation of interestingness with an extensive list of image attributes, including emotional, aesthetic and content related aspects. We use the dataset of Isola et al. [67], extended in [66] to explore memorability. In Fig. 2.2a we relate the provided image attributes to the interestingness ground truth we collected (cf. Sec. 2.5.3). This figure shows the Spearman rank correlation of all attributes and highlights several with high correlations (either positive or negative). Fig. 2.2b shows the correlations of four example attributes in more detail. In keeping with the work in psychology we find three main groups with high influence: novelty/unusualness (attributes: unusual, is strange, mysterious), aesthetics (attributes: is aesthetic, pleasant, expert photography) and general preferences for certain scene types (attributes: outdoor-natural vs. indoor and enclosed spaces).

As the predictability of related concepts (e.g., aesthetics) has been approached successfully in the past, there is good hope that we can computationally predict interestingness, based on the above cues. This assumption is supported by our experiments. When comparing the data of [66] with our own we find that people agree, to a large extent, on which images are interesting, despite personal preferences (cf. Sec. 2.5.3). This observation of a high inter-individual agreement for real-world images was also shown by [177].

The cues that we implemented were selected on the basis of their experimentally verified correlation with interestingness. That is important, as intuition
can often be misleading. For instance, Isola et al. [66] have shown that human prediction of what is memorable (i.e. assumed memorability) is negatively correlated with actual memorability. Interestingness, on the other hand, has its highest correlation with this assumed memorability. What a human observer finds interesting is what he wants to remember and believes he will. Unfortunately the latter is often not the case.

Additionally, we investigated the preference for certain scene types (Fig. 2.3) and found, in agreement with [12], that people prefer natural outdoor scenes rather than man-made scenes. While interestingness is higher for images containing sky, actual memorability decreases if sky is present. Indeed, when comparing actual memorability and interestingness, we find them to be negatively correlated\(^1\). Nevertheless, we believe that also for applications like selecting images for advertisements (mentioned by Isola et al. [66]), it makes more sense to select an interesting image, than a memorable but dull one. In the end, the goal is not to have people remember the image, but the message combined with a positive connotation.

2.4 Computational approach for interestingness prediction

In this section we propose features that computationally capture the aspects/cues of interestingness which we found most important (cf. Fig. 2.2) and are imple-
mentable: unusualness, aesthetics and general preferences. Then, we use these to predict the interestingness of images. Thereby we build upon the set of features we used in our previous work for image sequences [52] and extend it with additional features suitable for single images.

Formally, given an image $I$ we are looking for an interestingness score $s$. Our pipeline to achieve this task consists of two stages: (i) exploring various features to capture each of the above cues for interestingness and (ii) combining these individual features.

2.4.1 Unusualness

As said, unusualness/novelty is an important cue for interestingness. Abnormality detection goes in that direction, but most contributions consider constrained surveillance settings with fixed cameras. Instead, we want to capture unusualness in single images from arbitrary scenes. We propose two different methods, one relying on global image descriptors and one on image parts.

Global outliers. We detect global outliers in the dataset by applying the Local Outlier Factor (LOF) algorithm [16] to global image descriptors. LOF gives a measure to what degree a data point (an image, in our case) is outlying, taking into account its $k$ nearest neighbors. It is called local, as the outlier factor is calculated with respect to the density of its closest cluster. In all our experiments we use a 10-distance neighborhood and as features (i) the raw RGB pixel values $s_{\text{pixel}}^{\text{unusual}}$, (ii) GIST [129] $s_{\text{gist}}^{\text{unusual}}$ and (iii) Spatial Pyramids on SIFT histograms [89] $s_{\text{pyr}}^{\text{unusual}}$. We use these features, as they have been found effective for scene categorization [89, 129] and image classification [89].

A similar idea was used by Datta and Wand [30], where they propose a Familiarity feature. This measure is computed as the average distance of a test image to the $k$ closest training images (based on local features). The higher this distance the less familiar (more unusual) an image. Interestingly they found this feature to play a crucial role in the classification of image aesthetics (cf. Fig. 5.1, correlation of interestingness and aesthetics $\rho = 0.59$).

Composition of parts. We also propose an approach that operates on image parts. It is inspired by the work of Boiman and Irani for the detection irregularities in images and videos [13]. Ensembles of patches from the tested image are matched against patches from the database, which defines what is “regular”. If
there is a good match, it is considered regular, otherwise as irregular. Instead of using square patches, overlapping between foreground and background, we over-segment the image using superpixels (SLIC [1]). This allows for a delineation of the image parts. We model the image as a graph with superpixels as nodes. The graph’s energy determines how unusual the configuration of patches is:

$$E(L) = \sum_{i \in S} D_i(l_i) + \lambda \sum_{\{i,j\} \in N} V(l_i, l_j)$$  \hspace{1cm} (2.1)$$

where $S$ is the set of superpixels and $N$ the set of superpixel neighbors. $D_i(l_i)$ denotes the unary cost of assigning label $l$ to the superpixel $i$. The label space $L$ is the set of images in the database (i.e. $|L|$ is equal to the number of database images). The unary cost $D_i(l_i)$ is the Euclidean distance in the descriptor space of a superpixel $i$ to the nearest-neighboring superpixel in the database with label $l$. The set of descriptors is that of [164], which includes features such as SIFT, Texton and Color histograms as well as location information. The binary terms $V(l_i, l_j)$ denote the cost of two neighboring nodes taking labels $l_i$ and $l_j$, respectively. They encourage label smoothness, as $V(l_i, l_j) = 0$ if $l_i = l_j$ and 1 otherwise, i.e. a simple Potts penalty. Empirically we found the weighting parameter $\lambda$ to be robust and we keep it fixed at 0.02.

To find the labeling that minimizes the cost, we apply MAP inference based on a standard GraphCut algorithm [15]. With $L$ being that optimal labeling, the unusualness by composition is defined as $s_{\text{compose}}^{\text{unusual}} := E(L)/|S|$, i.e. the energy in the Markov Random Field normalized by the number of superpixels.

Intuitively this feature encodes how well an image can be composed with parts of images in the database, while encouraging the composition of connected regions from only one image.

### 2.4.2 Aesthetics

To capture the aesthetics of an image, we propose several features that are rather simple in comparison to other, more extensive works in the area. For example [33] uses content preferences, such as the presence of people and animals or the preference for certain scene types to classify aesthetically pleasing images. We capture such general preferences with global scene descriptors in Sec. 2.4.3. For predicting aesthetics, we focus on capturing visually pleasing images, without semantic interpretation.
Colorfulness. We measure colorfulness as proposed by Datta and Wang [30], i.e. as the Earth Mover distance (in the LUV color space) of the color histogram of an image \( H_I \) to a uniform color histogram \( H_{uni} \). A uniform color histogram is the most colorful possible, thus the smaller the distance the more colorful the image, \( s_{colorful}^{aesth} := -\text{EMD}(H_I, H_{uni}) \).

Arousal. Machadjik and Hanbury [111] extracted emotion scores from raw pixels. Their features are based on the empirical findings of [172], which characterized emotions that are caused by color using the space of arousal, pleasure and dominance. As interest is correlated with arousal (cf. Fig. 2.2a), we use an arousal score as in [111]. It is calculated as the average over all pixels \( p \) of the image as \( s_{arousal}^{aesth} := \sum_p -0.31 \text{brightness}(p) + 0.60 \text{saturation}(p) \).

Complexity. To capture the complexity of an image, we compare its size after JPEG compression against its uncompressed size, specifically we compute \( s_{complex}^{aesth} := \frac{\text{bytes(compress}(I))}{\text{bytes}(I)} \). We use JPEG as it is a lossy compression, which compresses an image according to the human visual system [179]. If the compression rate is high \( s_{complex}^{aesth} \) is low, as there is little visually important information in the image.

Contrast. We use the same contrast quality measure as [76], i.e. we calculate the minimal range of the 98% mass of a gray-scale histogram to obtain \( s_{contrast}^{aesth} \).

Edge distribution. Following [76] we calculate the image bounding box that contains 98% of the edges in each dimension (i.e. along \( x \) and \( y \)). Smaller bounding boxes typically correspond to less clutter and a more uniform background. We use \( s_{edges}^{aesth} := 1 - w_x w_y \), with \( w_x \) and \( w_y \) being the box’s normalized width and height;

2.4.3 General preferences

Following the observation that certain scene types tend to be more interesting than others (cf. Fig. 2.3 and [12]), we propose to learn such features from global image descriptors. We train a Support Vector Regressor (\( \nu \)-SVR [21]) on raw RGB-pixels \( s_{pixel}^{pref} \), GIST [129] \( s_{gist}^{pref} \), spatial pyramids of SIFT histograms [89] \( s_{pyr}^{pref} \), and color histograms \( s_{hist}^{pref} \). Spatial pyramids and GIST are known to capture scene categories well. RBF kernels served our purposes in all cases. We used \( \nu = 0.5 \) and optimized the parameters \( \gamma \) and \( C \) through
grid search on the validation set. We tested \( C = \{2^{-5}, 2^{-3}, \ldots, 2^7, 2^9\} \) and \( \gamma = \{2^{-15}, 2^{-13}, \ldots, 2^1, 2^3, 2^5\} \).

### 2.4.4 Combination

The scores obtained from the respective features are first normalized with respect to their mean and variance. Second, they are mapped into the interval \([0, 1]\) using a sigmoid function \( \bar{s} = \frac{\exp(\mu s)}{1+\exp(\mu s)} \) where the parameter \( \mu \) is estimated using least-square minimization on the validation set. To combine the individual features, we perform greedy forward feature selection. Starting from the best single feature, we select additional features until the combination does not improve further, as a quality measure using Spearman’s \( \rho \). As a model we use a simple linear combination \( \bar{s}_{\text{comb}} = \mathbf{w}^T \bar{s}_{\text{sel}} \), where \( \bar{s}_{\text{sel}} \) is the vector of selected features. The weights are trained using least-squares.

As we use a linear model that assumes uncorrelated features, we also applied whitening to decorrelate the features before training the model. We define \( \bar{s}_{\text{decorr}} = \Sigma^{-1/2} \bar{s} \), where \( \Sigma \) is calculated on the training set. This whitening step leads to only a marginal improvement, suggesting that the features are indeed complementary (cf. Tab. 2.1).

### 2.5 Experiments

In this section we discuss the performance of the different interestingness features. As we will see, the strength of the contextual cues that are relevant in the tested setting determines – in part – which types of features are most effective in capturing interestingness. First, we specify the selection of parameters and the evaluation criteria. Then, we run through the results for three datasets.

**Parameters.** For the features based on raw pixels \( s_{\text{pixel}}^{\text{unusual}} \) and \( s_{\text{pixel}}^{\text{pref}} \) we used downscaled images of \( 32 \times 32 \) pixels, which is the size we found to work best. This agrees with [165], where it was shown sufficient to capture scene types and important objects. For each dataset we use a training/validation/test split. The training set serves as a database for all the outlier methods, i.e. their response is high if a test image is not somehow similar to the training data. As for the general preference features, we trained the \( \nu \)-SVR on the training set and optimized the hyperparameters using grid search on the validation set.
The estimation of $\mu$ for the sigmoid function of each feature and the feature selection and estimation of the weight vector $w$ for the combinations are also performed on the validation set. Both, the test and validation set consist of 240 randomly selected images (unless specified otherwise).

**Evaluation.** In order to evaluate feature performance quantitatively, we use multiple measures. These include standard measures such as Recall-Precision (RP), Average Precision (AP) and Spearman’s rank correlation $\rho$. For the RP evaluation we use images with significant agreement between individuals. Images with a ground truth score $s^* > 0.75$ are taken as positive and $s^* < 0.5$ as negative examples. Images with in-between scores are excluded in the computation of RP, as there is no clear agreement between individuals.

In addition, we use the $\text{Top}_N$ score, which quantifies how well the computer-ranked top $N$ images agree with the human ranking. Suppose that $s^*_i$ is the human interestingness score of image $I_i$, then $\text{Top}_N := \frac{\sum_{i \in P_N} s^*_i}{\sum_{i \in S_N} s^*_i}$, where $P_N$ is the set of $N$ images ranked highest by a method, and $S_N$ the set of $N$ images ranked highest according to human consensus. As can be easily seen, $\text{Top}_N \in [0, 1]$, where a higher value corresponds to a better performance of the algorithm.

We use the following datasets: Firstly, a set of webcam sequences [52]. Since the presented webcam images are sequentially evolving, there is a strong context in which a viewer rates interestingness. Secondly, we use the 8 scene category dataset [129], which provides some weaker semantic context. Last, we use the memorability dataset [67], which contains arbitrary photographs and offers practically no context. The overview of the results is shown in Tab. 2.1.

### 2.5.1 Strong context: Webcam dataset

This dataset consists of 20 different webcam streams, with 159 images each. It is annotated with interestingness ground truth, acquired in a psychological study [52]. Thereby the subjects were shown the webcam images in sequence, and instructed to press a button if they considered something interesting. No further instruction was given so that the participants were free to judge what they considered as interesting [52]. From that data, the interestingness score of an image is calculated as the fraction of people who considered it interesting.
2.5. Experiments

(a) Human labeling. **Top:** most interesting
**Bottom:** least interesting.

(b) Interestingness vs. predicted score for Sequence 1.

(c) Predicted interestingness.
**Top:** most interesting
**Bottom:** least interesting.

(d) Recall-Precision curve. We show the combination and the five highest weighted cues.

Figure 2.4: An example (Sequence 1) out of the 20 webcam sequences [52] (GT: ground truth, Est: the estimated scores from our method).

There are only a few interesting events in these streams (mean interestingness score of 0.15). Interestingness is highly subjective and there are individuals who did not consider any image interesting in some sequences. An example of these sequences is shown in Fig. 2.4a.

As a consequence of the low mean interestingness, we use different thresholds for RP calculation: $s^* > 0.5$ as positive and $s^* < 0.25$ as negative samples, which results in about five interesting images per sequence, on average. As we are interested in determining the frames with high interest, (cf. Fig. 2.4b), $Top_5$ scores provide a good characterization. We tested each sequence separately and split the remaining sequences randomly into training and validation sets (80% for training / 20% for validation) to train the SVRs and the combination of the features.
2. The Interestingness of Images

(a) Human labeling. **Top:** most **Bottom:** least interesting.

(b) Interestingness vs. predicted score.

(c) Predicted interestingness. **Top:** most interesting **Bottom:** least interesting.

(d) Recall-Precision curve. We show the combination along with the five highest weighted cues.

Figure 2.5: The 8 scene category dataset (GT: ground truth, Est: the estimated scores from our method).

Following the setup of the experiment, we sequentially added the images to the database, as they were tested. *i.e.* the unusualness scores are computed with respect to the previous frames only (while [52] uses the whole sequence).

**Results.** The mean performance over all 20 sequences is shown in Tab. 2.1. Results for a sample sequence are shown in Fig. 2.4. Fig. 2.4a shows frames of the sample sequence, while Fig. 2.4c shows the top and bottom predictions of our algorithm. Fig. 2.4b shows the correlation of predicted interestingness and ground truth score and Fig. 2.4d plots the Recall-Precision curve for the combination of features along with the five single features having the highest weights.
2.5. Experiments

(a) Human labeling. **Top:** most **Bottom:** least interesting.

(b) Interestingness vs. predicted score.

(c) Predicted interestingness. **Top:** most interesting **Bottom:** least interesting.

(d) Recall-Precision curve. We show the combination along with the five highest weighted cues.

Figure 2.6: The memorability dataset (**GT:** ground truth, **Est:** the estimated scores from our method).

Outlier methods perform best in this setting. Yet, not everything predicted as unusual is rated as interesting by humans, *e.g.*, for image 1, Fig. 2.4c, the method overestimates interestingness, because of cloud formations. This is not unusual at the semantic level and therefore not considered interesting by humans. Other typical failure cases include camera shifts (global outlier methods) and direct sunlight or shades. Aesthetics and general preference features show a lower performance. When comparing median scores of our approach to [52] we achieve comparable performance (**AP:** 0.39 (ours) vs. 0.36 [52]; **Top3:** 0.66 (ours) vs. 0.72). The better performance of our method in terms of **AP** is due to better visual features. [52], on the other hand, has a better **Top3** score due to its temporal regularization. Without regularization, [52] has a **Top3** of 0.59.
2.5.2 Weak context: Scene categories dataset

The 8 scene categories dataset of Oliva and Torralba [129] consists of 2’688 images with a fixed size of 256 × 256 pixels. The images are annotated with their scene categories, which allows us to investigate the correlation between scene types and interestingness. The images are typical scenes from one of the 8 categories (coast, mountain, forest, open country, street, inside city, tall buildings and highways). Examples are shown in Fig. 2.5a.

We extended this dataset with an interestingness score by setting up a simple binary task on Amazon Mechanical Turk. A worker was presented with a pair of randomly selected images from different scene types and asked to choose the one he/she considered more interesting. 20 pairs were grouped into one task. The interestingness score of an image was calculated as the fraction of selections over views. We used this approximation, as a full pairwise comparison \(O(n^2)\) is infeasible for a dataset of this size. Every image was viewed by 11.9 workers on average (min. 10), which is equal to the number of views of [66]. To ensure high quality results, only workers with the “Masters” level were allowed for the task.

Results. Fig. 2.5 and Tab. 2.1 show the results of our features on this dataset. The scene categories provide a weak context, given by the prior on the scene type, which allows to capture novelty/unusualness, as outliers to what are typical images of a certain scene category. Nonetheless, all outlier methods perform less well on this dataset. Novelty is not only harder to capture in this setting, it is also less clearly defined, than in the case of webcams. The algorithm can only capture unusualness with respect to the training images (the prior knowledge of our algorithms), not the observer’s prior experience. Furthermore this dataset contains very few unusual images. Therefore a viewer mainly rates the images in this dataset according to aesthetics and general preferences, which transpires from the performance of the individual features.

General preference features yield the highest performance, as they are able to capture scene type and illumination effects \(G_{hist}^{pref}\), such as the color of a sunset. The features learn the preference for certain scene types (cf. Sec. 2.3, Fig. 2.3) and the aversion for road scenes.
2.5.3 Arbitrary photos: Memorability dataset

The memorability dataset consists of 2’222 images with a fixed size of $256 \times 256$ pixels. It was introduced in [67] by Isola et al. and further extended in [66] to investigate the memorability of images (see examples in Fig. 2.6).

The annotation of Isola et al. [66] also includes an attribute on interest in their study (attribute “is_interesting”). In their experimental setting, they asked a user to classify an image as interesting/non-interesting. In contrast, we conducted the same study as in Sec. 2.5.2 on this dataset: We performed a binary experiment, where a user had to select the more interesting image from a pair. The availability of these two experiments allows us to analyze and compare them. Despite the different experimental setting, the scores obtained show a strong correlation ($\rho = 0.63$), suggesting that images hold an intrinsic interestingness.

**Results.** Fig. 2.6 and Tab. 2.1 show the results of our features on this dataset. The trained regressor for general preferences performs best. Not surprisingly, unusualness features perform badly. Based on the psychological findings (cf. Sec. 2.3) unusualness/novelty probably remains equally important here. Unfortunately, we are not able to capture it for two reasons: (i) What is unusual or novel, in this unconstrained setting, depends on the prior knowledge of the observers, which is unknown to the algorithm. (ii) Semantics are crucial in the appraisal of what is unusual in this dataset. Take, for example, image 3 in the top row of Fig. 2.6a. This image is interesting, as it shows the end of a glacier. To predict the interestingness of such an image correctly, we need to understand such semantics.
2.6 Conclusion

Interestingness is an important image property. It is clearly subjective and depends, to a certain degree, on personal preferences and prior knowledge. Nonetheless there exists a substantial agreement about it among observers. This also allowed us to capture it computationally. We proposed a set of features able to capture interestingness in varying contexts. With strong context, such as for static webcams, unusualness is the most important cue for interestingness. In single, context-free images, general preferences for certain scene types are more important. Fig. 2.7 illustrates the importance of the different interestingness cues as context gets weaker. Unusualness, while remaining important, becomes more difficult to capture with weak contexts. To overcome the current limitations of interestingness prediction, one would need: (i) an extensive knowledge of what is known to most people, (ii) algorithms able to capture unusualness at the semantic level and (iii) knowledge about personal preferences of the observer.

Apart from these more fundamental improvements, better visual features can also help. While we used simple visual features for this work, Convolutional Neural Networks have shown strong performance on image recognition tasks, since then. [36, 146] have shown that the learnt representations of these networks generalize to a wide variety of tasks. In our recent follow up work [57], we relied to these representations and showed that interestingness can be predicted more accurately as well. Using deep features increased the performance on the memorability dataset [67] substantially, where the rank correlation improved from $\rho = 0.6$ to $\rho = 0.7$. 
<table>
<thead>
<tr>
<th>Context</th>
<th>Cue</th>
<th>Feature</th>
<th>$\rho$</th>
<th>AP</th>
<th>$Top_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Strong</strong></td>
<td>Unusual</td>
<td>compose</td>
<td>0.29</td>
<td>0.35</td>
<td>0.51</td>
</tr>
<tr>
<td>Webcams [52]</td>
<td></td>
<td>pixel</td>
<td>0.23</td>
<td>0.22</td>
<td>0.53</td>
</tr>
<tr>
<td>Static camera:</td>
<td></td>
<td>pyr</td>
<td>0.01</td>
<td>0.10</td>
<td>0.31</td>
</tr>
<tr>
<td>20 different</td>
<td></td>
<td>gist</td>
<td>0.03</td>
<td>0.12</td>
<td>0.28</td>
</tr>
<tr>
<td>outdoor sequences</td>
<td>Aesthetic</td>
<td>arousal</td>
<td>0.13</td>
<td>0.24</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td></td>
<td>complex</td>
<td>0.09</td>
<td>0.26</td>
<td>0.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>colorful</td>
<td>-0.06</td>
<td>0.06</td>
<td>0.26</td>
</tr>
<tr>
<td></td>
<td></td>
<td>edges</td>
<td>-0.04</td>
<td>0.07</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contrast</td>
<td>0.10</td>
<td>0.15</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>Pref.</td>
<td>pixel</td>
<td>0.04</td>
<td>0.11</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pyr</td>
<td>0.05</td>
<td>0.10</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gist</td>
<td>0.16</td>
<td>0.18</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td></td>
<td>colorhist</td>
<td>0.05</td>
<td>0.12</td>
<td>0.36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>combined</td>
<td>0.32</td>
<td>0.39</td>
<td>0.57</td>
</tr>
<tr>
<td></td>
<td></td>
<td>comb. decorr.</td>
<td>0.31</td>
<td>0.42</td>
<td>0.61</td>
</tr>
<tr>
<td><strong>Weak</strong></td>
<td>Unusual</td>
<td>compose</td>
<td>0.18</td>
<td>0.28</td>
<td>0.38</td>
</tr>
<tr>
<td>Scene categories [129]</td>
<td></td>
<td>pixel</td>
<td>0.23</td>
<td>0.32</td>
<td>0.32</td>
</tr>
<tr>
<td>8 scenes types:</td>
<td></td>
<td>pyr</td>
<td>0.17</td>
<td>0.27</td>
<td>0.66</td>
</tr>
<tr>
<td>coast, mountain,</td>
<td></td>
<td>gist</td>
<td>0.19</td>
<td>0.23</td>
<td>0.47</td>
</tr>
<tr>
<td>forest, open</td>
<td></td>
<td>arousal</td>
<td>0.43</td>
<td>0.45</td>
<td>0.65</td>
</tr>
<tr>
<td>country, street,</td>
<td></td>
<td>complex</td>
<td>0.19</td>
<td>0.31</td>
<td>0.53</td>
</tr>
<tr>
<td>inside city,</td>
<td></td>
<td>colorful</td>
<td>0.24</td>
<td>0.33</td>
<td>0.67</td>
</tr>
<tr>
<td>tall building,</td>
<td></td>
<td>edges</td>
<td>0.30</td>
<td>0.34</td>
<td>0.51</td>
</tr>
<tr>
<td>highway</td>
<td></td>
<td>contrast</td>
<td>0.19</td>
<td>0.34</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td>Pref.</td>
<td>pixel</td>
<td>0.43</td>
<td>0.40</td>
<td>0.62</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pyr</td>
<td>0.64</td>
<td>0.78</td>
<td>0.70</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gist</td>
<td><strong>0.67</strong></td>
<td>0.75</td>
<td>0.76</td>
</tr>
<tr>
<td></td>
<td></td>
<td>colorhist</td>
<td>0.54</td>
<td>0.69</td>
<td><strong>0.83</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>combined</td>
<td><strong>0.71</strong></td>
<td>0.83</td>
<td>0.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td>comb. decorr.</td>
<td>0.70</td>
<td>0.83</td>
<td>0.68</td>
</tr>
<tr>
<td><strong>None</strong></td>
<td>Unusual</td>
<td>compose</td>
<td>0.10</td>
<td>0.35</td>
<td>0.46</td>
</tr>
<tr>
<td>Memorability[67]</td>
<td></td>
<td>pixel</td>
<td>0.01</td>
<td>0.31</td>
<td>0.65</td>
</tr>
<tr>
<td>Arbitrary photos:</td>
<td></td>
<td>pyr</td>
<td>-0.11</td>
<td>0.29</td>
<td>0.60</td>
</tr>
<tr>
<td>Indoor, Outdoor,</td>
<td></td>
<td>gist</td>
<td>-0.01</td>
<td>0.30</td>
<td>0.45</td>
</tr>
<tr>
<td>man-made,</td>
<td></td>
<td>arousal</td>
<td>-0.03</td>
<td>0.31</td>
<td>0.47</td>
</tr>
<tr>
<td>natural,</td>
<td></td>
<td>complex</td>
<td>0.27</td>
<td>0.42</td>
<td>0.63</td>
</tr>
<tr>
<td>people, animals</td>
<td></td>
<td>colorful</td>
<td>0.03</td>
<td>0.34</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td></td>
<td>edges</td>
<td>0.11</td>
<td>0.42</td>
<td>0.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>contrast</td>
<td>0.05</td>
<td>0.33</td>
<td>0.67</td>
</tr>
<tr>
<td></td>
<td>Pref.</td>
<td>pixel</td>
<td>0.25</td>
<td>0.51</td>
<td>0.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pyr</td>
<td>0.52</td>
<td>0.66</td>
<td><strong>0.78</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>gist</td>
<td><strong>0.58</strong></td>
<td>0.69</td>
<td><strong>0.77</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>colorhist</td>
<td>0.33</td>
<td>0.55</td>
<td>0.64</td>
</tr>
<tr>
<td></td>
<td></td>
<td>combined</td>
<td><strong>0.60</strong></td>
<td>0.73</td>
<td><strong>0.82</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>comb. decorr.</td>
<td><strong>0.60</strong></td>
<td>0.77</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Table 2.1: The interestingness cues and their performance on the 3 datasets. We highlight best combination and best and second best single feature.
3

Automatic Generation of Animated GIFs from Video

3.1 Chapter overview

After investigating the sources of interest, we turn to predicting video interestingness in this chapter. We introduce the novel problem of automatically generating animated GIFs from video, where the parts selected for GIFs can be considered the most interesting. We pose the question: Can we automate the entirely manual and elaborate process of GIF creation by leveraging the plethora of user generated GIF content? We propose a Robust Deep RankNet that, given a video, generates a ranked list of its segments according to their suitability as GIF. We train our model to learn what visual content is often selected for GIFs by using over 100K user generated GIFs and their corresponding video sources. We show that our approach is robust to outliers and picks up several patterns that are frequently present in popular animated GIFs. On our new large-scale benchmark dataset, we show the advantage of our approach over several state-of-the-art methods.

3.2 Introduction

Animated GIF is an image format that continuously displays multiple frames in a loop, with no sound. Although first introduced in the late 80’s, its popularity has increased dramatically in recent years on social networks, such as Tumblr and reddit, generating numerous famous Internet memes and creative Cinemagraphs [4]. In response, various websites have been created to provide easy-to-use tools to generate GIF from video, e.g., GIFSoup, Imgflip, and
Figure 3.1: Our goal is to rank video segments according to their suitability as animated GIF. We collect a large-scale dataset of animated GIFs and the corresponding video sources. This allows us to train our Robust Deep RankNet using over 500K pairs of GIF and non-GIF segment pairs, learning subtle differences between video segments using our novel adaptive rank Huber loss.

Ezgif. However, while becoming more prevalent, the creation of GIF remains an entirely manual process, requiring the user to specify the timestamps of the beginning and the end of a video clip, from which a single animated GIF is generated. This way of manually specifying the exact time range makes existing solutions cumbersome to use and requires extensive human effort.

In this chapter, we introduce the novel problem of automatically generating animated GIFs from video, dubbed Video2GIF. From the computer vision perspective, this is an interesting research problem because GIFs have some unique properties compared to conventional images and videos: A GIF is short, entirely visual with no sound, expresses various forms of emotions, and sometimes contains unique spatio-temporal visual patterns that make it appear to loop forever. The task has some connections to existing computer vision problems – such as visual interestingness (Chapter 2), creativity [139], video highlights [157, 187] and summarization [55, 155] – but differs from them due to
the unique properties described above. Apart from research interest, the task is supported by real-world demand and has many practical application scenarios including photojournalism, advertising, video sharing and preview, as well as video promotion on social media.

To handle this task, we propose a novel RankNet that, given a video, produces a ranked list of segments according to their suitability as animated GIF. Our framework has several novel components designed to learn what content is frequently selected for popular animated GIFs. First, to capture the highly dynamic spatio-temporal visual characteristics of GIFs, we use 3D convolutional neural networks [166] to represent each segment of a video. Second, to unravel the complex relationships and learn subtle differences between segments of a given video, we construct a ranking model that learns to compare pairs of segments and find the ones that are more suitable as GIF. Third, to make our learning task robust to the noisy web data, we design a new robust adaptive Huber loss function in the ranking formulation. Lastly, to account for different degrees of quality in user generated content, we encode the popularity measure of GIFs on social media directly into our loss.

Crucial to the success of our approach is our new large-scale animated GIF dataset: We collected more than 100K user generated animated GIFs with their corresponding video sources from online sources. There are hundreds of thousands of GIFs available online and many provide a link to the video source. This allows us to create a dataset that is one to two orders of magnitude larger than existing datasets in the video highlight detection and summarization literature [54, 155, 157]. We use this dataset to train our deep neural network by making comparisons between more than 500K GIF and non-GIF pairs. Experimental results suggest that our model successfully learns what content is suitable for GIFs, and that our model generalizes well to other tasks, namely video highlight detection [157].

In summary, we make the following contributions:

1. We introduce the task of automatically generating animated GIFs from video. This is an interesting computer vision research problem that, to the best of our knowledge, has not been addressed before.

2. We propose a Robust Deep RankNet with a novel adaptive Huber loss in the ranking formulation. We show how well our loss deals with noisy web data, and how it encodes the notion of content popularity to account for different degrees of content quality.
3. We collect a new, large-scale benchmark dataset of over 100K user generated animated GIFs and their video sources. The dataset is one to two orders of magnitude larger than existing video highlighting and summarization datasets. The dataset is publicly available\(^1\).

### 3.3 Related Work

Our work is closely related to image aesthetics and interestingness, as well as video highlight detection and summarization. We review some of the most relevant work and discuss the differences. We also review and make connections to recent efforts on learning deep neural networks for ranking and trained on large-scale weakly-labeled data.

**Image aesthetics and interestingness.** Finding the best images in a collection has been studied from several angles. Early approaches aimed at predicting the quality [76] or aesthetics [30, 33] of an image. More recently, several approaches for predicting visual interestingness of an image have been proposed [44, 52], including our work in Chapter 2. While interestingness is a subjective property assessed by the viewer, there is considerable consistency across annotated ratings (see Section 2.3). This makes it possible to model interestingness with computational means, but ground truth is typically noisy. Fu et al. [44] propose an approach accounting for this, by learning a ranking model and removing outliers in a joint formulation. Khosla et al. [77] analyze the related property of image popularity. Using a large-scale dataset of Flickr images, they analyze and predict what types of images are more popular than others, surfacing trends similar to those we observed for interestingness in Chapter 2. In a similar direction is the work of Redi et al. [139], which analyzes creativity. Rather than analyzing images, however, they focus on Vines videos, whose lengths are restricted to 6 seconds.

**Video summarization.** The task of video summarization is to select a subset of the frames of a video, such that the video is summarized well. As such, summarization is a structured prediction task. While a large body of work on summarization exists, we focus on the discussion of two recent trends here. (i) using web-image priors [78, 79, 107, 155, 186] and (ii) supervised learning-based methods [49, 55, 92, 107]. A thorough discussion of earlier research can

\(^1\)https://github.com/gyglim/video2gif_dataset
be found in [167]. Methods using web-image priors are based on the observation that web images for a specific topic or query are often canonical visual examples for the topic. This allows one to compute frame scores as the similarity between a frame and a set of web images [78, 79, 155]. Learning-based methods, on the other hand, use supervised models to obtain a scoring function for frames [49, 92, 107] or segments [55]. Lee et al. [92] learn a regression model and combine it with a clustering approach to diversify the results. Instead, [49, 55] directly learn an objective function that scores a set of segments, based on relative importance between different aspects of a summary (e.g. balancing highlights and diversity). Crucial to these learning-based methods is some notion of importance or interestingness of a segment. As such, automatic GIF creation or highlight detection can be seen as an important component of summarization. But they are not aiming to select a set of elements, i.e. are not structured prediction tasks. Next, we will discuss methods focusing only on detecting highlights, while ignoring other aspects of good summaries such as diversity and information coverage.

**Video highlights.** The definition of highlight is both subjective and context-dependent [167]. Nevertheless, it has been shown that there exists some consistency among human ratings for this task [54, 155]. Several methods exploit, for example, that close-ups of faces are generally of interest [54, 92, 167]. But these approaches are limited in that they rely on a few hand-crafted features for capturing highlights in highly diverse settings. Instead, several approaches for domain-specific models have been proposed. In particular, in sport games highlight is more clearly defined (e.g., scoring a goal) which has been exploited in many works (see [167] for an overview). Recently, Sun et al. [157] and Potapov et al. [135] proposed a more general approach. Based on annotated videos for a specific topic (e.g., surfing), they use machine learning on top of generic features to train a highlight predictor. In order to train their model, [135] uses a large, manually annotated dataset for action recognition. Instead, [157] use a smaller dataset obtained by crawling YouTube data. They find pairs of raw and edited videos, used in training, by matching all pairs of videos within a certain category (e.g., gymnastics). The size of their dataset is, however, limited by the availability of domain-specific videos in both raw and edited forms.

Obtaining a large-scale video highlight dataset is difficult. Thus, Yang et al. [187] propose an unsupervised approach for finding highlights. Relying on an assumption that highlights of an event category are more frequently cap-
tured in short videos than non-highlights, they train an auto-encoder. Our work instead follows a supervised approach, introducing a new way to obtain hundreds of thousands of labeled training videos (10x larger than the unlabeled dataset of [187]), which allows us to train a deep neural network with millions of parameters.

Learning to rank with deep neural networks. Several works have used CNNs to learn from ranking labels. The loss function is often formulated over pairs [50, 107] or triplets [63, 88, 180, 181]. Pairwise approaches typically use a single CNN, while the loss is defined relatively over the output. For example, Gong et al. [50] learn a network to predict image labels and require the scores of correct labels to be higher than the scores of incorrect labels. Triplet approaches, on the other hand, use Siamese networks. Given an image triple (query, positive, negative), a loss function requires the learned representation of the query image to be closer to that of the positive, rather than the negative image, according to some metric [63, 88, 180, 181].

Supervised deep learning from noisy labels. Several previous works have successfully learned models from weak labels [74, 107, 180]. Liu et al. [107] considers the video search scenario. Given click-through data from Bing, they learn a joint embedding between query text and video thumbnails in order to find semantically relevant video frames. In contrast, [74, 180] use labels obtained through automatic methods to train neural networks. Karpathy et al. [74] train a convolutional neural network for action classification in videos. Their training data is obtained from YouTube where it is labeled automatically by analyzing meta data associated with the videos. Wang et al. [180] learn a feature representation for fine-grained image ranking. Based on existing image features they generate labels used for training the neural network. Both approaches obtain state-of-the-art performance, showing the strength of large, weakly-labeled datasets in combination with deep learning.

3.4 Video2GIF Dataset

Inspired by the recent success with large, weakly-labeled datasets applied in combination with deep learning, we harvest social media data with noisy, human generated annotations. We use websites that allow users to create GIFs from video (Make-a-GIF and GIFSoup). Compared to edited videos used
### 3.4. Video2GIF Dataset

<table>
<thead>
<tr>
<th>Property</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of animated GIFs</td>
<td>121,647</td>
</tr>
<tr>
<td>Mean GIF duration</td>
<td>5.8 sec</td>
</tr>
<tr>
<td>Total number of videos</td>
<td>84,754</td>
</tr>
<tr>
<td>Mean video duration</td>
<td>7,379 hr</td>
</tr>
<tr>
<td>Total number of videos (CC-BY)</td>
<td>432</td>
</tr>
<tr>
<td>GIFs used in experiment</td>
<td>100,699</td>
</tr>
<tr>
<td>Videos used in experiment</td>
<td>70,456</td>
</tr>
</tbody>
</table>

#### Table 3.1: Statistics on the Video2GIF dataset. We show numbers for the complete dataset and for the one after discarding too short or too long videos (see text). We also show the number of videos that come with the Creative Commons license (CC-BY).

In [157], GIFs have the intriguing property that they are inherently short and focused. Furthermore they exist in large quantities and typically come with reference to the initial video, which makes alignment scale linearly in the number of GIFs. Aligning GIFs to their source videos is crucial, as it allows us to find non-selected segments, which serve as negative samples in training. In addition, videos provide a higher frame-rate and fewer compression artifacts, ideal for obtaining high quality feature representations.

Using these GIF websites, we collected a large-scale dataset with more than 120K animated GIFs and more than 80K videos, with a total duration of 7,379 hours. This is one to two orders of magnitude larger than the highlight datasets of [187] and [157]. We will show further statistics on the dataset after discussing the alignment process.

**Alignment.** We aligned the GIFs to their corresponding videos using frame matching. In order to do this efficiently, we encoded each frame with a perceptual hash based on the discrete cosine transform [192]. The perceptual hash is fast to compute and, given its binary representation, can be matched very efficiently using the Hamming distance. We matched the set of GIF frames to the frames of its corresponding video. This approach requires $O(nk)$ distance computations, where $n, k$ is the number of frames in the video and GIF, respectively. Since the GIFs are restricted in length and have a low frame-rate,
Figure 3.2: Length distribution of the input videos.

Figure 3.3: Most frequent video tags on the used dataset. We can observe that not all tags are equally informative. While several describe a specific visual concept (e.g., cat or wrestling) others describe abstract concepts that cannot be expected to help the task at hand.

Figure 3.4: Distribution over video categories. Note how the categories are highly imbalanced and often not specific. e.g., Entertainment is an extremely broad category with strong visual and semantic variation.
they typically contain only a few frames \((k < 50)\). Thus, this method remains computationally efficient while it allows for the alignment to be accurate.

In order to test the accuracy of our alignment process, we manually annotated a small random set of 20 GIFs with ground-truth alignments and measured the error. Our method has a mean alignment error of 0.34 seconds (median 0.20 seconds), which is accurate enough for our purpose. In comparison, Sun et al. [157] aligned blocks of 50 frames \((\approx 2 \text{ seconds})\), i.e. on a much coarser level.

**Dataset Analysis.** We analyze what types of video are often used to create animated GIFs. Figure 3.3 shows the most frequent tags of videos in our dataset, and Figure 3.4 shows the category distribution of the videos. Several tags give a sense of what is present in the videos, which can potentially help GIF creation, e.g., cute and football. Others are not visually informative, such as 2014 or YouTube. Figure 3.2 shows a histogram of video lengths (median: 2m51s, mean: 5m12s). As can be seen, most source videos are rather short, with a median duration of less than 3 minutes.

**Splits.** From the full dataset we used videos with a maximal length of 10 minutes. Longer videos are discarded as the selected GIF segments become too sparse and the videos are more affected by chronological bias [155]. We split the data into training and validation sets, with about 65K and 5K videos, respectively. For the test set, we use videos with Creative Commons license, which allows us to distribute the source videos for future research. As the task is trivial for videos shorter than 30sec we only consider videos of longer duration. The final test set consists of 357 videos. Table 3.1 shows the statistics of the dataset we used in our experiments.

## 3.5 Method

This sections presents our approach to the Video2GIF task, with a novel adaptive Huber loss in the ranking formulation to make the learning process robust to outliers; we call our model the Robust Deep RankNet.
3. Automatic Generation of Animated GIFs from Video

3.5.1 Video Processing

We start by dividing a video into a set of non-overlapping segments $S = \{s_1, \ldots, s_n\}$. We use the efficient shot boundary detection algorithm of Song et al. [155], which solves the multiple change point detection problem to detect shot boundaries.

The segments are not necessarily aligned perfectly with the boundaries of the actual animated GIF segments. We determine whether a segment $s$ belongs to GIF segment $s^*$ by computing how much of it overlaps with $s^*$. A segment is considered as a GIF segment only if the overlap is larger than 66%. Segments without any overlap serve as negatives. The segments are then fed into our robust deep ranking model, described next.
3.5. Method

3.5.2 Robust Deep RankNet

Architecture overview. Figure 3.5 illustrates the architecture of our model. During training, the input is a pair of GIF and non-GIF segments. The model learns a function \( h : \mathbb{R}^d \rightarrow \mathbb{R} \) that maps a segment \( s \) to its GIF-suitability score \( h(s) \). This score is of course unknown even during training; we learn the function by comparing the training segment pairs so that a GIF segment gets a higher score than a non-GIF segment. During testing, the model is given a single segment and computes its GIF-suitability score using the learned scoring function. We compute the score \( h(s) \) for all segments \( s \in S \) and produce a ranked list of the segments for their suitability as an animated GIF.

Feature representation. Animated GIFs contain highly dynamic visual content; it is crucial to have a feature representation that captures this aspect well. To capture both the spatial and the temporal dynamics of video segments, we use C3D [166] pretrained on the Sports-1M dataset [74] as our feature extractor. C3D extends the image-centric network architecture of AlexNet [85] to the video domain by replacing the traditional 2D convolutional layers with a spatio-temporal convolutional layer, and has been shown to perform well on several video classification tasks [166].

Inspired by previous methods using category specific models [135, 157], we optionally add contextual features to the segment representation. These can be considered meta-information, supplementing the visual features. They have the potential to disambiguate segment rankings and allow a model to score segments conditioned on the semantic category of a video. The features include the category label, a semantic embedding of the video tags (mean over their word2vec representation [117]) and positional features. For positional features, we use the timestamp, rank and the relative position of the segment in the video.

Problem formulation. A straightforward way to formulate our problem is by posing it as a classification problem, i.e., treat GIF and non-GIF segments as positive and negative examples, respectively, and build a binary classifier that separates the two classes of examples. This formulation, however, is inadequate for our problem because there is no clear cut definition of what is a good or a bad segment. Rather, there are various degrees of GIF suitability that can only be inferred by comparing GIF and non-GIF pairs. Indeed, a non-GIF segment is not truly a negative, but is likely just less interesting than the segments that were chosen.
A natural formulation is therefore posing it as a ranking problem. We can define a set of rank constraints over the dataset $\mathcal{D}$, where we require GIF segments $s^+$ to rank higher than non-GIF segments $s^-$, i.e.

$$h(s^+) > h(s^-), \quad \forall (s^+, s^-) \in \mathcal{D}.$$  

This formulation compares two segments even if they are from different videos. This is problematic because a comparison of two segments is meaningful only within the context of the video, e.g., a GIF segment in one video may not be chosen as a GIF in another. To see this, some videos contain many segments of interest (e.g., compilations), while in others even the selected parts are of low quality. The notion of GIF suitability is thus most meaningful only within, but not across, the context of a single video.

To account for this, we revise the above video-agnostic ranking formulation to be video-specific, i.e.

$$h(s^+) > h(s^-), \quad \forall (s^+, s^-) \in S.$$  

That is, we require a GIF segment $s^+$ to score higher than negative segments $s^-$ that come from the same video only. Next we define how we impose the rank constraints.

**Loss function.** One possible loss function for the ranking problem is an $l_p$ loss, defined as

$$l_p(s^+, s^-) = \max (0, 1 - h(s^+) + h(s^-))^p, \quad (3.1)$$

where $p = 1$ [69] and $p = 2$ [91, 157] are the most popular choices. The $l_p$ loss imposes the ranking constraint by requiring a positive segment to score higher than its negative counterpart by a margin of 1. If the margin is violated, the incurred loss is linear in the error for the $l_1$ loss, while for the $l_2$ loss it is quadratic. One drawback of the $l_1$ loss, compared to the $l_2$ loss, is that it over-penalizes small margin violations. The $l_2$ loss does not have such problem, but it quadratically penalizes margin violations, and thus is more affected by outliers (see Figure 3.6).

Our dataset contains animated GIF contents created by online users, so some of the contents will inevitably be of low quality; these can be considered as outliers. This motivates us to propose a novel robust rank loss, which is an adaption of the Huber loss formulation [64] to the ranking setting. This loss
gives a low penalty to small violations of the margin (where the ranking is still correct), and is more robust to outliers compared to the $l_2$ loss. We define our loss as

$$l_{\text{Huber}}(s^+, s^-) = \begin{cases} 
\frac{1}{2} l_2(s^+, s^-), & \text{if } u \leq \delta \\
\delta l_1(s^+, s^-) - \frac{1}{2} \delta^2, & \text{otherwise}
\end{cases}$$ (3.2)

where $u = 1 - h(s^+) + h(s^-)$. Thus, if the margin is violated, the loss corresponds to a Huber loss, which is squared for small margin-violations and linear for stronger violations. The parameter $\delta$ defines the point at which the loss becomes linear. We illustrate the three different forms loss functions in Figure 3.6.

Considering the source of our dataset (social media), not all GIFs are expected to be of equal quality. Some might be casually created by beginners and from mediocre videos, while others are carefully selected from a high quality source. Thus, some GIFs can be considered more reliable as positive examples than others. We take this into account by making the parameter $\delta$ GIF dependent: We assign a higher value to $\delta$ to more popular GIFs. Our intuition behind this adaptive scoring scheme is that popular GIFs are less likely to be outliers and therefore do not require a loss that becomes linear early on.
Objective function. Finally, we define our objective as the total loss over the dataset $D$ and a regularization term with the squared Frobenius norm on the model weights $W$:

$$L(D, W) = \sum_{S_i \in D} \sum_{(s^+, s^-) \in S_i} l_{\text{Huber}}(s^+, s^-) + \lambda \|W\|_F^2,$$ (3.3)

where $\lambda$ is the regularization parameter.

3.5.3 Implementation Details

We experimented with various network architectures. While the loss function turned out to be crucial, we empirically found that performance remains relatively stable for different depths of a network. Thus, we opt for a simple 2 hidden layer fully-connected model, where each hidden unit is followed by a ReLu non-linearity [123]. We use 512 units in the first and 128 in the second hidden layer. The final prediction layer, which outputs $h(s)$, is a simple single linear unit, predicting an unnormalized scalar score. The final network has 2,327,681 parameters.

We minimize the objective in Eq. 3.3 using mini-batch stochastic gradient descent with backpropagation [142]. We use mini-batches of 50 pairs. In order to accelerate convergence, we apply Nesterov’s Accelerated Momentum [8] for updating the weights. The momentum is set to 0.9 and $\lambda = 0.001$ (weight decay). We initialize training with a learning rate of 0.001 and reduce it every 10th epoch. The learning is stopped after 25 epochs. We apply dropout [156] regularization to the input (0.8) and after the first hidden layer (0.25). Dropout is a simple, approximate way to do model averaging that increases robustness of the model [156].

We obtain the training set segment pairs $(s^+, s^-)$ by using all positive segments, randomly sampling $k = 4$ negatives per video, and combining them exhaustively. We limit the negatives in order to balance the positive-negative pairs per video. Finally, we obtain 500K pairs for training. For the Huber loss with a fixed $\delta$ we set $\delta = 1.5$ based on the performance on the validation set. For the adaptive Huber loss, we set $\delta = 1.5 + p$, where $p$ is normalized viewcount proposed in [77].

In order to further decrease the variance of our model, we use model averaging, where we train multiple models from different initializations and average
their predicted scores. The models were implemented using Theano [10] with Lasagne [34].

## 3.6 Experiments

We evaluate our method against several state-of-the-art methods on our dataset. In Section 3.6.3 we further evaluate cross-task performance on the highlight dataset of [157].

**Evaluation metrics.** Two popular performance metrics used in video highlight detection are mean Average Precision (mAP) [157] and average meaningful summary duration (MSD) [135]. Both mAP and MSD are, however, sensitive to video length: the longer the video is, the lower the score (think about finding the needle in the haystack). To compensate for a variety of video lengths in our dataset (see Figure 3.2), we propose a normalized version of MSD.

The normalized MSD (nMSD) corresponds to the relative length of the selected GIF at a recall rate of $\alpha$. We define it as:

$$
nMSD = \frac{|G^*| - \alpha|G^{gt}|}{|V| - \alpha|G^{gt}|},
$$

where $|.|$ denotes the length of a GIF or video, and $G^*$ is the GIF with $\alpha$ recall w.r.t. the ground truth GIF $G^{gt}$. The score is normalized with the length of the ground truth GIF and the video $V$, such that it is 0 if the selection equals to $G^{gt}$ (is perfect), and 1 if the ground truth has the lowest predicted score. The added normalization helps make the scores of different videos more comparable, in contrast to mAP, which is strongly affected by the length of the ground truth GIF, relative to the video. To account for inaccuracies in segmentation we set $\alpha = 0.5$. For videos with multiple GIFs we use their mean nMSD as the video score. In addition to nMSD, we also evaluate performance using the traditional mAP.

### 3.6.1 Compared Methods

We compare our method to three state-of-the-art methods in highlight detection. We also provide an approximate upper bound that puts the results in perspective.
Domain-specific highlights [157]. We learn a domain-specific rankSVM per video category. Sun et al. [157] use an EM-like approach to handle long, loosely selected highlights. For our dataset, this problem does not occur because the GIFs are already short and focused. We therefore simply train a rankSVM [91] per video category using C3D features. We set $C = 1$ for all models.

Deep visual-semantic embedding [107]. We train a network using triplets of segment, true and random titles $(s^+, t^+, t^-)$. The titles are embedded into $\mathbb{R}^{300}$ using word2vec [117]. In contrast to our method, the loss of [107] is defined over positive and negative titles and uses only positive segments (or images in their case) for training.

Category-specific summarization [135]. This approach trains a one-vs-all SVM classifier for each video category. Thus, the classifier learns to separate one semantic class from the others. At test time it uses the classifier confidence to assign each segment an importance score, which we use to obtain a ranked list.

Approximate upper bound. This bound provides a reference for how well an automatic method can perform. To obtain the upper bound, we first find all videos in our dataset that have animated GIFs from multiple creators. We then evaluate the performance of one GIF w.r.t. the remaining ones from the same video. Thus, the approximate upper bound is the performance users achieve in predicting the GIFs of other users. And it allows us to put the performance of automatic methods in perspective. We note, however, that this bound is only approximate because it is obtained in a very different setting than other methods.

3.6.2 Results and Discussions

Table 3.2 summarizes the results. Figure 3.7 shows qualitative results obtained using our method. As can be seen, our method (“Ours” in Table 3.2) outperforms the baseline methods by a large margin in terms of nMSD. The strongest baseline method is domain-specific rankSVM [157]. Their learning objective is similar to ours, i.e., they use pairs of positive and negative segments from the same video for training. In contrast, two other baselines [107, 135] use a “proxy” objective, i.e., learn semantic similarity of segments to video category [135] or segments to video title [107]. We believe this different training
3.6. Experiments

<table>
<thead>
<tr>
<th>Method</th>
<th>nMSD ↓</th>
<th>mAP ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joint embedding [107]</td>
<td>54.38%</td>
<td>12.36%</td>
</tr>
<tr>
<td>Category-spec. SVM [135]</td>
<td>52.98%</td>
<td>13.46%</td>
</tr>
<tr>
<td>Domain-spec. rankSVM [157]</td>
<td>46.40%</td>
<td>16.08%</td>
</tr>
<tr>
<td>Classification</td>
<td>61.37%</td>
<td>11.78%</td>
</tr>
<tr>
<td>Rank, video agnostic</td>
<td>53.71%</td>
<td>13.25%</td>
</tr>
<tr>
<td>Rank, $l_1$ loss</td>
<td>44.60%</td>
<td>16.09%</td>
</tr>
<tr>
<td>Rank, $l_2$ loss</td>
<td>44.43%</td>
<td>16.10%</td>
</tr>
<tr>
<td>Rank, Huber loss</td>
<td>44.43%</td>
<td><strong>16.22%</strong></td>
</tr>
<tr>
<td>Rank, adaptive Huber loss</td>
<td>44.58%</td>
<td>16.21%</td>
</tr>
<tr>
<td>Rank, adaptive Huber loss + context (Ours)</td>
<td>44.19%</td>
<td>16.18%</td>
</tr>
<tr>
<td>Ours + model averaging</td>
<td><strong>44.08%</strong></td>
<td>16.21%</td>
</tr>
<tr>
<td>Approx. bounds</td>
<td>38.77%</td>
<td>21.30%</td>
</tr>
</tbody>
</table>

Table 3.2: Experimental results. A lower nMSD and higher mAP represent better performance.

The objective is crucial, allowing both our method and rankSVM [157] to significantly outperform the two baselines.

Domain-specific rankSVM [157] with C3D features performs fairly well; but our method outperforms it. We believe the reason for this performance difference is two-fold: (1) the $l_2$ loss in [157] is not robust enough to outliers; and (2) the learning capabilities of [157] are limited by the use of a linear model, compared to highly nonlinear neural nets. Next, we analyze different configurations of our method in greater detail and discuss impacts of each design choice. The configurations differ in terms of used inputs, network architecture and objective.

**What loss function is most robust?** We analyze performance with different loss functions and training objectives discussed in Section 3.5.2. As expected, classification models always performs poorly compared to ranking models. Also, using video agnostic training data performs poorly. This indicates that the definition of a highlight is most meaningful within the video. When comparing $l_1$ loss and $l_2$ loss, we find that $l_1$ loss penalizes small margin violations \((i.e., 0 < h(s^+) - h(s^-) < 1)\) too strongly, while the $l_2$ loss is affected by outliers. Our Huber rank loss avoids the two issues by combining the robustness...
Figure 3.7: **Qualitative results.** Examples of top 3 and bottom 3 predicted segments. Our approach picks up aspects that are related to GIF suitability. For example, it learns that segments with people in motion are suitable for GIFs (e.g., (a) and (c)), while low contrast segments without any (main) objects are not (e.g., (a) the 4th image). It also scores segments showing the goal area of soccer games higher than the crowd in the stadium (b). We show a failure case (d): the network scores the segments with people on the ground higher than the landing plane (4th image).

to outliers ($l_1$ loss) and the decrease in the gradient for small margin violations ($l_2$ loss); it thus performs better than the other losses.

**The role of context.** Inspired by previous methods using category specific models [135, 157] we used contextual information as input to our model (category label, a semantic of the video tags and positional features). When comparing the performance with and without context, we find that they perform similarly (Table 3.2). We believe that most of the information about the context is already present in the segment representation itself. This is supported by [105] who show that the context can be extracted from the segment itself with high accuracy.

### 3.6.3 Cross Dataset Performance

As discussed, automatic GIF creation is related to video highlight detection. Of course, they are not identical: GIFs have a different focus and often de-
Table 3.3: Cross-dataset results (mAP). We train on our dataset and test on the video highlight dataset of [157]. Our method outperforms rankSVM and [187], which learns an unsupervised model for each domain. Sun et al. [157] performs best, but it is directly trained on their dataset and learns multiple models, one per category. Instead, we learn a single global model for GIF suitability.

We evaluate our model on the dataset of [157], which contains videos from hand-selected categories such as surfing and skiing. We also evaluate the best performing baseline, domain-specific rankSVM [157], trained on our dataset and tested on the highlight dataset. The results are summarized in Table 3.3 (we borrow previously reported results [157, 187]).

Our method outperforms rankSVM by a large margin, which suggests that our model generalizes much better than the baseline method. It also significantly outperforms the method of Yang et al. [187], which trains an auto-encoder model for each domain. Sun et al. [157] tops the performance, but they use video category labels (which are handpicked) and learn multiple models, one per category, directly on the highlight dataset. Instead, our method learns a single global model on the GIF data, with much more diverse video categories. Nonetheless, it shows competitive performance.

3.7 Conclusion

We introduced the problem of automatically generating animated GIFs from video, and proposed a Robust Deep RankNet that predicts the GIF suitability
of video segments. Our approach handles noisy web data with a novel adaptive Huber rank loss, which has the advantage of being robust to outliers and able to encode the notion of content quality directly into the loss. On our new dataset of animated GIFs we showed that our method successfully learns to rank segments with subtle differences, outperforming existing methods. Furthermore, it generalizes well to highlight detection.

Our novel Video2GIF task, along with our new large-scale dataset, opens the way for future research in the direction of automatic GIF creation. For example, more sophisticated language models could be applied to leverage video meta data, as not all tags are informative. Thus, we believe learning an embedding specifically for video tags may improve a contextual model. While this work focused on obtaining a meaningful ranking for GIFs, we only considered single segments. Since some GIFs range over multiple shots, it would also be interesting to look at when to combine segments or even do joint segmentation and selection.
4

Summarization by Learning Mixtures of Objectives

4.1 Chapter overview

The previous chapters focused on understanding interest and selecting the single best image or video segment. The goal of summarization, however, is select multiple segments, in order to create a short overview that still conveys the story of the initial video. The summary should thus be both, interesting and representative for the input video. We now present a novel method that allows to create these kind of summaries, using a model of visual interestingness, as introduced above, as one of its components.

Previous methods often used simplified assumptions or only optimized for one of the objectives of a good summary, e.g., interestingness (cf. Chapter 3 and [135, 157]). Alternatively, they used hand-defined objectives that were optimized sequentially by making consecutive hard decisions [78, 92]. This limits their use to a particular setting. Instead, this chapter introduces a method that (i) uses a supervised approach in order to learn the importance of global characteristics of a summary and (ii) jointly optimizes for multiple objectives and thus creates summaries that posses multiple properties of a good summary. Experiments on two challenging and very diverse datasets demonstrate the effectiveness of our method, where we outperform or match current state-of-the-art.
4.2 Introduction

With the success of mobile phones, activity cameras, Google Glass, activity cameras, etc. video recording devices have become omnipresent. As a consequence, vast amounts of videos are recorded every day to capture special moments or log daily activities. At the same time, with video capture becoming so easy and cheap, and with the strongly egocentric viewpoints that the devices often induce, videos are recorded casually. As in digital photography, many users follow a capture first, filter later mentality, where little thought is spent on timing, cutting, content and view selection. As a result, such casual videos are too long, shaky, redundant and low-paced to watch in their entirety. Therefore, reducing videos to their gist and removing bad parts is of increasing importance. As a result, video summarization, which automates this process, has gained a lot of attention in the last few years [3, 54, 78, 79, 92, 110, 135, 157, 196].

Automatically creating skims is challenging, as even a strongly shortened version should still convey the story of the initial video. A good summary must comply with at least two objectives [167]. Firstly, it should contain the most interesting parts of a video e.g., in a base jumping video one doesn’t want to miss highlights such as the start or landing. Secondly, the summary should be representative in keeping the diversity of the original, while removing redundancy.

Many recent methods predict a score per segment and ignore the structure of the video [54, 135, 157], and therefore have difficulties to jointly optimize both objectives. Methods that go in this direction typically cluster the video into events and select the most important segment(s) per event [78, 92], following a kind of successive optimization of the objectives. Others optimize diversity only locally using a Markov assumption [49]. Instead, our method optimizes for multiple objectives globally, avoiding hard decisions early on. Rather than using supervision only for some components [92] or making simplifying assumptions [54, 135, 157], our method learns the importance of summarization objectives directly from reference summaries created by human annotators, as depicted in Fig. 4.1. Using supervision for the task of video summarization is crucial, since it is extremely complex and highly task-dependent – summaries from surveillance or life-logging data are expected to meet different criteria than summaries of short clips obtained by a mobile phone. Our approach is able to automatically adapt to the type of video and the desired output. It is therefore much more general and can be applied in all of these settings. In-
Figure 4.1: **Overview.** Our method consists of two parts: A supervised learning stage (training) and inference (testing). Given pairs of videos and their user created summaries as training examples, we learn a combined objective. Then, when given a new video as input, our method creates summaries that are both interesting and representative.

Indeed, our experiments show that our method obtains state of the art performance in summarizing hour long life-logging videos [92], as well as short user videos [54].

## 4.3 Related Work

Videos can be summarized into many different representations: Keyframes [78, 79, 92, 185], skims [54, 110], storyboards [47], time-lapses [82], montages [158] or video synopses [136]. Here, we focus on approaches for generating and evaluating *skims* (dynamic video summaries)\(^1\), *i.e.* methods that output a shortened version of the initial video, rather than transforming the video into *e.g.*, a collection of images. Skims have the advantage that they retain motion information and can provide a nice viewing experience. Following Truong and Venkatesh [167], we review related work categorized into methods optimizing for (i) the preservation of interesting segments and (ii) representativeness of the summary. Further, we (iii) analyze methods optimizing for multiple objectives.

---

\(^1\)For a systematic and detailed review of existing techniques, the readers are referred to [167].
**Interestingness/relevance.** In order to select keyframes or segments for a summary, many methods predict the importance score for each keyframe or segment. This is typically formulated as a regression (e.g., Chapter 2) or ranking problem (e.g., [157] and Chapter 3). Thereby some features are extracted from a video segment, in order to predict its relevance. For this, Potapov et al. [135] use videos annotated for a certain event category. Instead, Sun et al. [157] and our work in Chapter 3 mine YouTube videos in order to train a model. Thereby both use the correspondence between the raw and edited version of a video in order to obtain labels for training. This is based on the assumption, that segments contained in the edited version are more relevant than the ones that are not. These method are however not evaluated in terms of summary quality, but rather in terms of their ability to rank segments for a certain category [135, 157], a criterion for which the overall structure of the video and the summary plays a minor role.

**Representativeness.** While optimizing for interestingness ignores the global structure of a summary, optimizing for representativeness only risks leaving out the most crucial event(s). Therefore only a few approaches in this area exist. Li and Merialdo [97] adapt the Maximal Marginal Relevance (MMR) approach [18] from the text to the video domain. This approach greedily selects a summary using an objective that optimizes for relevance w.r.t. the input video and penalizes redundancy within the summary. [196] uses sparse coding, in order to create a dictionary that serves as a summary. This method is particularly useful for longer videos, as it can be run in an online fashion.

**Multi-objective.** Several methods optimize for multiple objectives. Khosla et al. [78] use web priors to predict relevance. Thereby they cluster web images to learn canonical viewpoints as used in a specific domain (e.g., cars). In order to create a summary, they select the most central video frame per cluster. This way, the keyframes are similar to web images, while the summary remains diverse. Kim et al. [79] combine web priors with sub-modular maximization. They formulate the problem as a subset selection in a graph of web images and video frames. Given this graph, they optimize an anisotropic diffusion objective to select a set of densely connected but diverse nodes. This leads to summaries that strike a balance between relevance to the event and representativeness within the video. Lee et al. [92] propose a comprehensive method for summarization of egocentric videos. They introduce a method that clusters the video into events using global image features and a temporal regularization, which ensures that clusters are compact in time. For each cluster they predict
4.3. RELATED WORK

Table 4.1: Taxonomy of the most recent and relevant video summarization methods. We differentiate in terms of objectives they optimize and how they combine multiple objectives. Many methods score segment locally. Others combine multiple objectives, but do so based on a hand-defined sequential optimization. In opposition, we learn the importance of each objective from data and optimize them jointly.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Interesting</td>
<td>√</td>
<td>√</td>
<td>(√)</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Representative</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>√</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Uniform</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(√)</td>
<td>-</td>
<td>√</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comb.</th>
<th>Learnt weights</th>
<th>Optimized jointly</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>√</td>
</tr>
</tbody>
</table>

the importance of the objects it contains and select the most important ones for the final summary. As our work, Li et al. [96] uses a structured learning formulation, but focuses on transfer learning from text and has no approximation guarantees, since it doesn’t restrict the objectives to be submodular [102].

We summarize the most related works in a taxonomy in Tab. 4.1. Thereby we analyze the objectives used for each method and how these objectives are combined and optimized. While existing methods focused on interestingness or representativeness, we also find temporal distribution of the summary to be important. In line with [78], we observe (Sec. 4.6.1), that uniform sampling provides a strong baseline, typically outperforming clustering based approaches. Uniform sampling, as naïve as it is, retains temporal coherence and thus gives a good sense of the story of the initial video. Many previous methods made simplified assumptions or defined an objective based on heuristics. Instead, we follow a supervised learning approach, where we learn the importance of the different objectives. Given a new video, these objectives are optimized jointly to create a summary. Following our work, and the work of [49], learning to summarize in a supervised setting has become a popular approach. Thereby several recent works have been using submodular mixtures [133, 175] or DPPs [145, 195].
Evaluation. Objectively evaluating a summary is a hard task, as there is not one true summary, but rather many ways to summarize a video well. Early methods used user studies, where viewers were asked to score \[92, 110\] or compare \[130\] automatically generated summaries. A consensus has grown that videos should be evaluated automatically to simplify evaluation and comparison \[54, 135, 167, 191\]. This is either done in the video \[54\] or text domain \[191\] using multiple reference summaries. Gygli et al. \[54\] evaluate using the frame overlap between an automatically generated summary and some reference summaries. As different summaries with a practically equivalent semantic meaning are possible, they use a large number of human annotated reference summaries per video to reflect this ambiguity. Instead, Yeung et al. \[191\] map a video summary into text and use an existing text summarization evaluation \[99\]. This has the advantage, that summaries are compared in terms of semantics. It however also means that the evaluation does not take into account visual aspects such as shaky cameras, etc., as long as a certain content is depicted.

4.4 Structured prediction with submodular functions

We formulate the task of video summarization as a subset selection problem. We are given a video \(V\) and a budget \(B\). Let \(\mathcal{Y}_V\) denote the set of all possible solutions \(y \subseteq V\) given this constraint.

The task of our method is to select a summary \(y^*\), such that it optimizes an objective \(o\):

\[
y^* = \arg \max_{y \in \mathcal{Y}_V} o(x_V, y),
\]

(4.1)

where \(x_V\) are features extracted from the video \(V\). We define \(o(x_V, y)\) as a linear combination of objectives \(f(x_V, y) = [f_1(x_V, y), f_2(x_V, y), ..., f_n(x_V, y)]^T\), each capturing a different aspect of a summary:

\[
o(x_V, y) = w^T f(x_V, y).
\]

(4.2)

The objectives are defined in Sec. 4.5. Since \(\mathcal{Y}_V\) is growing exponentially with the length of the video, optimally solving Eq. (4.2) quickly becomes intractable. Therefore, we restrict the objectives \(f(x_V, y)\) to be monotone submodular and \(w\) to be non-negative. This allows to find a near optimal solution for Eq. (4.1) in an efficient way \[125\].
Next, we give a brief overview of submodular maximization and show how to learn the weights $w$. Then, Sec. 4.5 proposes functions $f(x_V, y)$ adapted to the problem of video summarization.

### 4.4.1 Submodular maximization

Set functions are submodular if they fulfill the diminishing returns property, i.e. given arbitrary sets $T \subseteq U \subseteq V \setminus \{s\}$ and a set function $f$, $f$ is submodular, if it satisfies: $f(T \cup \{s\}) - f(T) \geq f(U \cup \{s\}) - f(U)$. Linear combinations of submodular functions are also submodular for non-negative weights [84].

Submodular functions offer several properties desirable for optimization. It has been shown by Nemhauser et al. [125] that maximizing a monotonous submodular function under cardinality constraints with a greedy algorithm yields a good approximation of the optimal solution: the score of the greedy solution is lower bounded by $\frac{1}{e} (\approx 63\%)$ times the optimal value [125]. With cost constraints, i.e. the submodular knapsack problem, the greedy algorithm can perform arbitrarily bad. However Leskovec et al. [93] showed that by solving a standard and a cost-benefit greedy optimization and selecting the solution with the higher score, this is lower bounded by $\frac{1}{2} \frac{e-1}{e}$ times the optimal value.

In practice, however, the greedy solution often performs much better, with an approximation factor close to 1 [100] and can be speeded up with lazy evaluations [119]. These properties are crucial for the task at hand, in order to have a scalable algorithm. In our work, we use the algorithm of [93] with lazy evaluations [119] to optimize Eq. (4.1), shown in Algo. 1.

For more information on submodular function maximization we refer the reader to [84].

### 4.4.2 Learning

Given $T$ pairs of a video and a reference summary $(V, y_{gt})$, we learn the weight vector $w$ of Eq. (4.2). Thereby we optimize the following large-margin formulation:

$$
\min_{w \geq 0} \frac{1}{T} \sum_{t=1}^{T} \hat{L}_t(w) + \frac{\lambda}{2} \|w\|^2, \quad (4.3)
$$

where $\hat{L}_t(w)$ is the generalized hinge loss of training example $t$ [102]:
\[
\hat{L}_t(w) = \max_{y \subseteq Y_T} (w^T f(x_T^{(t)}, y) + l_t(y)) - w^T f(x_T^{(t)}, y_{gt}^{(t)}),
\] (4.4)

where we use superscript \((t)\) to refer to the features and subsets of video \(t\). The intuition behind this objective is that each human reference summary \(y_{gt}^{(t)}\) should score higher than any other summary by some margin. Given the complexity of the subset selection problem, finding the best scoring element in Eq. (4.4) can only be done approximately, as discussed above. We therefore resort to approximately learning and optimizing the objective using projected subgradient descent [102].

For the margin, we propose a recall loss, similar to the one used in [102] for text summarization:

\[
l_t(y) = \frac{1}{B} \left( |y| - \left| y \cap y^{(t)} \right| \right),
\] (4.5)

i.e. it is a count of how many of the candidate summary \(y\) are not represented in the ground truth, normalized by the maximal length of the summary. We found this to work best in our experiments, but other loss functions are also possible.

Summarizing, the problem of subset selection is difficult to optimize. But if the optimization can be posed as submodular maximization, we have seen that there exist efficient algorithms, which yield good approximations.

### 4.5 Submodular functions for video summarization

Submodular functions have already been used for summarization problems, e.g., for document [101, 102, 103] and also image collection [148, 168] and video summarization using keyframes [79]. This is not a coincidence, since summarization inherently has a diminishing returns property: The more segments that have already been selected from a video, the less an additional segment helps to get a better overview.

Defining submodular functions for the task of video summarization is not straightforward, however. While sentences of a document can be compared relatively easy, e.g., by n-gram overlap, the problem of finding a semantic similarity between video segments is largely unexplored. While the dominant theme of
a text can be found based on frequent sentences (n-grams), finding frequent
visual content does not suffice to create a good summary. Even persons or ob-
jects appearing only for a short period of time can be of high importance for
the whole video. It is therefore insufficient to optimize representativeness as
for document summarization [102]. Additional measures need to be used to
score video segments.

In the following, we define several submodular functions, aimed at captur-
ing the quality of a summary. Since our method creates skims, we use seg-
ments as the atomic entities, i.e. a video is defined as a set of segments:
\( \mathcal{V} = \{ s_1, s_2, \ldots, s_n \} \) from which we select a subset \( y^* \subset \mathcal{V} \).

**Interestingness.** Following existing approaches, we predict the importance
of a segment locally, i.e. without taking into account the rest of the video.
Specifically, we want to predict a score \( I(k) \) of each frame \( k \) given its features \( x_k \). This prediction might come from a general interestingness model as in [54, 92], or from a model that predicts a score of domain relevance, as in [157]. To
allow for overlapping segmentation’s, we use the union of frames in \( y \) and
score them with \( I(k)^2 \). We use

\[
\text{f}_{\text{imp}}(x_\mathcal{V}, y) = \sum_{k \in \bigcup_{s \in y} I(k), \quad (4.6)}
\]

where \( s \) is a segment in the solution \( y \). This function is called a weighted
coverage function, which is known to be submodular [84].

**Representativeness.** This function scores how well a summary represents
the initial video. While many existing methods clustered the video into events, we
believe this is not appropriate for raw videos, as they are continuous and there-
fore have gradual changes between locations and events. Instead, we propose
an objective that favors representative solutions while avoiding a hard cluster-
ing.

Finding the best \( k \) segments to represent a dataset is known as the k-medoids
problem. Its objective is to select a set of medoids, such that the sum of squared
errors between the datapoints and the nearest selected medoid is minimal, i.e.

\[
L(y) = \sum_{i \in \mathcal{V}} \min_{s \in y} ||x_i - x_s||_2^2, \quad (4.7)
\]

\(^2\text{For the case of non-overlapping segmentations, this simply becomes: } f_{\text{imp}}(x_\mathcal{V}, y) = \sum_{s \in y} I(s), \text{ i.e. it is modular and a score can be assigned to a segment directly, which is more computationally efficient.}\)
where we use as features $x_i$ global image features averaged over the segment frames. This objective can be reformulated as a submodular objective as follows:

$$f_{rep}(x_V, y) = L(\{p'\}) - L(y \cup \{p'\}),$$

(4.8)

where $p'$ is a phantom exemplar [48], necessary to avoid taking the minimum over an empty set in Eq. (4.7).

**Uniformity.** As a good summary tells the story of the input video, it needs to retain temporal coherence. Large jumps ahead can confuse a viewer. Similarly, a summary with many temporally adjacent segments risks being redundant. In order to avoid such problems, we propose a uniformity objective, using the same form as representativeness:

$$f_{uni}(x_V, y) = L(\{p'\}) - L(y \cup \{p'\}),$$

(4.9)

where we represent a segment using its mean frame number, i.e. the features $x_i$ are single scalars in this case. This objective scores how well the temporal dimension is represented by the solution $y$, effectively leading to solutions that are more uniformly distributed over the video.

Using these objective functions, we can now estimate the summarization objective Eq. (4.2). Given a set of videos and their summaries as training examples, we learn the importance of each function by optimizing Eq. (4.3). In the next section, we evaluate the summaries generated by our method and compare them to existing works.

### 4.6 Experiments

We evaluate the performance of our method and its individual components using two datasets: (i) the egocentric dataset of [92] and (ii) the SumMe dataset [54]. These datasets are extremely diverse: While the SumMe dataset consists of short user videos, the egocentric dataset contains hour long life-logging data from wearable cameras. Therefore, we analyze them separately in Sec. 4.6.1 and Sec. 4.6.2.

**Evaluation.** We evaluate w.r.t. the nearest-neighbor summary, i.e. the one that is the most similar to the automatically created one. This helps to account
for the fact that there exists not a single ground truth summary, but multiple summaries are possible. This approach was also used in ROUGE [99], which is the standard metric in document summarization. We follow [54, 99, 191] and report the recall and f-measure, motivated by the fact that including crucially important events in more important than having perfect precision.

**Compared methods.** We compare to several baselines, as well as state of the art methods: (i) Uniform sampling, (ii) a previous method for the used dataset (SumMe: [54], egocentric: [92]) and (iii) Video MMR (Maximal Marginal Relevance) [18]. MMR, initially proposed for document summarization, was adapted to the video domain by [97]. It uses a greedy maximization of an objective that favors representativeness and penalizes redundancy of elements within the summary. We use the approach of [97], but with deep features [36], rather than SIFT+BoW to compute affinities between segments.

**Implementation details.** To extract the representativeness of a segment, we compute deep features trained on ImageNet [36]. We use deep features, as they are the state of the art visual features. Since they are trained for object classification, they capture objects of a scene. We used layer 6 of DeCAF [36], which has show the best performance on various recognition tasks. For Eq (4.8) and Eq. (4.9), we use a phantom element \( p' \), which has the same distance to all points in the dataset. For this, we take the mean distance of the data points. Since the learning process receives the data points in random order, the output is also non-deterministic. Therefore we run learning and inference 100 times and average the results. We do the same for all objectives, since some might give the same score to multiple segments, i.e. there multiple elements might have a maximal gain (see Algo. 1, Line 13/15) We use cross-testing with 4 and 12 splits, respectively. All objectives were normalized such that the function values lie within \([0, 1]\).

### 4.6.1 Egocentric daily life dataset

The egocentric dataset of [92] contains 4 videos from wearable cameras. These videos log the day of the camera-wearer and have a duration of 3-5 hours, each, amounting to over 17 hours of video. The dataset does not include video reference summaries, but was annotated in [191] using text. Given the textual annotations for each segment of the video, a video summary can be mapped
into the textual domain. There, it is compared to reference summaries using the ROUGE [99] evaluation package. We use the same ROUGE parameters as [191]. Since our method requires reference summaries to train, it also requires an inverse mapping. We follow [191] and generate video summaries using a greedy bag of words and an ordered subshot method. In order to obtain multiple summaries, we vary the parameters (the n for the n-gram scoring as well as the order and maximal jump in the ordered subshot). We score these and remove the bottom 25%. Finally, we obtain 60 reference summaries (15 per video).

In order to predict the interestingness of a segment (Eq. (4.6)), we train an classifier using deep features [36] and the training data provided by [92]. Rather than learning to classify an image region as [92], we only learn to classify whether a frame contains important objects or not. We learn a linear classifier and use its prediction confidence as an importance estimate. While more sophisticated temporal segmentation’s are possible, e.g., [134], we use uniform segments with a length of 5 seconds. Since [191] provides annotation for the same segmentation, this allows for a non-ambiguous mapping to the textual domain in the evaluation.

As in [191], we produce and evaluate summaries of 2 minutes. In order to be comparable with [92], we also report on results for summaries having the same length as theirs.
Figure 4.2: **Learnt weights per objective:** We can observe how the learning algorithm adapts to the specific summary length: While interestingness, *i.e.* a local prediction of importance for each segment, is the most important objective for shorter lengths, having a representative and well distributed solution becomes more important, as the summaries get longer. Weights averaged over the whole dataset.

**Results.** We evaluated our method using textual reference summaries. Quantitative results are shown in Tab. 4.2 & 4.3, while Fig. 4.3 & 4.4 show qualitative examples in text and visual form. We observe that uniform sampling quantitatively outperforms all previous methods. Our approach, which learned a combination of objectives, however, produces summaries that are better than all compared methods. This is despite the fact, that the used evaluation metric only measures semantic summary quality and ignores whether a particular segment is a good representative for a certain event or has bad quality/motion blur. There, our method, as well as [92], has an additional advantage over uniform sampling, which might perform well in terms of semantics, but often selects visually inferior segments. Our method is able to learn the importance of the individual objective for this difficult task. Previous methods, on the other hand, are outperformed by uniform sampling. This might be a result of the hand-designed objectives. In particular it seems that [92] places too much emphasis on objects, such as phones, TV’s, etc. These however tell little about the story of the camera-wearer, for which people and scenes seem more im-
Table 4.3: **Egocentric dataset.** We use the same summary length as Lee *et al.* [92], in order allow a comparison (≈ 1 minute and 20 seconds)

<table>
<thead>
<tr>
<th>Method</th>
<th>F-measure</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>19.44 ± 2.56%</td>
<td>13.76 ± 1.99%</td>
</tr>
<tr>
<td>Uniform</td>
<td>21.37 ± 1.88%</td>
<td>15.06 ± 1.48%</td>
</tr>
<tr>
<td>Lee <em>et al.</em> [92]</td>
<td>17.40 ± 4.07%</td>
<td>12.20 ± 3.30%</td>
</tr>
<tr>
<td>Video MMR [97]</td>
<td>17.73 ± 0.00%</td>
<td>12.49 ± 0.00%</td>
</tr>
<tr>
<td>Uniformity</td>
<td>18.75 ± 1.36%</td>
<td>12.92 ± 1.11%</td>
</tr>
<tr>
<td>Interestingness</td>
<td>20.93 ± 0.00%</td>
<td>15.15 ± 0.00%</td>
</tr>
<tr>
<td>Representative</td>
<td>19.08 ± 0.00%</td>
<td>12.95 ± 0.00%</td>
</tr>
<tr>
<td>Combined</td>
<td>21.91 ± 0.06%</td>
<td>15.73 ± 0.04%</td>
</tr>
</tbody>
</table>

important. We show a quantitative comparison of uniform sampling, [92] and our method in Fig. 4.5. That uniform sampling provides such a strong baseline can be explained by the type of video: Such videos are very slow paced and do contain few highlights. It is therefore more important to give an overview over a camera-wearers day, for which uniform sampling is a simple, but reasonable approach. But uniform sampling does not take into account visual quality, which can be problematic, especially when selecting keyframes rather than segments as done in [92].

What is interesting to observe is how the learning also adapts to different summary lengths (see Fig 4.2). While for shorter summaries, interestingness is dominant, representativeness and uniformity get more weight for longer summaries. Thus, in short summaries the method focuses more on highlights, while it avoids getting redundant in longer summaries and therefore gives more weight to selecting representative and well distributed segments.

### 4.6.2 User video dataset

The SumMe dataset [54] consists of short user videos (1 to 7 minutes). These depict a certain event of interest, *e.g.*, a plane landing or a base jump. The dataset contains 25 videos, each annotated with ≥ 15 user summaries (390 reference summaries). The annotation was created in a controlled environment, where users were asked to create their own summary for a given video. To evaluate the generated summaries, we compute the overlap with these user
4.6. **Experiments**

Figure 4.3: **Egocentric dataset, Video P1**: The selected segments of the interestingness objective and our method (shown in blue). Using multiple competing objectives helps to regularize the summarization. This leads to summaries that are more representative, while still laying focus on the most most “interesting” parts (Also see Fig. 4.2). Using the interestingness objective only leads to a redundant summary, where, in this example, 4 of 16 segments depict the visit in a shoe store. Thus, it misses other substantial event from the initial video.

sumaries using the code provided\(^4\). For learning, we can directly use the user summaries of the training videos, as they already are in the video domain.

In order to predict the interestingness of a segment, we use the method of [54] with the same superframe segmentation. Given our submodular formulation however, it is not necessary to pre-commit to a fixed set of disjoint segments (see Eq. (4.6)). We therefore run the superframe segmentation with multiple initializations. We follow [54] and generate summaries with a maximal length of 15%.

**Results.** Quantitative results are shown in Tab. 4.4. Given the content of the videos (they mostly capture some interesting event) and the length of the videos in this dataset, interestingness prediction is the dominant objective. Therefore, adding structure only leads to marginal gains over a local prediction approach. This is also reflected in the learnt weights, where interestingness has a weight of 97.5%. Nonetheless, our method learns the dominance of interestingness over the other objective and still obtains a marginal improvement over each individual objective. Please note that while [54] uses a knapsack optimization,

\(^4\)[http://vision.ee.ethz.ch/~gyglim/vsum/](http://vision.ee.ethz.ch/~gyglim/vsum/)
Figure 4.4: **Egocentric dataset, Video P1**: Textual representation of a summary created with our method. Our method selects expressive segments from each of the main events and the travel between them. It tells the same story as the summary created by a human annotator. Please also be aware of the fact that the reference summaries are not extractive, *i.e.* they can contain formulations and sentences that are not in the annotation and can thus never be selected. One consequence of this are more repetitive sentences in the automatic summary. We give a keyframe visualisation for the same video in Fig. 4.3.

we use a greedy procedure. On these short videos with segments of variable length, a knapsack optimization yields lengths closer to the given maximum. Therefore our method produces slightly shorter summaries, which is reflected in the recall.

### 4.7 Discussion & Conclusion

Based on our experiments and observations, we now discuss some of the insights on the advantages and limitations of our work.

**Summarization Objective.** We are the first to learn the summarization objective from user summaries and jointly optimize for them. Given the complexity of the task, using hand-defined objectives as in [78, 79, 92] is often difficult to implement. Indeed, our experiments have shown the advantage of our approach.

What is a good summary cannot be defined absolutely. It depends on multiple factors, where the intention with which a video was taken is the most obvious (life-logging vs. professional videos). We have seen in our experiments that the summary length is also important. Short summaries focus on highlights,
Figure 4.5: **Egocentric dataset, Video P2:** Quantitative comparison of uniform sampling, [92] and our method. [92] is prone to include duplicates and misses the visit of the frozen yogurt shop. While uniform sampling includes this, it is hardly visible for the selected segments/frames: Uniform sampling more often selects frames that are non-informative. Instead, our method is diverse and selects segments that help understand the story.

<table>
<thead>
<tr>
<th>Method</th>
<th>F-measure</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>29.09 ± 1.65%</td>
<td>30.60 ± 1.87%</td>
</tr>
<tr>
<td>Uniform</td>
<td>26.77 ± 1.24%</td>
<td>28.92 ± 1.16%</td>
</tr>
<tr>
<td>Gygli et al.</td>
<td>39.34 ± 0.00%</td>
<td><strong>44.44 ± 0.00%</strong></td>
</tr>
<tr>
<td>Video MMR</td>
<td>26.58 ± 0.00%</td>
<td>26.37 ± 0.00%</td>
</tr>
<tr>
<td>Uniformity</td>
<td>24.69 ± 0.05%</td>
<td>27.11 ± 0.09%</td>
</tr>
<tr>
<td>Interestingness</td>
<td>39.52 ± 0.00%</td>
<td>42.50 ± 0.00%</td>
</tr>
<tr>
<td>Representative</td>
<td>26.69 ± 0.00%</td>
<td>26.65 ± 0.00%</td>
</tr>
<tr>
<td>Combined</td>
<td><strong>39.68 ± 0.09%</strong></td>
<td>43.01 ± 0.08%</td>
</tr>
</tbody>
</table>

Table 4.4: **User video dataset.** Performance of the individual objectives and previous methods vs. our approach.

longer summaries are expected to capture more diverse content. Furthermore, each person has different expectations when it comes to an optimal summary. Therefore an automatic summarization method should be able to adapt to user preferences. Our method could incorporate this by learning from summaries of a specific user.

**Datasets and ground truth collection.** For evaluation and training, human reference summaries are crucial. There is however still a shortage on the availability of large datasets for summarization. Several datasets have recently been introduced or annotated for video skimming [54, 135, 157, 191] or keyframe extraction [78, 79]. But regrettably, they are limited in size and many of them
are not publicly available [78, 79, 196]. Others are not annotated for the task of summarization, but only for highlight/relevance prediction [135, 157].

Collecting summarization ground truth is time consuming, especially for longer videos. Furthermore, multiple reference summaries are necessary per video, to account for the ambiguity of the task. An ideal solution would be to have pairs of raw and professionally edited videos. An approach in this direction is of Sun et al. [157] who mine YouTube videos for training. Thereby, they used the correspondence between the raw and edited version of a video. Even if the quality is not perfect, such an approach could potentially be used for evaluation and would provide a way to create large datasets without explicitly letting users annotate the videos.

**Interestingness.** Interestingness is context dependent [53, 167]. In sports games it might be a specific semantic event, such as a goal or a foul, while in the case of a static camera in your home, a summary should contain what is rare and unusual. In the general setting, it is harder to grasp what is interesting, even though first attempts have been made at least for images (cf. Chapter 2). For videos, this is still a largely unexplored problem. But even more than in text summarization, it is important to understand what is interesting, to avoid non-informative segments or junk and instead spot the highlights. Thus far, these problems are often circumvented by using additional information on the content in the video, such as the content category [78, 135, 157] or video titles [155]. Typically, this information is used to obtain a webprior on this topic [78, 79, 155].

**Conclusion.** We have proposed a new method for video summarization, where we formulated the problem as a subset selection problem. Using submodular maximization, a good approximate solution can be found. We have proposed adapted submodular functions and learnt a linear combination of them using structured learning with a large-margin formulation. Our experiments have shown the potential and generality of our method. In the future, it would be interesting to apply the method on more specific problems, such as sports games, where domain-knowledge can be incorporated.
Algorithm 1 Inference algorithm for submodular maximization with approximation bounds and lazy evaluations [93, 119].

1: function INFERENCE($\mathcal{V}, x_{\mathcal{V}}, c, w, f, B$)
2:     $y_{\text{uc}} \leftarrow \text{LAZYGREEDY}($$\mathcal{V}, x_{\mathcal{V}}, c, w, f, B,$$ uniform cost$)$
3:     $y_{\text{cb}} \leftarrow \text{LAZYGREEDY}($$\mathcal{V}, x_{\mathcal{V}}, c, w, f, B,$$ cost benefit$)$
4:     return arg max ($$w^T f(x_{\mathcal{V}}, y_{\text{uc}}), w^T f(x_{\mathcal{V}}, y_{\text{cb}})$$)
5: end function

6: function LAZYGREEDY($\mathcal{V}, x_{\mathcal{V}}, c, w, f, B, \text{type}$)
8:     $y \leftarrow \emptyset$ ▷ Start from an empty solution
9:     $\delta_{s} \leftarrow \infty, \forall s \in \mathcal{V}$ ▷ Initialize marginal gains
10: while $\exists s \in \mathcal{V} \setminus y : c(y \cup \{s\}) \leq B$ do ▷ Set gains to outdated
11:     $\text{cur}_{s} \leftarrow \text{false}, \forall s \in \mathcal{V} \setminus y$
12:     while true do
13:         if \text{type} = \text{uniform cost} then ▷ Max gain
14:             $s^* \in \arg \max_{s \in \mathcal{V} \setminus y, c(y \cup \{s\}) \leq B} \delta_{s}$
15:         elseif \text{type} = \text{cost benefit} then ▷ Max gain / cost
16:             $s^* \in \arg \max_{s \in \mathcal{V} \setminus y, c(y \cup \{s\}) \leq B} \frac{\delta_{s}}{c(s)}$
17:         end if
18:         if $\text{cur}_{s^*}$ then ▷ If gain of $s^*$ is up to date
19:             $y \leftarrow y \cup \{s^*\}$; ▷ Select the element
20:             break
21:         else ▷ Else, update marginal
22:             $\delta_{s} \leftarrow w^T f(x_{\mathcal{V}}, y \cup \{s^*\}) - w^T f(x_{\mathcal{V}}, y)$
23:         end if
24:     end while
25: end while
26: return $y$
27: end function
Query-Adaptive Video Summarization

5.1 Chapter overview

In the previous chapter we have shown how multiple objectives can be incorporated into a summarization model. We have further shown how the importance of these objectives can be learnt from data. However, the model requires annotation in form of several reference summaries and thus cannot be easily adapted to the task the summary is used for. As an example, summarization is often used in a search scenario, where a user can benefit from a summary that is specifically targeted towards the content he searches for. In the following we thus propose an adoption of our model to allow to generate video summaries that highlight content relevant to a search query. As before, we pose summarization as a subset selection problem. The key extension is a model for query relevance prediction. We quantify relevance by measuring the distance between frames and queries in a common textual-visual semantic embedding space induced by a neural network. In addition, we extend the model to capture query-independent properties, such as frame quality and composition. We incorporate these relevance scores into the submodular mixture model, which lets us optimize for summaries which are simultaneously diverse, representative, and relevant to a text query. We compare our method against previous state of the art on textual-visual embeddings for thumbnail selection and show that our model outperforms them on relevance prediction. Finally, to train and test our complete model for video summarization, we introduce a new dataset with diversity and query-specific relevance labels. On this dataset, our method outperforms several standard baselines such as Maximal Marginal Relevance.
Figure 5.1: Our query-adaptive video summarization model picks frames that are relevant to the query while also giving a sense of entire video. We want to summarize a video of an ironman competition, in which participants swim, bike and run. Query-adapted summaries are representative by showing all three sports, while placing more focus on the frames matching the query.

5.2 Introduction

As we have discussed in Chapter 4, Video recording devices have become omnipresent and users record videos with a *capture first, filter later* mentality. However, most raw videos never end up getting curated and remain too long, shaky, redundant and boring to watch. This raises new challenges in making these vast amounts of video data more accessible and organizing them. Apart from video summarization, automatic tagging [5, 115, 137] is a popular way to help organize video data. In automatic tagging, the goal is to predict metadata in form of tags, which allows to categorize videos and search them via text queries. In this Chapter we propose a model that combines the goals of summarization and text-based video search. Specifically, we propose a novel method that generates video summaries adapted to a text query (See Fig. 5.1). Our approach improves previous works in the area of textual-visual embeddings [80, 107] and proposes an extension of our model from Chapter 4 to allow to create the summaries that are query-adaptive.
5.3. Related Work

Our method for creating query-relevant summaries consists of two parts. We first develop a relevance model which allows us to rank frames of a video according to their relevance, given a text query. Relevance is computed as the cosine similarity between embeddings of frames and text queries in a learned visual-semantic embedding space, plus a query-independent term. While the embedding captures semantic similarity between video frames and text queries, the query-independent term predicts relevance based on generic frame properties such as the composition, blurriness and the interestingness of the content itself (cf. Chapter 2). We train this model on a large dataset of image search data [61] and our newly introduced Relevance and Diversity dataset (Sec 5.6.1). The second part of the summarization method is an adaptation of the submodular mixture model introduced in Chapter 4. This model is extended to optimize the selected set of frames for relevance, in addition to representativeness and diversity. We make the following contributions:

- Several improvements on learning a textual-visual embedding for thumbnail selection compared to previous work by Liu et al. [107]. These include better alignment of the learning objective to the task at test time and modeling the text queries by using LSTMs, leading to significant performance gains.

- A way to model semantic relevance and aesthetic properties of frames jointly, leading to better performance compared to using the similarity to text queries only.

- A method for diverse, query-adaptive video summaries based on submodular mixtures (Chapter 4) and our frame-based relevance model.

- A new video thumbnail dataset providing query relevance and diversity labels. As the judgments are subjective, we collect multiple annotations per video and analyze the consistency of the obtained labeling.

5.3 Related Work

The goal of video summarization is to select a subset of frames that gives a user an idea of the video’s content at a glance [167]. To find informative frames for this task, two dominant approaches exist: (i) modelling generic frame importance [58, 92] or (ii) using additional information such as the video title or a text query to find relevant frames [106, 107, 155]. In this
work we combine the two into one model and make several contributions for query-adaptive relevance prediction. Such models are related to automatic tagging [5, 115, 137], textual-visual embeddings [43, 107, 151] and image description [6, 26, 29, 35, 41, 72, 73, 113]. In the following we discuss approaches for video summarization, generic importance prediction models and previous works for obtaining embeddings which are the most relevant for this Chapter.

**Video summarization.** Video summarization methods can be broadly classified into extractive and abstractive approaches. The goal of extractive methods is to select an informative subset of keyframes [78, 79, 92, 185] or video segments [55, 110] from the initial video. Abstractive or compositional approaches instead transform the initial video into a more compact and appealing representation, e.g. hyperlapses [82], montages [158] or video synopses [136]. Our method is extractive. Extractive methods need to optimize at least two properties of the summary: the importance/interestingness of the selected frames and their diversity (Chapter 4), [49, 145]. Sometimes, additional objectives such as temporal uniformity (Chapter 4) and relevance [145] are also optimized. The simplest approach to obtain a representative and diverse summary is to cluster videos into events and select the best frame per event [31]. More sophisticated approaches jointly optimize for importance and diversity by using determinantal point process (DPPs) [49, 145, 195] or submodular mixtures [55, 102]. For this work, we follow Chapter 4 and formulate video summarization as a maximization problem over a set of submodular functions.

Most related to our approach is the work of Sharghi et al. [145], who present an approach for query-adaptive video summarization using DPPs. Their method is however limited to a small, fixed set of concepts such as car or flower. The authors leaves handling of more complex, unconstrained queries, as in our approach, for future work.

**Generic frame importance/interestingness.** Methods that predict the importance or interestingness of frames based on visual data only typically rely on supervised learning. While some approaches formulated it as a classification [135] or regression [92, 193] problem, most recent works treat it as a ranking problem [58, 157, 159, 190], which performs best, as shown in Chapter 3. To simplify the prediction problem, some approaches assume the domain of the video given and train a model for each domain [135, 157, 190]. Rather
than using a supervised model, Xiong et al. [186] detect “Snap points” by using a web image prior. Their generative model considers frames suitable as keyframes, if they match the composition of images, indifferent of their content. Our approach is partially inspired by this work in that it predicts relevance in the absence of a query, but relies on supervised learning.

Unconstrained Textual-visual models. Several methods exist that can retrieve images given unconstrained text or vice versa [35, 41, 43, 72, 73, 113]. These typically project both modalities into a joint embedding space [43], where semantic similarity can be compared using a measure like cosine similarity. To obtain the embedding of texts, word2vec [117], GloVe [131], etc. are popular choices for the vector representation of words. Then these methods learn a linear projection of deep image features into the same space so that they can be compared [43]. Liu et al. [107] applied this idea to video thumbnail selection. Our relevance model is based on [107], but we provide several important improvements. (i) Rather than keeping the word representation fixed, we jointly optimize the word and image projection. (ii) Instead of embedding each word separately, we train an LSTM model that combines a complete query into one single embedding vector, thus it even learns multi-word combinations such as Visit to lake and Star Wars movie. (iii) In contrast to [107], we directly optimize the target objective, learning to score a query-relevant image higher than any random image for a given query. In our experiments we show that these changes lead to significantly improved performance in predicting relevant thumbnails.

5.4 Method for Relevance Prediction

The goal of this work is to introduce a method to automatically select sets of video thumbnails that are both relevant, given a query, but also diverse enough to represent the video. To later optimize relevance and diversity jointly, we first need a way to evaluate the relevance of frames, given a query.

Our relevance model learns a projection of video frames \( v \) and text queries \( t \) into the same embedding space. We denote the projection of \( t \) and \( v \) as \( \mathbf{t} \) and \( \mathbf{v} \), respectively. Once trained, the relevance of a frame \( v \) given a query \( t \) can be estimated via some similarity measure. As [43], we use the cosine similarity

\[
    s(t, v) = \frac{\mathbf{t} \cdot \mathbf{v}}{\|\mathbf{t}\| \|\mathbf{v}\|}. \quad (5.1)
\]
Figure 5.2: A visualization of the latent semantic embedding space. Semantically similar concepts are projected to similar locations in the vector space.

While this allows to assess the semantic relevance of a frame w.r.t. a query, it is also possible to make a prediction on the suitability as thumbnails a priori, based on the frame quality, composition, etc. [186]. Thus, we propose to extend the above notion of relevance and model the quality aspects of thumbnails explicitly via

$$r(t, v) = s(t, v) + q_v,$$ (5.2)

where $q_v$ is a query-independent score determining the suitability of $v$ as a thumbnail, based on quality and general interestingness of a frame.

In the following, we investigate the how to formulate the task of obtaining the embeddings $t$ and $v$, as well as $q_v$.

### 5.4.1 Training objective

Intuitively, our model should be able to answer “What is the best thumbnail for this query?” Thus, the problem of picking the best thumbnail for a video is naturally formulated as a ranking problem. We desire that the embedding vectors of a query and frame that are a good match are more similar than ones
of the same query and a non-relevant frame\textsuperscript{1}. Thus, our model should learn to satisfy the following rank constraints:

\[ r(t, v^+) > r(t, v^-), \]  

(5.3)

where \( v^+ \) and \( v^- \) are relevant and irrelevant frame, given query \( t \). Alternatively, one can require the model to correctly rank frames when just using the similarity to the query \( t \) or query-independent information, \textit{i.e.} imposing the following constraints:

\[ s(t, v^+) > s(t, v^-) \]

\[ q_{v^+} > q_{v^-}. \]  

(5.4)

Experimentally, we find that training with these explicit constraints leads to slightly improved performance (See Tab. 5.1).

In order to impose these constraints and train the model, we define the loss as

\[
loss(t, v^+, v^-) = l_p \left( \max \left( 0, \gamma - s(t, v^+) + s(t, v^-) \right) \right) \\
+ l_p \left( \max \left( 0, \gamma - q_{v^+} + q_{v^-} \right) \right)
\]

(5.5)

where \( l_p \) is a cost function and \( \gamma \) is a margin parameter. We follow our work from Chapter 3 and use a Huber loss for \( l_p \), \textit{i.e.} the robust version of an \( l_2 \) loss. Next, we describe how to parameterize the \( t, v \) and \( q_v \), so that they can be learnt.

### 5.4.2 Text and Frame Representation

For predicting \( v \) and \( q_v \), we use a convolutional neural network, while \( t \) is obtained via a recurrent neural network. To jointly learn their parameters, we use a Siamese ranking network, trained with triplets of \( (t, v^+, v^-) \). We now describe the textual representation \( t \) and the image representations \( v \) and \( q_v \) in more detail.

**Textual representation.** As a feature representation of the textual query, we first project each word of the query into a 300-dimensional semantic space using the word2vec model \cite{word2vec} which is fine-tuned on unique queries from the

\textsuperscript{1}\cite{inverse} does the inverse. It poses the problem as learning to assign a higher similarity to corresponding frame and query than to the same frame and a random query. Thus, the model learns to answer the question “what is a good query for this image?”.
Bing Clickture dataset [61]. Then, we encode the individual word representations into a single embedding for the query using an LSTM [60]. This model will learn to attend to visually informative words more and learn about word combinations.

**Image representation.** To represent the image, we leverage the feature representations of a pre-trained VGG-19 network [149]. We replace the softmax layer with a linear layer $M$ with 301 dimensions. Thereby the first 300 dimensions are used as the embedding $t$, while the last dimension represents the query-independent score $q_v$.

### 5.5 Summarization model

In order to create a summary based on the relevance estimation obtained in the previous section, we propose to use a submodular mixtures framework [102], as in Chapter 4. As discussed, this framework poses summarization as the problem of selecting a subset (in our case, subset of frames) $y^*$ that maximizes a linear combination of submodular objective functions $f(x_V, y) = [f_1(x_V, y), ..., f_n(x_V, y)]^T$. Specifically,

$$y^* = \arg \max_{y \in \mathcal{Y}_V} w^T f(x_V, y),$$

where $\mathcal{Y}_V$ denote the set of all possible solutions $y$ and $x_V$ the features of video $V$. In this work, we assume that the cardinality $|y|$ is fixed to some value $k$. In For non-negative weights $w$, the objective in Eq. (5.6) is submodular [84]. Thus, it can be optimized near-optimally in an efficient way using a greedy algorithm with lazy evaluations [119, 125].

**Objective functions.** We choose a small set of objective functions, each capturing different aspects of the summary.

1. Query relevance $f(\cdot, \cdot) = \sum_{v \in y} s(t, v)$ where $t$ is the query embedding, $v$ is frame embedding and $s(\cdot, \cdot)$ denotes the cosine similarity defined in Eq. (5.1).
2. Quality score \( f(\cdot, \cdot) = \sum_{v \in \mathbf{y}} q_v \), where \( q_v \) represents score that is based on the quality of \( v \) as a thumbnail. This model scores the image relevance in a query-independent manner based on properties such as contrast, composition, etc.

3. The diversity of the elements in the summary
\[
f(x_{V}, y) = \sum_{i \in y} \min_{j<i} D_{x_{V}}(i, j),
\]
according to some dissimilarity measure \( D \). We use the Euclidean distance in of the FC2 features of the VGG-19 network for \( D^2 \).

4. Representativeness (Chapter 4). This objective favors selecting the medoid frames of a video, such that the visually frequent frames in the video are represented in the summary.

**Weight learning.** To learn the weights \( w \) in Eq. (1), ground truth summaries for query-video pairs are required. Previous works on query-adaptive methods typically only optimized for relevance [107] or used small datasets with limited vocabularies [145]. Thus, to be able to train our model, we collected a new dataset with relevance and diversity annotations, which we introduce in the next section.

If relevance and diversity labels are known, we can estimate the optimal mixing weights of the submodular functions through subgradient descent [102] or hyper-parameter search methods such as randomized parameter search [9].

In order to directly optimize for the F1-score using the submodular mixture framework [102], we use a locally modular approximation based on the procedure of [124] and optimize the weights using AdaGrad [37].

## 5.6 Relevance And Diversity Dataset (RAD)

We collected a dataset with query relevance and diversity annotation to allow to train and evaluate query-relevant summaries. Our dataset consists of 200 videos which are retrieved for a given query. Using Amazon Mechanical Turk (AMT) we annotated these videos with relevance labels regarding a given query as well an assignment of their frames into semantic clusters. These kind

\footnote{Derivation of submodularity of this objective is provided in the Appendix A.}
of labels were used previously in the MediaEval diverse social images challenge [65] and allowed to evaluate the performance of automatic methods in creating relevant and diverse summaries.

To select a representative sample of queries and videos for the dataset, we used the following procedure: We take the top YouTube queries between 2008 and 2016 from 22 different categories as seed queries. These queries are typically rather short and generic concepts. To obtain more complex and realistic queries we use YouTube auto-complete over the above queries. Using this approach we collect 200 queries. For each query, we take the top video result with a duration of 2 to 3 minutes.

In order to annotate the videos, we set up two consecutive tasks on Mechanical Turk. For this, we sample the videos at 1 frame per second. In the first task, a worker is asked to judge the relevance of each frame w.r.t. the given query. The following labels are possible: “Very Good”, “Good”, “Not good” and “Trash”, where trash indicates that the frame is both irrelevant and low-quality (e.g., blurred, bad contrast, etc.). After annotating the relevance to the query, the worker is asked to distribute the frames into clusters according to their visual similarity. We obtain one clustering per worker, where each clustering consists of mutually exclusive subsets of video frames as clusters. The number of clusters in the clustering is chosen by the worker. Each video is annotated by 5 different people and a total of 48 subjects participated in the annotation of the dataset. To ensure high-quality annotations, we defined a qualification task, where we check the results manually to ensure the workers provide good annotations. Only workers who passed this test were allowed to take further assignments.

5.6.1 Analysis

We now analyze the two kinds of annotations obtained through this procedure and describe how we merge these annotations into one set of ground truth labels per video.

Label distributions. The distribution of relevance labels is “Very Good”: 17.55%, “Good”: 57.40%, “Not good”: 12.31% and “Trash”: 12.72%. The
minimum, maximum and mean number of clusters per video are 4.9, 25.2 and 13.4 respectively over all videos of RAD.

**Relevance annotation consistency.** Given that there is some inherent subjectivity in this task and to check the quality of the annotations provided by the study subjects, we examine the consistency of the ratings. Thereby, we follow previous work \cite{53, 66, 182} and compute the Spearman’s rank correlation ($\rho$) between the relevance scores of different subjects, splitting five annotations of each video into two groups of two and three raters each as in \cite{182}. We take all split combination to find mean $\rho$ for a video. Our dataset has an average correlation of $\rho = 0.73$ over all videos, where 1 is a perfect correlation while 0 would indicate no consistency in the scores. \cite{182} reports a consistency of $\rho = 0.4$ for labels on event-specific image importance, which is a very related task. Thus, we can conclude that our relevance labels are of high quality and consistency.

**Cluster consistency.** To the best of our knowledge, we are the first to annotate multiple clusterings per video and look into the consistency of multiple annotators. MediaEval, for example, used multiple relevance labels but only one clustering \cite{65}. Various ways of measuring the consistency of clusterings exist, e.g. Variation of Information, Normalized Mutual Information or the Rand index (See Wagner and Wagner \cite{178} for an excellent overview). In the following we propose to use Normalized Mutual Information (NMI), an information theoretic measure \cite{42}:

$$NMI(C, C') = \frac{2 \ast I(C, C')}{H(C) + H(C')}$$

(5.7)

where $I(C, C')$ is the mutual information of clusterings $C, C'$ and $H(\cdot)$ is the entropy. We chose NMI over the more recently proposed Variation of Information (VI) \cite{116}, as NMI is more interpretable while closely related to VI (see Appendix B). NMI allows to compare consistency across different videos as it lies in $[0, 1]$. NMI is 0 if two clusterings are independent and 1, iff they are identical. As such it is the ideal measure as it allows for a simple and clear interpretation, comparable to Spearman’s rank correlation.

Our dataset has a cluster consistency of 0.54. Thus, the clusterings of different users are thus closer to being identical than being random, which we consider encouraging.
Ground truth For evaluation of test videos, we require a single ground truth annotation for each video. We merge the five annotations of relevance prediction and clustering of each query-video pair. For the final ground truth of relevance prediction, we require the labels as either Positive/Negative for each video frame. We map all VG labels to 1, G labels to 0.5 and Not good and Trash labels to 0. We compute the mean of five relevance annotation labels and label the frame as Positive if mean is $\geq 0.5$ and as Negative otherwise. In order to merge clustering annotation, we calculate NMI between all pairs of clustering and choose that clustering which has the highest mean NMI among the five clusterings, i.e. the most prototypical one. An example of relevance and clustering annotation is provided in Fig. 5.4.

5.7 Configuration testing

Before comparing our proposed relevance model against state-of-the-art, in Section 5.8, we first analyse the importance of different components of our model.

We analyse the performance of using different objectives, cost functions, text representation and training data. For evaluation we use the Query-dependent Video Thumbnail Selection Dataset provided by Liu et al. [107]. The dataset contains 20 candidate thumbnails for each video, each of which is labeled one of five scores: Very Good (VG), Good (G), Fair (F), Bad (B), or Very Bad (VB). We evaluate on 749 query-video pairs.

Objective. We compare our proposed training objective to that of [107]. Their method uses triplets of the form $(v, t^+, t^-)$ for training, i.e. a frame and a positive and a negative query. We also test the combination of the two objectives as e.g., in [72].

Cost function. Different loss functions $l_p$ (cf. Eq. (5.5)) are possible. We experiment with $l_1$ and Huber loss($l_{huber}$), which is a robust compromise between the $l_1$ and $l_2$ losses. More importantly, we also investigate the importance of modelling quality aspects. In particular, we compare different ways of modelling quality. (i) As defined in Eq.(5.3) ($Q_{expl}$), (ii) Combine it in the loss, as per Eq.(5.3) ($Q_{impl}$) or (iii) not modelling quality at all.
Text representation. As mentioned in Sec. 5.4.2, we represent the words of the query using a word vector. For aggregating the representations of multi-word queries into a single vector representation, we investigate two approaches: (i) averaging the word embedding vectors and (ii) using LSTM model that takes individual word embeddings as inputs and yield a query embedding.

Training dataset. For training, we use two datasets: (i) the Bing Clickture dataset [61] and (ii) the RAD dataset (Sec. 5.6.1).

The Clickture dataset, consists of queries and retrieved images from Bing Image search. The annotation is in form of triplets \((K, Q, C)\) meaning that the image \(K\) was clicked \(C\) times in the search results of the query \(Q\). This dataset is well suited for training our relevance embedding model, since our task is targeting the retrieving relevant keyframes from a video. We train on a subset of 0.5 million (query, clicked image, random image) triplets.

For the RAD dataset, we use the complete set of videos comprising of 200 of them. As mentioned in Sec. 5.6.1, we have labels: Positive/Negative for each frame based on the query. We leverage these labels to train our models for relevance prediction as in Tab. 5.1.

We use two datasets, to investigate the importance of the modality. While the Clickture dataset provides an abundance of data, it consists of images. The RAD dataset is smaller, but from the video domain. Thus, we can expect that its data is more suitable, in particular because frame annotation is expected to contain information on the quality of a frame, which is essential for \(q_v\) in Eq.(5.2).

Implementation details. We preprocess the images before passing them into the network. We truncate the number of words in the query at 14, as a tradeoff between the mean and maximum query length (5 and 26 respectively) [121]. We set the margin parameter \(\gamma\) in the loss in Eq. (5.5) to 1 and the tradeoff parameter \(\delta\) for the Huber loss to 1.5 as in Chapter 3. The LSTM consists of a hidden layer with 512 units. We train the parameters of the LSTM and projection layer \(M\) using stochastic gradient descent with adaptive weight updates (AdaGrad) [37]. We add an \(l_2\) penalty on the weights, with a \(\lambda\) of \(10^{-3}\). We train for 20 epochs using minibatches of 128 triplets.
### Table 5.1: Comparison of model configurations trained on a subset of the Clickture dataset or our Video Thumbnail dataset (RAD). As can be seen, having a better text representation and training objective substantially improves performance compared to the model proposed by Liu et al. [107]

<table>
<thead>
<tr>
<th>Method</th>
<th>Settings</th>
<th>Metrics</th>
<th>HIT@1</th>
<th>Spear ρ</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Cost</strong></td>
<td>LSTM</td>
<td>$q_v$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>57.17</td>
<td>-</td>
</tr>
<tr>
<td><strong>IMAGE DOMAIN (MSR)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours: Huber</td>
<td>$l_{huber}$</td>
<td>✓</td>
<td>✓</td>
<td>64.08</td>
<td>0.156</td>
</tr>
<tr>
<td>Two-way loss [72]</td>
<td>$l_1$</td>
<td>✓</td>
<td>✓</td>
<td>63.41</td>
<td>0.163</td>
</tr>
<tr>
<td>Ours: L1</td>
<td>$l_1$</td>
<td>✓</td>
<td>✓</td>
<td>66.62</td>
<td>0.173</td>
</tr>
<tr>
<td>Loss of Liu et al. [107]</td>
<td>$l_1$</td>
<td>✓</td>
<td>×</td>
<td>68.35</td>
<td>0.187</td>
</tr>
<tr>
<td>Loss of Liu et al. + LSTM</td>
<td>$l_1$</td>
<td>✓</td>
<td>✓</td>
<td>68.22</td>
<td>0.277</td>
</tr>
<tr>
<td>Ours: L1 + LSTM</td>
<td>$l_1$</td>
<td>✓</td>
<td>×</td>
<td>65.55</td>
<td>0.263</td>
</tr>
<tr>
<td>Ours: Huber + LSTM</td>
<td>$l_{huber}$</td>
<td>✓</td>
<td>×</td>
<td>67.15</td>
<td>0.273</td>
</tr>
<tr>
<td>Ours: Huber + LSTM + $Q_{impli}$</td>
<td>$l_{huber}$</td>
<td>✓</td>
<td>✓</td>
<td>69.02</td>
<td>0.270</td>
</tr>
<tr>
<td>Ours: Huber + LSTM + $Q_{expli}$</td>
<td>$l_{huber}$</td>
<td>✓</td>
<td>✓</td>
<td>67.42</td>
<td>0.319</td>
</tr>
</tbody>
</table>

| **VIDEO DOMAIN (RAD)** |         |       |       |         |      |
| Loss of Liu et al.      | $l_1$    | ✓    | ×    | 69.15   | 0.186 | 0.6313 |
| Loss of Liu et al. + LSTM| $l_1$    | ✓    | ✓    | 70.49   | 0.269 | 0.6502 |
| Ours: Huber + LSTM      | $l_{huber}$ | ✓    | ×    | 68.75   | 0.328 | 0.6586 |
| Ours: Huber + LSTM + $Q_{impli}$ | $l_{huber}$ | ✓    | ✓    | 70.76   | 0.340 | 0.6596 |
| Ours: $Q_{expli}$       | $l_{huber}$ | ✓    | ✓    | 70.62   | 0.319 | 0.6496 |
| Ours: Huber + LSTM + $Q_{expli}$ | $l_{huber}$ | ✓    | ✓    | 71.69   | 0.349 | 0.6611 |

**Results.** We show the results of our detailed experiments in Tab. 5.1. The evaluation metrics will be discussed in more depth in Sec 5.8. The results show several important points.

**Text representation.** Modelling queries with an LSTM, rather than averaging the individual word representations, improves performance on both datasets. This is not surprising, as this model can learn to ignore stop words and words that are not visually informative (e.g., 2014).

**Dataset.** As can be seen from the results, the dataset matters: All approaches tested on both datasets perform better when trained on RAD compared to Clickture. The only exception is the method of Liu et al. [107], which has to do with how the objective is formulated, as we discuss next.
**Objective and Cost function.** The analysis shows that training with our objective leads to better performance compared to using the objective of [107]. While the performance similar on image data, the former leads to increased performance when trained on video data. This can be explained with the different properties of videos vs. images returned by a search engine. The images are typically of good quality, while videos contain many frames that are low-quality and not visually informative [153]. Thus, formulating the thumbnail task in a way that the model can learn about this quality aspects is beneficial. Using the appropriate triplets for training boosts performance substantially (correlation with the loss of [107]: 0.269, ours: 0.328). When modelling the quality (Eq.(5.2)), performance improves further, where explicitly optimizing for quality and semantic similarity performs slightly better.

To conclude, we see that the better alignment of the objective to the keyframe retrieval task, the addition of an LSTM and modelling quality of the thumbnails improves performance. Together, they provide a substantial improvement compared to Liu et al.’s model, when both trained on RAD dataset. Our method achieves an absolute improvement of 2.98% in mAP, while the correlation improves from 0.186 to 0.349.

## 5.8 Experiments

Having determined the optimal configuration of our relevance model in Sec. 5.7, we now compare it against state-of-the-art models on the MSR Evaluation dataset and RAD. We also evaluate our summarization model on RAD. For this experiment, we split RAD into 100 videos for training, 50 for validation, and 50 for testing.

**Evaluation metric.** We evaluate relevance prediction methods by using the HIT@1, mean Average Precision (mAP) metrics as reported and defined in Liu et al. [107], as well as the Spearman’s Rank Correlation. HIT@1 is computed as the hit ratio for the highest ranked thumbnail.

To evaluate video summaries on RAD, we use F1 score. It is the harmonic mean of precision of relevance prediction and cluster recall [65]. It is high, if a method selects relevant frames from diverse clusters.
### Table 5.2: Performance of our relevance models on the RAD dataset in comparison with previous methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>HIT@1</th>
<th>Spear. $\rho$</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NO TEXTUAL INPUT</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Random</td>
<td>66.6 ±3.5</td>
<td>-</td>
<td>0.674</td>
</tr>
<tr>
<td>Video2GIF [58]</td>
<td>66.67</td>
<td><strong>0.167</strong></td>
<td>0.708</td>
</tr>
<tr>
<td><strong>TITLES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Liu et al. [107] +LSTM</td>
<td>70.0</td>
<td>0.134</td>
<td>0.7312</td>
</tr>
<tr>
<td>Ours: CNN-LSTM</td>
<td>70.0</td>
<td>0.182</td>
<td>0.7432</td>
</tr>
<tr>
<td>Ours: CNN-LSTM + $Q_{expli}$</td>
<td><strong>71.0</strong></td>
<td><strong>0.221</strong></td>
<td><strong>0.7599</strong></td>
</tr>
<tr>
<td><strong>QUERIES</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Liu et al. [107] +LSTM</td>
<td>72.0</td>
<td>0.204</td>
<td>0.7298</td>
</tr>
<tr>
<td>Ours: CNN-LSTM</td>
<td><strong>76.0</strong></td>
<td><strong>0.268</strong></td>
<td><strong>0.7516</strong></td>
</tr>
<tr>
<td>Ours: CNN-LSTM + $Q_{expli}$</td>
<td>72.0</td>
<td>0.264</td>
<td><strong>0.7688</strong></td>
</tr>
</tbody>
</table>

5.8.1 Evaluating the Relevance Model

In the following we evaluate the best configurations of our model, trained on 0.5M triplets of Clickture dataset, and compare it to [107] and Video2GIF (Chapter 3). We also finetune all the models on the training videos of RAD.

**MSR evaluation dataset** [107] The results on this dataset are shown in Tab. 5.3. We report the performance of Liu et al. [107] from their paper. Note, however, that the results are not directly comparable, as they use query-video pairs for predicting relevance, while only the titles are shared publicly. Thus, we use the titles instead. This difference is important, as the relevance is annotated with respect to the queries on MSR Evaluation dataset, which often differ from the titles. We quantify the value of these two different inputs in Tab. 5.2, where we separate the performances of query relevance and title relevance on the test set of the RAD dataset. As can be seen from this table, using queries leads to significantly better performance. Encouragingly, our model performs well even when just using the titles and no video search dataset as in Tab. 5.3. Our models improves mAP by 4.22% over [107] and correlation triples.
5.8. Experiments

Figure 5.3: Qualitative Results of top ranked keyframes on RAD. a) Liu et al. b) Video2GIF c) Our new model (from left). Video titles are shown on the left. Ground truth relevance labels are shown in Blue. P=Positive, N=Negative.

Our dataset We also evaluate our relevance model on our RAD testset. We observe that Ours: CNN-LSTM + Q\textsubscript{expl} performs relatively well compared to others in terms of mAP. The reduced performance in HIT@1 may be due to the effect of scores of the Q part of the model, which scores frame solely based on appearance. From Tab. 5.2, we observe that our model improves by 2.87% when using TITLES and 4.9% when using QUERIES in terms of mAP over our implementation of the model of Liu et al. [107]+LSTM. We present some qualitative results of different methods on relevance prediction for the videos from RAD testset in Fig. 5.3.

5.8.2 Evaluating the Summarization Model

As mentioned in Sec. 5.5, we use four objectives for our summarization model. Referring to Tab. 5.4, we use the QAR model to get similarity and quality scores while diversity and representativeness scores are obtained as described in Sec. 5.5. We compare the performance of our full model with each individual objective, a baseline based on Maximal Marginal Relevance (MMR) [18] and Hecate [153]. MMR greedily builds a set that maximises the weighted sum
Table 5.3: Thumbnail selection performance on the MSR Evaluation dataset. Note that [107] uses queries for their method which are not publicly available. Thus the numbers are not directly comparable.

<table>
<thead>
<tr>
<th>Method</th>
<th>HIT @ 1</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>QUERIES</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Liu et al. [107]</td>
<td>40.625</td>
<td>73.83</td>
<td>0.122</td>
<td>0.629</td>
</tr>
<tr>
<td>TITLES</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours: CNN-LSTM</td>
<td>36.71</td>
<td>72.63</td>
<td>0.367</td>
<td>0.6685</td>
</tr>
<tr>
<td>Ours: CNN-LSTM + Q_{impli}</td>
<td>38.08</td>
<td>72.09</td>
<td>0.382</td>
<td>0.6690</td>
</tr>
<tr>
<td>Ours: CNN-LSTM + Q_{expli}</td>
<td><strong>38.86</strong></td>
<td><strong>74.76</strong></td>
<td>0.376</td>
<td><strong>0.6712</strong></td>
</tr>
</tbody>
</table>

Results Quantitative results are shown in Tab. 5.4, while Fig. 5.4 shows qualitative results. As can be seen, combining all objectives with our model works best. It outperforms all single objectives, as well as the MMR [18] baseline, even though MMR also uses our well-performing similarity estimation. Similarity alone has the highest precision, but tends to pick frames that are visually similar (cf. Fig. 5.4), thus resulting in low cluster recall. Diversification objectives (diversity and representativeness) have a high cluster recall, but the frames are less relevant. Somewhat surprisingly, Hecate [153] is a relatively strong baseline. In particular, it performs well in terms of relevance, despite using a simple quality score. This further highlights the importance of quality for the thumbnail selection task. It also indicates that the used VGG-19 architecture might be suboptimal for predicting quality. CNNs for classification use small input resolutions, thus making it difficult to predict quality aspects such as blur. Finding better architectures for that task is actively researched, e.g., [109, 112], and might be used to improve our method.
Figure 5.4: Each row represents a video summary created by Video2GIF, Relevance objective and Our model (rows: 1, 6 and 11 in Tab. 5.4). Green number on images depicts frame number. We plot the distribution of relevance scores and cluster annotations over the video and marked the selected frames for the methods.

When analysing the learned weights (cf. Tab. 5.4) we find that the similarity prediction is the most important objective, which matches our expectations. Quality gets a lower, but non-zero weight, thus showing that it provides information that is complementary to query-similarity. Thus, it helps predicting the relevance of a frame. The reader should however be aware that differences in the variance of the objectives can affect the weights learned. Thus, they should be taken with a grain of salt and only be considered tendencies.

5.9 Conclusion

We introduced a new method for query-adaptive video summarization. At its core lies a textual-visual embedding, which allows to find frames relevant to a query. In contrast to earlier works, such as [145, 194], this model allows us to handle unconstrained queries and even full sentences. We proposed and empirically evaluated different improvements over [107], for learning a relevance
5. Query-Adaptive Video Summarization

<table>
<thead>
<tr>
<th>Method</th>
<th>(&lt; PR &gt;)</th>
<th>(&lt; CR &gt;)</th>
<th>(&lt; F1 &gt;)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similarity</td>
<td>Diversity</td>
<td>Quality</td>
<td>Repr</td>
</tr>
<tr>
<td>–</td>
<td>–</td>
<td>–</td>
<td>√</td>
</tr>
<tr>
<td>–</td>
<td>–</td>
<td>√</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>√</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>√</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>√</td>
<td>√</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>MMR [18]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>√ (33%)</td>
<td>√ (66%)</td>
<td>–</td>
</tr>
<tr>
<td>Hecate [153]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>√ (45%)</td>
<td>√ (43%)</td>
<td>√ (2%)</td>
</tr>
<tr>
<td>Upper bound</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.4: Performance of summarization methods on the RAD dataset. \(Repr\) means Representativeness. \(\checkmark\) and \(\_\) depict whether an objective was used or not. MMR and ours learn their corresponding weights. Percentage in parentheses the normalized learnt weights. Upper bound refers to the best possible performance, obtained using the ground truth annotations of RAD.

Our empirical evaluation showed that a better training objective, a more sophisticated text model, and explicitly modelling quality leads to significant performance gains. Finally, we introduced a new dataset for thumbnail selection which comes with query-relevance labels and a grouping of the frames according to visual and semantic similarity. On this data, we tested our full summarization framework and showed that it compares favorably to standard baselines such as Video-MMR [97]. We hope that our new dataset will spur further research in query adaptive video summarization.
6

Deep Value Networks for Iterative Structured Prediction

6.1 Chapter overview

In the two previous chapters we have relied on a Submodular Mixture Model, one of the two prominent ways to incorporate the multiple objectives of summarization into a computational model. While both, our approach and Sequential Determinantal Point Processes (DPPs) [49] work well in practice, Submodular Mixtures are linear in their learnt weights and DPPs can only model unary potentials and repulsion between elements. Thus, both models have limited modelling capabilities. In the following, we therefore propose a method that uses deep neural networks to directly learn a non-linear model for structured prediction. The key idea is to learn a novel Deep Value Network that quantifies the quality of different solution hypotheses, given an input such as an image. Once the value network is optimized, at inference, we find output structures that maximize the score of the value net via gradient descent on continuous relaxations of structured outputs. Thus, the DVN takes advantage of the joint modeling of the inputs and outputs. As this approach is still exploratory, we apply it on simple multi-label classification and image segmentation tasks, where it outperforms previous state of the art. In the future, our method can potentially be applied to summarization as well, which would render manually defining summarization objectives obsolete and allow to learn the complete summarization model directly from data. Apart from defining an appropriate objective, the key challenge thereby lies in extending DVNs to sequential models, so that they can be applied to videos with varying length.
6.2 Introduction

Structured output prediction is a fundamental problem in machine learning that entails learning a mapping from input objects to complex multivariate output structures. Because structured outputs live in a high-dimensional combinatorial space, one needs to design factored prediction models that are not only expressive, but also computationally tractable for both learning and inference. Due to computational considerations, a large body of previous work (e.g., [87, 169]) has focused on relatively weak linear graphical models with pairwise or small clique potentials. Such models are not capable of learning complex non-linear correlations between output variables, making them not suitable for tasks requiring complicated high level reasoning to resolve ambiguity.

By contrast, in this work we do not restrict the expressive power of the prediction model, and we simply rely on gradient descent as our inference algorithm. Our key intuition is that learning to critique different output hypotheses is easier than learning to directly come up with optimal predictions. Accordingly, we build a deep value network (DVN) that takes an input $x$ and a corresponding output structure $y$, both as inputs, and predicts a scalar score $v(x, y)$ evaluating the quality of the hypothesis $y$ and its correspondence with the input $x$. In teaching a DVN to evaluate different output hypotheses, a loss function $\ell(y, y^*)$ that compares an output structure $y$ against ground truth labels $y^*$, is the key learning signal. Our goal is to distill the loss function into the weights of a value network so that during inference, in the absence of a labeled output $y^*$, one can still rely on the value judgments of the neural network to rank output structures.

To enable effective iterative refinement of structured outputs via gradient descent on the score of a DVN, we relax our outputs to live in a continuous space instead of a discrete space, and extend the domain of loss function so the loss applies to continuous variable outputs. For example, for multi-label classification, instead of enforcing each output dimension $y_i$ to be binary, we let $y_i \in [0, 1]$ be continuous, and we generalize the notion of $F_1$ score to apply to continuous predictions. For image segmentation, we define a similar generalization of intersection over union. Then, we train a DVN on many output examples encouraging the network to predict very accurate (negative) loss scores for any output hypothesis. We generate the output hypotheses via gradient descent at training time, so that the value net’s estimate around the inference trajectory is as accurate as possible. We also generate output hy-
Gradient based inference

<table>
<thead>
<tr>
<th>Input $x$</th>
<th>Step 5</th>
<th>Step 10</th>
<th>Step 30</th>
<th>GT label $y^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 6.1: Segmentation results of DVN on Weizmann horses test samples. Our gradient based inference method iteratively refines segmentation masks to maximize the predicted scores of a deep value network. Starting from a black mask at step 0, the predictions converge within 30 steps yielding the output segmentation.

hypotheses by finding adversarial cases [51, 162] – output structures that have a large disagreement between the value network scores and the loss function.

Given a trained deep value network, we iteratively refine structured outputs by gradient ascent to maximize the score of the value network as depicted in Figure 6.1. All of the output variables are initialized at 0, but in about 20 to 30 gradient steps descent results are obtained.

We assess the effectiveness of the proposed deep value network, training on both multi-label classification based on text data and on image segmentation. In both cases we obtain state-of-the-art results even though the inputs are from different domains and the loss functions are different. Our algorithm implicitly learns a prior over output variables and takes advantage of the joint modeling of the inputs and outputs. Even given a small number of input-output pairs,
we find that we are able to build powerful deep value networks. For example, on the Weizmann horses dataset \cite{14}, without any form of pre-training, we are able to optimize 2.5 million network parameters only on 200 training images (with multiple crops). Our model is able to outperform methods that are pre-trained on large datasets such as ImageNet \cite{32} or methods that operate on larger input dimensions.

This chapter presents a new Deep Value Network architecture paired with a gradient descent inference algorithm for structured output prediction. We propose a novel training objective inspired by value-based reinforcement learning algorithms to accurately evaluate the quality of any input-output pair based on a notion of a loss function.

### 6.3 Background

Structured output prediction is a supervised learning problem that entails learning a mapping from some input objects $x \in \mathcal{X}$ (e.g., $\mathcal{X} \equiv \mathbb{R}^M$) to multivariate discrete outputs $y \in \mathcal{Y}$ (e.g., $\mathcal{Y} \equiv \{0, 1\}^N$). Given a training dataset of input-output pairs, $\mathcal{D} \equiv \{(x^{(i)}, y^{*^{(i)}})\}_{i=1}^N$, we aim to learn a mapping $\hat{y}(x) : \mathcal{X} \rightarrow \mathcal{Y}$ from inputs to ground truth outputs. Because finding the exact ground truth output structures in a high-dimensional space is often infeasible, one measures the quality of a mapping via a loss function $\ell(y, y') : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ that evaluates the distance between different output structures. Given such a loss function, the quality of a mapping is measured by empirical loss over a validation dataset $\mathcal{D}'$,

$$\sum_{(x,y^*) \in \mathcal{D}'} \ell(\hat{y}(x), y^*)$$

(6.1)

This loss can take an arbitrary form and is often non-differentiable. For multi-label classification, a common choice of the loss is $F_1$ scores and for image segmentation, a typical loss is intersection over union (IOU).

Some structured output prediction methods \cite{163, 169} learn a mapping from inputs to outputs via a score function $s(x, y; \theta)$, which evaluates different joint configurations of inputs and outputs based on a linear function of some joint input-output features $\psi(x, y)$,

$$s(x, y; \theta) = \theta^T \psi(x, y).$$

(6.2)
The goal of learning is to optimize a score function such that the model’s prediction $\hat{y}$,

$$\hat{y} = \arg\max_y s(x, y; \theta), \quad (6.3)$$

are closely aligned with ground-truth labels $y^*$ as measured by empirical loss in (6.1) on the training set.

Empirical loss is not amenable to numerical optimization because the argmax in (6.3) is discontinuous. Structural SVM formulations [163, 169] introduce a margin violation (slack) variable for each training pair, and define a continuous upper bound on the empirical loss. The upper bound on the loss for an example $(x, y^*)$ and the model’s prediction $\hat{y}$ takes the form:

$$\ell(\hat{y}, y^*) \leq \max_y \left[ \ell(y, y^*) + s(x, y; \theta) \right] - s(x, \hat{y}; \theta) \quad (6.4a)$$

$$\leq \max_y \left[ \ell(y, y^*) + s(x, y; \theta) \right] - s(x, y^*; \theta). \quad (6.4b)$$

Previous work [163, 169], defines a surrogate objective on the empirical loss, by summing over the bound in (6.4b) for different training examples, plus a regularizer. This surrogate objective is convex in $\theta$, which makes optimization convenient.

This work is inspired by the structural SVM formulation above, but we give up the convexity of the objective to obtain more expressive power using a multi-layer neural network. Specifically, we generalize the formulation above in three ways: 1) use a non-linear score function denoted $v(x, y; \theta)$ that fuses $\psi(\cdot, \cdot)$ and $\theta$ together and jointly learns them. 2) use gradient descent in $y$ for iterative refinement of outputs to approximately find the best $\hat{y}(x)$. 3) optimize the score function with a regression objective so that the predicted scores closely approximate the negative loss values,

$$\forall y \in \mathcal{Y}, \quad v(x, y; \theta) \approx -\ell(y, y^*). \quad (6.5)$$

Thus, we call our model a deep value network (DVN) because it is a non-linear function trying to evaluate the value of any $y \in \mathcal{Y}$ accurately. In the structural SVM’s objective, the score surface can vary as long as it does not violate margin constraints in (6.4b). By contrast, we restrict the score surface much more by penalizing it whenever it over- or underestimates the loss values. This seems to be beneficial as a neural network $v(x, y; \theta)$ has a lot of flexibility, and adding more suitable constraints can benefit learning.
Our value network and its inference strategy resembles the structured prediction energy network (SPEN) framework by Belanger & McCallum [7]. Importantly, our motivation and training objective are different. Belanger & McCallum rely on the structural SVM surrogate objective to train their neural models, whereas inspired by value based reinforcement learning, we learn an accurate estimates of the values as in (6.5). Empirically, we find that the DVN outperforms SPENs on multi-label classification using a similar neural network architecture.

6.4 Learning a Deep Value Network

We propose a deep value network architecture, denoted \( v(x, y; \theta) \), to evaluate a joint configuration of an input and a corresponding output via a neural network. More specifically, the deep value network takes as input both \( x \) and \( y \) jointly, and after several layers followed by non-linearity, predicts a scalar \( v(x, y; \theta) \), which evaluates the quality of an output \( y \) and its compatibility with \( x \). We assume that during training, one has access to an oracle value function \( v^*(y, y^*) = -\ell(y, y^*) \) that positively relates to the quality of the masks. Such an oracle value function assigns optimal values to any input-output pairs given ground truth labels \( y^* \). During training, the goal is to optimize the parameters of a value network, denoted \( \theta \), to mimic the behavior of the oracle value function \( v^*(y, y^*) \) as much as possible.

Example oracle value functions for image segmentation and multi-label classification include IOU and \( F_1 \) metrics, which are both defined on \( (y, y^*) \in \{0, 1\}^M \times \{0, 1\}^M \),

\[
\begin{align*}
    v^*_{\text{IOU}}(y, y^*) &= \frac{y \cap y^*}{y \cup y^*}, \\
    v^*_{F_1}(y, y^*) &= \frac{2 (y \cap y^*)}{(y \cap y^*) + (y \cup y^*)}.
\end{align*}
\]

(6.6) \hspace{1cm} (6.7)

Here \( y \cap y^* \) denotes the number of dimension \( i \) where both \( y_i \) and \( y_i^* \) are active and \( y \cup y^* \) denotes the number of dimensions where at least one of \( y_i \) and \( y_i^* \) is active. Assuming that one has learned a suitable value network that attains \( v(x, y; \theta) \approx v^*(y, y^*) \) at every input-output pairs, in order to infer a prediction for an input \( x \), which is valued highly by the value network, one needs to find \( \hat{y} = \arg\max_y v(x, y; \theta) \), which is described below.
6.4.1 Gradient based inference

Since $v(x, y; \theta)$ represents a complex non-linear function of $(x, y)$ induced by a neural network, finding $\hat{y}$ is not straightforward, and approximate inference algorithms based on graph-cut [15] or loopy belief propagation [122] are not easily applicable. Instead, we advocate using a simple gradient descent optimizer for inference. To facilitate that, we relax the structured output variables to live in a real-valued space. For example, instead of using $y \in \{0, 1\}^M$, we use $y \in [0, 1]^M$. The key to make this inference algorithm work is that during training we make sure that our value estimates are optimized along the inference trajectory.

Given a continuous variable $y$, to find a local optimum of $v(x, y; \theta)$ w.r.t. $y$, we start from an initial prediction $y^{(0)}$ (i.e. $y^{(0)} = [0]^M$ in all of our experiments), followed by gradient ascent for several steps,

$$y^{(t+1)} = P_Y (y^{(t)} + \eta \frac{d}{dy} v(x, y^{(t)}; \theta)),$$

where $P_Y$ denotes an operator that projects the predicted outputs back to the feasible set of solutions so that $y^{(t+1)}$ remains in $Y$. In the simplest case, where $Y = [0, 1]^M$, the $P_Y$ operator projects dimensions smaller than zero back to zero, and dimensions larger than one to one. After the final gradient step $T$, we simply round $y^{(T)}$ to become discrete. Empirically, we find that for a trained DVN, the generated $y^{(T)}$’s tend to become nearly binary themselves.

6.4.2 Optimization

To train a DVN using an oracle value function, first, one needs to extend the domain of $v^*(y, y^*)$ so it applies to continuous output $y$’s. For our IOU and $F_1$ scores, we simply extend the notions of intersection and union by using element-wise min and max operators,

$$y \cap y^* = \sum_{i=1}^M \min (y_i, y_i^*), \quad (6.9)$$

$$y \cup y^* = \sum_{i=1}^M \max (y_i, y_i^*). \quad (6.10)$$

Substituting (6.9) and (6.10) into (6.6) and (6.7) provides a generalization of IOU and $F_1$ score to $[0, 1]^M \times [0, 1]^M$. 
Our training objective aims at minimizing the discrepancy between \( v(x^{(i)}, y^{(i)}) \) and \( v^*(i) \) on a training set of triplets (input, output, value\(^*\)) denoted \( D \equiv \{(x^{(i)}, y^{(i)}, v^*(i))\}_{i=1}^N \). Very much like Q-learning \[184\], this training set evolves over time, and one can make use of an experience replay buffer. In Section 6.4.3, we discuss several strategies to generate training tuples and in our experiments we evaluate such strategies in terms of their empirical loss, once a gradient based optimizer is used to find \( \hat{y} \).

Given a dataset of training tuples, one can use an appropriate loss to regress \( v(x, y) \) to \( v^* \) values. More specifically, since both IOU and \( F_1 \) scores lie between 0 and 1, we used a cross-entropy loss between oracle values vs. our DVN values. As such, our neural network \( v(x, y) \) has a sigmoid non-linearity at the top to predict a number between 0 and 1, and the loss takes the form,

\[
L_{CE}(\theta) = \sum_{(x, y, v^*) \in D} -v^* \log v(x, y; \theta) - (1 - v^*) \log(1 - v(x, y; \theta))
\]  

The exact form of the loss does not have a significant impact on the performance and other loss functions can be used, e.g., \( L_2 \). A high level overview for training a DVN is shown in Algorithm 2. For simplicity, we show the case when not using a queue and batch size = 1.

### 6.4.3 Generating training tuples

Each training tuple comprises an input, an output, and a corresponding oracle value, \emph{i.e.} \((x, y, v^*)\). The way training tuples are generated significantly impacts the performance of our structured prediction algorithm. In particular, it is important that the tuples are chosen such that they provide a good coverage of the space of possible outputs and result in a large learning signal. There exist several ways to generate training tuples including:

- running gradient based \emph{inference} during training.
- generating \emph{adversarial tuples} that have a large discrepancy between \( v(x, y; \theta) \) and \( v^*(y, y^*) \).
- \emph{random samples} from \( \mathcal{Y} \), maybe biased towards \( y^* \).

We elaborate on these methods below, and present a comparison of their performance in Section 6.6.4. Our ablation experiments suggest that combining examples from gradient based inference with adversarial tuples works best.
Algorithm 2 Deep Value Network training

1: function TRAIN EPOCH (training buffer $D$, initial weights $\theta$, learning rate $\lambda$)  
2: \hspace{1em} while not converged do  
3: \hspace{2em} $(x, y^*) \sim D$ \hspace{1em} $\triangleright$ Get a training example  
4: \hspace{2em} $y \leftarrow$ GENERATEOUTPUT ($x, \theta$) \hspace{1em} $\triangleright$ cf. Sec. 6.4.3  
5: \hspace{2em} $v^* \leftarrow v^*(y, y^*)$ \hspace{1em} $\triangleright$ Get oracle value for $y$  
6: \hspace{2em} $\triangleright$ Compute loss based on estimation error cf. (6.11)  
7: \hspace{2em} $L \leftarrow -v^* \log v(x, y; \theta)$  
8: \hspace{3em} $-\left(1 - v^*\right) \log \left(1 - v(x, y; \theta)\right)$  
9: \hspace{2em} $\theta \leftarrow \theta - \lambda \frac{d}{d \theta} L$ \hspace{1em} $\triangleright$ Update DVN weights  
10: end while  
11: end function

Ground truth. In this setup we simply add the ground truth outputs $y^*$ into training with a $v^* = 1$ to provide some positive examples.

Inference. In this scenario, we generate samples by running a gradient based inference algorithm (Section 6.4.1) along our training. This procedure is useful because it helps learning a good value estimate on the output hypotheses that are generated along the inference trajectory at test time. To speed up training, we run a parallel inference job using slightly older neural network weights and accumulate the inferred examples in a queue.

Random samples. In this approach, we sample a solution $y$ proportional to its exponentiated oracle value, i.e. $y$ is sampled with probability $p(y) \propto \exp\{v^*(y, y^*)/\tau\}$, where $\tau > 0$ controls the concentration of samples in the vicinity of the ground truth. At $\tau = 0$ we recover the ground truth samples above. We follow [128] and sample from the exponentiated value distribution using stratified sampling, where we group $y$’s according to their values. This approach provides a good coverage of the space of possible solutions.

Adversarial tuples. We maximize the cross-entropy loss used to train the value network (6.11) to generate adversarial tuples again using a gradient based optimizer (e.g., see [51, 162]). Such adversarial tuples are the outputs $y$ for which the network over- or underestimates the oracle values the most. This strategy finds some difficult tuples that provide a useful learning signal, while ensuring that the value network has a minimum level of accuracy across all outputs $y$. 
6.5 Related work

There has been a surge of recent interest in using neural networks for structured prediction [25, 154, 197]. The Structured Prediction Energy Network (SPEN) of [7] inspired in part by [90] is closely related to the DVN architecture as SPENs also assign a non-linear score to each input-output configuration and use a gradient based inference algorithm to find a final solution. Importantly, the motivation and the objective function for SPENs and DVNs are distinct – SPENs rely on a max-margin surrogate objective whereas we directly regress the energy of an input-output pair to its corresponding loss. Unlike SPENs that only consider multi-label classification problems, we also train a deep convolutional network to successfully address complex and high-dimensional image segmentation problems.

Deep neural networks have been successfully optimized as autoregressive models of sequences [160] and images [28, 173]). These methods impose an order on the output variables, and enable efficient training of deep expressive models. However, in autoregressive models, the inference run-time grows linearly in the number of output dimensions, which is not acceptable for high-dimensional output structures. By contrast, inference under our method is efficient as all of the output dimensions are updated in parallel.

Our approach is inspired in part by the success of previous work on value-based reinforcement learning (RL) such as Q-learning [183, 184] (see [161] for an overview). The main idea is to learn an estimate of the future reward under the optimal behavior policy at any point in time. Recent RL algorithms use a neural network function approximator as the model to estimate the action values [174]. We adopt similar ideas for structured output prediction, where we use the task loss as the optimal value estimator. Unlike RL, we use a gradient based inference algorithm to find optimal solutions at test time.

Gradient based inference, sometimes called deep dreaming has led to impressive artwork and has been influential in designing DVN [38, 45, 120, 126]. Deep dreaming and style transfer methods iteratively refine the input to a neural net to optimize a prespecified objective. Such methods often use a pre-trained network to define a notion of a perceptual loss [70]. By contrast, we train a task specific value network to learn the characteristics of a task specific loss function and we learn the network’s weights from scratch.
Image segmentation [2, 19, 46, 59], is a key problem in computer vision and a canonical example of structured prediction. Many segmentation approaches based on Convolutional Neural Networks (CNN) have been proposed [24, 39, 46, 108, 127, 141]. Most use a deep neural network to make a per-pixel prediction, thereby modeling pairs of pixels as being conditionally independent given the input.

To diminish the conditional independence problem, recent techniques propose to model dependencies among output labels to refine an initial CNN-based coarse segmentation. Different ways to incorporate pairwise dependencies within a segmentation mask to obtain more expressive models are proposed in [23, 24, 86, 197]. Such methods perform joint inference of the segmentation mask dimensions via graph-cut [94], message passing [83] or loopy belief propagation [122], to name a few variants. Some methods incorporate higher order potentials in CRFs [81] or model global shape priors with Restricted Boltzmann Machines [40, 71, 98, 188]. Other methods do not rely on a graphical model, but instead learn to iteratively refine an initial prediction with CNNs, which may just be a coarse segmentation mask [95, 132, 143].

By contrast, this chapter presents a new framework for training a score function by keeping a gradient based inference algorithm in mind during training. Our deep value network framework applies to image segmentation and other structured prediction tasks. A key difference with some of the methods above is that we do not exploit complex combinatorial structures and special constraints such as sub-modularity to design specific inference algorithms. Rather, we use expressive energy models and the simplest conceivable inference algorithm of all – gradient descent.

### 6.6 Experimental evaluation

We evaluate the proposed Deep Value Networks on three tasks: On multi-class text analysis (Section 6.6.1), on binary image segmentation (Section 6.6.2) and on a three-class face segmentation task (Section 6.6.3). Section 6.6.4 investigates the sampling mechanisms and Section 6.6.5 visualizes the prior over the label distribution that the model learned.
6.6.1 Multi-label classification

We start by evaluating the method on the task of predicting multiple tags on text inputs. We use standard benchmarks in multi-label classification, namely Bibtex and Bookmarks, introduced in [75]. In this task, multiple labels are possible per example, and the correct number is not known. Given the structure in the label space, methods modeling label correlations often outperform models with independent label predictions. We compare our method to standard baselines including per-label logistic regression from [104], and a two-layer neural network with cross entropy loss [7], as well as SPENs [7] and PRLR [104], which is the state-of-the-art on these datasets. To allow direct comparison with SPENs, we adopt the same architecture in this work. Such an architecture combines local predictions that are non-linear in $x$, but linear in $y$, with a so-called global network, which scores label configuration with a non-linear function of $y$ independent of prediction and global networks have one or two hidden layers with Softplus non-linearities. We follow the same experimental protocol and report $F_1$ scores on the same test split as [7].

The results are summarized in Table 6.1. As can be seen from the table, our method outperforms the logistic regression baselines by a large margin. It also significantly improves over SPEN, despite not using any pre-training. SPEN, on the other hand, relies on pre-training of the feature network with a logistic loss to obtain good results. Our results even outperform the multi-label classification of [104]. This is encouraging, as their method is specific to classification and encourages sparse and low-rank predictions, whereas our technique does not have such dataset specific regularizers.

<table>
<thead>
<tr>
<th>Method</th>
<th>Bibtex</th>
<th>Bookmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic regression</td>
<td>37.2</td>
<td>30.7</td>
</tr>
<tr>
<td>NN baseline</td>
<td>38.9</td>
<td>33.8</td>
</tr>
<tr>
<td>SPEN [7]</td>
<td>42.2</td>
<td>34.4</td>
</tr>
<tr>
<td>PRLR [104]</td>
<td>44.2</td>
<td>34.9</td>
</tr>
<tr>
<td><strong>DVN (Ours)</strong></td>
<td><strong>44.7</strong></td>
<td><strong>37.1</strong></td>
</tr>
</tbody>
</table>

Table 6.1: Tag prediction from text data. $F_1$ performance of Deep Value Networks compared to the state-of-the-art on multi-label classification. All results except ours are taken from [7, 104].
6.6. EXPERIMENTAL EVALUATION

6.6.2 Weizmann horses

The Weizmann horse dataset [14] is a commonly used dataset for evaluating image segmentation algorithms [98, 143, 188]. The dataset consists of 328 images of left oriented horses and their binary segmentation masks. We follow [98, 143, 188] and evaluate the segmentation results at $32 \times 32$ dimensions. This dataset is well established in the literature for segmentation and proper segmentation of horses requires learning strong shape priors. The low resolution setting of $32 \times 32$ is challenging because parts such as the legs are often barely visible in the RGB image, thus relying on a learnt, complex shape model is important. We follow the experimentation protocol and report results on the same test split as [98].

For the deep value network we use a simple CNN architecture consisting of 3 convolutional and 2 fully connected layers (Figure 6.2). We use a learning rate of 0.01 and apply dropout on the first fully connected layer with the keeping probability 0.75 as determined on the validation set. We empirically found $\tau = 0.05$ to work best for stratified sampling. For training data augmentation purposes we randomly crop the image, similar to [85]. At test time, various strategies are possible to obtain a full resolution segmentation, which we investigate in Section 6.6.4. For comparison we also implemented a Fully Convolutional Network (FCN) baseline [108], by using the same convolutional layers as for the value network (cf. Figure 6.2). If not explicitly stated, masks are averaged over over 36 crops for our model and [108] (see below).
We test and compare our model on the Weizmann horses segmentation task in Table 6.2. We tune the hyper-parameters of the model on a validation set, and once best hyper-parameters are found, then fine tune on the combination of training and validation sets. We report the mean image IOU, as well as the IOU over the whole test set combined, as commonly done in the literature. It is clear that our approach outperforms previous methods by a significant margin on both metrics. Our model shows strong segmentation results, without relying on externally trained CNN features as (e.g., [143]). The weights of our value network are learned from scratch on crops of just 200 training images. Even though the number of examples is very small for this dataset, we did not observe overfitting during training, which we attribute to being able to generate a large set of segmentation masks for training.

We show qualitative results for CHOPPS [98], our implementation of fully convolutional networks (FCN) [108], and our DVN model in Figure 6.4. When comparing our model to FCN, trained on the same data and resolution, we find that the FCN has challenges correctly segmenting legs and ensuring that the segmentation masks have a single connected component (e.g., Figure 6.4, last two row). Indeed, the masks produced by the DVN correspond to much more reasonable horse shapes as opposed to those of other methods – the DVN seem capable of learning complex shape models and effectively grounding them to visual evidence. We also note that in our comparison in Table 6.2, most prior

<table>
<thead>
<tr>
<th>Input size</th>
<th>Method</th>
<th>Mean IOU %</th>
<th>Global IOU %</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 × 32</td>
<td>CHOPPS [98]</td>
<td>69.9</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Fully conv (FCN) baseline</td>
<td>78.56</td>
<td>78.7</td>
</tr>
<tr>
<td></td>
<td>DVN (Ours)</td>
<td>84.1</td>
<td>84.0</td>
</tr>
<tr>
<td>128 × 128</td>
<td>MMBM2 [188]</td>
<td>-</td>
<td>72.1</td>
</tr>
<tr>
<td></td>
<td>MMBM2 + GC [188]</td>
<td>-</td>
<td>75.8</td>
</tr>
<tr>
<td></td>
<td>Shape NN [143]</td>
<td>-</td>
<td>83.5</td>
</tr>
</tbody>
</table>

Table 6.2: Test IOU on Weizmann-32×32 dataset. Our method outperforms all previous methods, despite using a much lower input resolution than [188] and [143].
methods use larger inputs (e.g., 128x128), but we are nonetheless able to obtain the best results.

### 6.6.3 Labeled Faces in the Wild

The Labeled Faces in the Wild (LFW) introduced in [62] is a dataset designed for face recognition and contains more than 13000 images. A subset of 2927 was later annotated for segmentation by [71]. The labels are annotated on a superpixel level and consist of three classes: face, hair and background. We use this data in order to test the adaption of our approach to multiclass segmentation problems. We use the same train, validation and test splits as [71, 170]. As our method predicts labels for each pixel, we follow [170] and map these to superpixels by using the most frequent label in a superpixel as its class. To train the value network we predict mean pixel accuracy instead of superpixel accuracy for efficiency of computing the labels.

Table 6.3 shows quantitative results. Our method performs reasonably well, but is outperformed by state of the art methods on this dataset. We attribute this to three reasons. (i) the pre-training and more direct optimization of the per-pixel prediction methods of [108, 170], (ii) the input resolution and (iii) the properties of the dataset. In contrast to the horses, faces do not have thin parts and exhibit limited deformations. Thus, a feed forward method as used in [108], which produces coarser and smooth predictions is sufficient to obtain good results. Indeed, this has also been observed in the negligible improvement of refining CNN predictions with Conditional Random Fields and Restricted Boltzmann machines (cf. Table 6.3 last three rows). Despite this, our model is able to learn a prior on the shape and align it with the image evidence in most cases. Some failure cases include failing to recognize subtle and more rare parts such as mustaches, given their small size, and difficulties in correctly labeling blond hair. Figure 6.5 shows qualitative results of our segmentation method on this dataset.

### 6.6.4 Ablation experiments

In this section we analyze different configurations of our method. As already mentioned, generating appropriate training data for our method is key to learning good value networks. We compare 3 main approaches: 1) inference +
<table>
<thead>
<tr>
<th>Method</th>
<th>SP Accuracy %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully conv (FCN) baseline</td>
<td>95.36</td>
</tr>
<tr>
<td>DVN (Ours)</td>
<td>92.44</td>
</tr>
<tr>
<td>CRF (as in [71])</td>
<td>93.23</td>
</tr>
<tr>
<td>GLOC [71]</td>
<td>94.95</td>
</tr>
<tr>
<td>DNN [170]</td>
<td>96.54</td>
</tr>
<tr>
<td>DNN+CRF+SBM [170]</td>
<td>96.97</td>
</tr>
</tbody>
</table>

Table 6.3: Superpixel accuracy (SP Acc.) on Labeled Faces in the Wild test set.

Figure 6.3: Visualization of the learned shape priors (Weizmann horses). From left to right (a) The mean mask of the training set (b) mask generated when providing the mean image of the training set (c, d) Samples generated by our model by adding Gaussian random noise to the mean image with $\sigma = 10$.

ground truth, 2) inference + stratified sampling, and 3) inference + adversarial training. These experiments are conducted on the Weizmann dataset, described above. Table 6.4, top portion, reports IOU results for different approaches for training the dataset. As can be seen, including adversarial training works best, followed by stratified sampling. Both of these methods help explore the space of segmentation masks in the vicinity of ground truth masks better, as opposed to just including the ground truth masks. Adding adversarial examples works better than stratified sampling, as the adversarial examples are the masks on which the model is least accurate. Thus, these masks provide useful gradient information as to help improve the model.
### 6.6. Experimental Evaluation

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Mean IOU %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inference + Ground Truth</td>
<td>76.7</td>
</tr>
<tr>
<td>Inference + Stratified Sampling</td>
<td>80.8</td>
</tr>
<tr>
<td>Inference + Adversarial (DVN)</td>
<td>81.6</td>
</tr>
<tr>
<td>DVN + Mask averaging (9 crops)</td>
<td>81.3</td>
</tr>
<tr>
<td>DVN + Joint inference (9 crops)</td>
<td>81.6</td>
</tr>
<tr>
<td>DVN + Mask avg. non-binary (25 crops)</td>
<td>69.6</td>
</tr>
<tr>
<td>DVN + Joint inf. non-binary (25 crops)</td>
<td>80.3</td>
</tr>
<tr>
<td>DVN + Mask averaging (25 crops)</td>
<td>83.1</td>
</tr>
<tr>
<td>DVN + Joint inference (25 crops)</td>
<td>83.1</td>
</tr>
</tbody>
</table>

Table 6.4: Test performance of different configurations on the Weizmann 32x32 dataset.

We also investigate ways to do model averaging (Table 6.4, bottom portion). Averaging the segmentation masks of multiple crops leads to improved performance. When the masks are averaged naively, the result becomes blurry, making it difficult to obtain a final segmentation. Instead, joint inference updates the complete segmentation mask in each step, using the gradients of the individual crops. This procedure leads to clean, near-binary segmentation masks. This is manifested in the performance when using the raw foreground confidence (Table 6.4, Mask averaging non-binary vs. Joint inference non-binary). Joint inference leads to somewhat improved segmentation results, even after binarization, in particular when using fewer crops.

#### 6.6.5 Visualizing the learned correlations

To visualize what the model has learned, we run our inference algorithm on the mean image of the Weizmann dataset (training split). Optionally, we perturb the mean image by adding some Gaussian noise. The masks obtained through this procedure are shown in Figure 6.3. As one can see, the segmentation masks found by the value network on (noisy) mean images resemble a side-view of a horse with some uncertainty on the leg and head positions. These parts have the most amount of variation in the dataset. Even though noisy images do not contain horses, the value network hallucinates proper horse silhouettes, which is what our model is trained on.
6.7 Conclusion

This chapter presented a framework for structured output prediction by learning a deep value network that predicts the quality of different output hypotheses for a given input. As the DVN learns to predict a value based on both, input and output, it implicitly learns a prior over output variables and takes advantage of the joint modelling of the inputs and outputs. By visualizing the prior for image segmentation, we indeed find that our model learns realistic shape priors. Furthermore, rather than learning a model by optimizing a surrogate loss, using DVNs allows to directly train a network to accurately predict the desired performance metric (e.g., IOU), even if it is non-differentiable. We apply our method to several standard datasets in multi-label classification and image segmentation. Our experiments show that DVNs apply to different structured prediction problems achieving state-of-the-art results with no pre-training.

As future work, we plan to improve the scalability and computational efficiency of our algorithm by inducing input features computed solely on $x$, which is going to be computed only once. The gradient based inference can improve by injecting noise to the gradient estimate, similar to Hamiltonian Monte Carlo sampling. Finally, one can explore better ways to initialize the inference process.
Figure 6.4: Qualitative results on the Weizmann $32 \times 32$ dataset. In comparison to previous works, DVN is able to learn a strong shape prior and thus correctly detect the horse shapes including legs. Previous methods are often misled by other objects or low contrast, thus generating inferior masks.
Figure 6.5: Qualitative results on the Labeled Faces in the Wild (LFW) 3-class segmentation. The last two rows show failure cases, where our model does not detect some of hair and moustache correctly.
Conclusion

In this work we have investigated interest-based video summarization. We have made contributions in various areas of this complex topic. We now summarize these contributions and then discuss challenges that remain open.

7.1 Contributions

Truong and Venkatesh [167] have identified three broad perspectives on video summarization: Coverage, interesting events, and query context/personalization. We have made contributions to all of these, in particular on finding interesting parts of a video and personalization through text queries. More specifically, we have made the following contributions:

Chapter 2 analyzed what kind of images are considered interesting and proposed a computational model to predict it.

Chapter 3 built on the idea of visual interestingness and proposed a new model for predicting video highlights. By cleverly exploiting web data, we collected a new large-scale dataset with weakly annotated data. This, in turn, allowed us to train a deep neural network that predicts the suitability of video segments as GIFs. It is also the first publicly available highlight detection model. The collected dataset is the largest of its kind and one to two orders of magnitude larger than previously published datasets.

Chapter 4 proposed a summarization method based on structured prediction. We introduced a model that learns the weights of a linear combination of multiple submodular set functions, each modeling different desirable properties of a summary.
Chapter 5 introduced a model for query-adaptive video summarization, where the summary is adapted to a text query. The method relies on a textual-visual embedding space, where queries and frames can be compared using the cosine similarity to assess relevance. Through this embedding, our model is able to handle complex free-form text queries, while previous methods were limited to handling small vocabularies such as one or several single-word categories [145, 157]. In addition to predicting query relevance, we further showed how quality and relevance can be predicted in a joint way, leading to improved performance.

Chapter 6 proposed a novel approach for structured prediction. We learn a Deep Value Network that quantifies the quality of different solution hypotheses for a given input. Given a trained model, at inference, a solution is obtained by iteratively updating an initial solution in order to maximize the quality of the predicted output. In our experiments we applied the model to multi-label tag prediction and semantic segmentation, where it outperforms previous state of the art. Given the generality of the method, this model also has the potential to be applied to video summarization.

Finally, we have made several datasets and computational models publicly available, thus helping to improve reproducibility and making video summarization available outside the research community.

7.2 Perspectives

Video summarization is a complex and broad research area. It relies on or relates to visual recognition, clustering algorithms, subset selection, structured prediction, recommender systems and many more. In addition, it is an ambiguous task in the sense that there is not one true, but rather a subject specific answer. As such, several challenges in improving and evaluating summarization methods remain.

Understanding interest and relevance Summarization requires a high-level understanding of the videos. As such, its performance is heavily dependent on the quality of visual recognition and will continue to profit from improvements in areas such as image classification and action recognition in videos. For summarization itself, we consider two topics promising: The use of meta-data and personalization. Often, videos come with meta-data such as the video title or
7.2. Perspectives

As we have shown, taking the title into account when predicting relevance performs better than using the visual input alone (cf. Section 5.8). This has been exploited in several works by using the title or category of the video [107, 135, 155, 157, 159]. However, these approaches are typically limited to simple categories [135, 157, 159], or involve an expensive inference procedure [155]. Instead, we proposed a fast method that works with free-form titles and queries (Section 5). We expect to see continued progress in this area. Related to this, models that are able to adapt to a user profile are a promising research direction. Adapting to a specific user allows to reduce the ambiguity of the task and therefore make better predictions.

Output presentation Apart from finding the right content, we find that an aesthetic and coherent presentation of the summary is important, but this remain scarcely studied. Coherence means that the summary has a comprehensible ordering and progression of events. While most approaches simply choose a chronological ordering, this might be improved upon, or even impossible to use if multiple videos are summarized into one [27]. This topic has however not been studied, with the notable exceptions of [27, 110]. Aesthetic aspects relate to this: How can the summary be presented such that it is visually pleasing and engaging? Apart from the order of events, cinematographic rules are important [3]. These include: Following the 180° rule, shot scale diversity, short duration, avoiding jump cuts, etc. [114]. Except for the pioneering work [3], focusing on multi-video summarization, few approaches exist in this important area. A major challenge for these kind of methods, and for summarization in general, is how to objectively evaluate and compare models. We discuss this issue next.

Datasets and evaluation Datasets and evaluation are important research areas. This is especially true in video summarization, where it is difficult to objectively measure and compare the performance of different methods [167]. The definition of a perfect summary is person-, context- and task-specific.

Currently, two main evaluation regimes exist, each with its own advantages and disadvantages: (i) Letting subjects directly evaluate the quality of different summaries through user studies. This approach has the advantage that is exactly measures the target objective, namely user satisfaction. Unfortunately, it is time-consuming, expensive, and difficult to use in learning. (ii) Comparing
the automatic results against manually created summaries using some similarity metric. This method is fast, cheap and can additionally provide training data. The question there is, however, how closely the employed metric approximates user satisfaction. This method could potentially be improved by using data from user studies to learn a better evaluation metric [176]. In the industrial environment a better evaluation metric exists: Directly measuring user satisfaction in production systems. YouTube, for example, uses Computer Vision to automatically select good thumbnails when a new video is uploaded [189]. Alternatively, users can manually select one. Measuring the percentage of manually created versus automatic thumbnails, for various thumbnail extraction methods, allows for cheap, accurate evaluation. Other such metrics exist and could ideally also be applied in the academic setting. To date, the challenges of objectively evaluating summaries, and the cost of providing annotations for hours of video, have prevented the appearance of large and general datasets. We hope to see continued progress in this regard and eventually convergence towards standard datasets and evaluation metrics.

In summary, while several challenges remain, this thesis has shown ways to make sense of the flood of video content through automatic highlight detection and summarization. It has introduced methods based on submodular mixtures, which can be trained for specific summarization scenarios and can optionally adapted to a natural language query. Through contributing open datasets and source code — something which was previously limited — it has helped to advance the field and make video summarization accessible to a broader audience.
In the following we prove submodularity for the diversity objective as defined in Section 5.5.

**Theorem.** Let $A$ be an arbitrary ordered set. Let $j < i$ denote that $j$ appears earlier than $i$ in the set. Then, $f(A) = \sum_{i \in A} \min_{j < i} D(i, j)$ is submodular for any distance function $D(i, j)$.

**Proof.** Consider arbitrary sets $A \subseteq B$ and any $s \notin B$. We have:

\[
\begin{align*}
    f(B \cup \{s\}) - f(B) &= \sum_{i \in B \cup \{s\}} \min_{j < i} D(i, j) \\
    &\quad - \sum_{i \in B} \min_{j < i} D(i, j) \\
    &= \sum_{i \in B} \min_{j < i} D(i, j) + \min_{j \in B} D(s, j) \\
    &\quad - \sum_{i \in B} \min_{j < i} D(i, j) \\
    &= \min_{j \in B} D(s, j) \\
    &= \min \left( \min_{j \in A} D(s, j), \min_{j \in B \setminus A} D(s, j) \right) \\
    &\leq \min_{j \in A} D(s, j) \\
    &= f(A \cup \{s\}) - f(A).
\end{align*}
\]

We also note that this objective is monotone for non-negative distance functions $D$. 
Relation of Normalized Mutual Information and Variation of Information

In our RAD dataset, we have two kinds of annotations: Relevance prediction and Clustering. For relevance, we use Spearman’s rank correlation to analyze the consistency in the annotations. For cluster consistency, many measures exist in the literature, such as Variation of Information (VI) [116], Normalized Mutual Information (NMI), the Rand index, etc. We use NMI due to its advantages in terms of interpretability, as mentioned in Sec. 5.6.1. Let us now analyze the relation between NMI and VI.

Variation of Information (VI) is a recently proposed approach for measuring consistency between clusterings. Instead, we use NMI which is closely related to VI. VI is defined as

\[ VI(C, C') = H(C) + H(C') - 2 \times I(C, C') \]  

(B.1)

while NMI is defined as

\[ NMI(C, C') = \frac{2 \times I(C, C')}{H(C) + H(C')} \]  

(B.2)

It is easy to show that the NMI of two clusterings is linearly related to their VI normalized by the VI of random clusterings. For two independent clus-
terings, the mutual information $I(\cdot, \cdot)$ is 0, thus the Normalized Variation of Information (NVI) is:

$$NVI(C, C') = \frac{H(C) + H(C') - 2 \times I(C, C')}{H(C) + H(C')}$$

$$= 1 - \frac{2 \times I(C, C')}{H(C) + H(C')}$$

$$= 1 - NMI(C, C'). \quad (B.3)$$

Hence, there is a simple relation between NVI and NMI. Despite this simple relation, VI is more difficult to interpret and compare across multiple datasets (videos in this case) than NMI [178]. We therefore use NMI, which lies in $[0, 1]$ and provide a simple and clear interpretation for clustering consistency as presented in Section 5.6.1.
Bibliography


[95] K. Li, B. Hariharan, and J. Malik. Iterative instance segmentation. *CVPR*, 2016. 95
[118] T. Mikolov and J. Dean. Distributed representations of words and phrases and their compositionality. NIPS, 2013. 71


[153] Y. Song, M. Redi, J. Vallmitjana, and A. Jaimes. To click or not to click: Automatic selection of beautiful thumbnails from videos. In CIKM, 2016. 79, 81, 82, 84


[192] C. Zauner. Implementation and benchmarking of perceptual image hash functions. 2010. 31


List of Publications

Referred Conference Proceedings


(* denotes equal contribution)

Others


  (* denotes equal contribution)
