



A Wasserstein Subsequence Kernel for Time Series

Conference Paper**Author(s):**

[Bock, Christian](#) ; Togninalli, Matteo; Ghisu, Elisabetta; Gumbsch, Thomas; [Rieck, Bastian Alexander](#) ; Borgwardt, Karsten

Publication date:

2019

Permanent link:

<https://doi.org/10.3929/ethz-b-000390132>

Rights / license:

[In Copyright - Non-Commercial Use Permitted](#)

Originally published in:

<https://doi.org/10.1109/ICDM.2019.00108>

Funding acknowledgement:

634541 - A Clinical Decision Support system based on Quantitative multimodal brain MRI for personalized treatment in neurological and psychiatric disorders (SBFI)

A Wasserstein Subsequence Kernel for Time Series

Christian Bock[†], Matteo Togninalli[†], Elisabetta Ghisu, Thomas Gumbsch, Bastian Rieck, Karsten Borgwardt

Machine Learning and Computational Biology Lab, D-BSSE, ETH Zurich, Switzerland

SIB Swiss Institute of Bionformatics, Switzerland

firstname.lastname@bsse.ethz.ch

[†]These authors contributed equally

Abstract—Kernel methods are a powerful approach for learning on structured data. However, as we show in this paper, simple but common instances of the popular \mathcal{R} -convolution kernel framework can be meaningless when assessing the similarity of two time series through their subsequences. We therefore propose a meaningful approach based on optimal transport theory that simultaneously captures local and global characteristics of time series. Moreover, we demonstrate that our method achieves competitive classification accuracy in comparison to state-of-the-art methods across a wide variety of data sets.

Index Terms—Time Series Classification, Optimal Transport, Wasserstein, \mathcal{R} -Convolution Kernels

I. INTRODUCTION

Time series are ubiquitous in numerous domains, ranging from astrophysics [1] to biomedical applications [2], with time series classification (TSC) remaining a highly active research topic. TSC methods make use of extremely diverse methodologies, relying on the extraction of short, predictive subsequences [3], for example or on distance measures, such as dynamic time warping (DTW). We approach this topic from a different point of view, taking inspiration from the field of kernel methods. While some attempts were made to develop relevant kernel functions for TSC, successes in this area are limited. We hypothesise that this might be due to the fact that kernel function construction for time series suffers from two fundamental pitfalls. First, many similarity measures are either hypersensitive or insensitive to time shifts. Second, some time series subsequence kernel functions, representing simple instances of the \mathcal{R} -convolution framework [4], can be meaningless, as we will show in Section II.

In this paper, we introduce the Wasserstein Time Series Kernel (WTK), a kernel for time series that captures the similarity between subsequence distributions in addition to their pairwise similarities. WTK relies on notions from optimal transport, a field increasingly popular in the data mining community that provides similarity measures between probability distributions [5]. In the remainder of the paper, we describe the following contributions in detail:

- 1) We show that a straightforward application of simple instances of the \mathcal{R} -convolution kernel framework to time series can be meaningless.
- 2) We develop the first subsequence-based distance measure for time series that relies on the Wasserstein distance.
- 3) We demonstrate its competitive classification performance in comparison to state-of-the-art methods.

II. BACKGROUND

We give a more formal definition of kernels and discuss recent advances in the field. Let \mathcal{X} be a set with n elements and $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a function that is symmetric and positive definite¹, i.e. $\sum_{i,j=1}^n c_i c_j k(x_i, x_j) \geq 0$ for every $c_i \in \mathbb{R}$ and $x_i, x_j \in \mathcal{X}$. Then there exists a *Hilbert space* \mathcal{H} , i.e. a complete inner product space, and a mapping $\phi: \mathcal{X} \rightarrow \mathcal{H}$ such that $k(\cdot, \cdot)$ can be equivalently expressed as $k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}$, where $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denotes the inner product on \mathcal{H} . The space \mathcal{H} is also referred to as a *Reproducing Kernel Hilbert Space* (RKHS) because its inner product *reproduces* k . Since proving that positive definiteness holds for a kernel function can be challenging [6], alternative approaches that are not based on an RKHS were developed [7]. They are known as *Reproducing Kernel Kreĭn Spaces* (RKKS). In an RKKS, the positive definiteness requirement for the kernel function is dropped, so that kernels are allowed to be indefinite. Previous research [8] also showed that support vector machine (SVM) classifiers can use indefinite kernel matrices while maintaining favourable predictive performance.

However, kernel construction can suffer from certain pitfalls, particularly when using simple instances of the \mathcal{R} -convolution kernel framework [4], which evaluates a base kernel between all substructures and aggregates them. Letting T, T' refer to two time series, $\mathcal{S}, \mathcal{S}'$ to their respective sets of subsequences, and k_{base} be a base kernel function, such a kernel takes the form of

$$k(T, T') := \frac{1}{|T| \cdot |T'|} \sum_{s \in \mathcal{S}} \sum_{s' \in \mathcal{S}'} k_{\text{base}}(s, s'). \quad (1)$$

Choosing k_{base} as a linear kernel will lead to

$$\begin{aligned} k(T, T') &= \frac{1}{|T| \cdot |T'|} \sum_{s \in \mathcal{S}} \sum_{s' \in \mathcal{S}'} s^\top s' \\ &\approx \frac{1}{|T| \cdot |T'|} \left(\sum_{s \in \mathcal{S}} s^\top \right) \left(\sum_{s' \in \mathcal{S}'} s' \right) \approx \bar{T}^\top \bar{T}', \end{aligned} \quad (2)$$

where the last approximation follows from the fact that in the respective sums over all subsequence feature vectors, almost all of the observations of length w , except for the *leading* $w - 1$ as well as the *trailing* $w - 1$ observations, will appear at all dimensions of the vectors in the sum. Hence, for many values of w , this \mathcal{R} -convolution kernel degenerates to a simple

¹For reasons of notational simplicity, we will not make a distinction between ‘positive definite’ and ‘positive semi-definite’ in this paper.

comparison of the means of two time series T and T' . The consequence of Eq. 2 is that, in particular for z -normalised data sets, which are the suggested default in time series analysis [9], the kernel becomes de facto *meaningless*. Our argumentation runs along the lines of the famous result [10] about the inherent meaninglessness of clustering time series subsequences. When looking at some data sets from the ‘UCR Time Series Classification Archive’ [11], we observe that the values obtained from a \mathcal{R} -convolution linear kernel are close to zero for short subsequence lengths. As our experiments in Section V demonstrate, even increasing the subsequence length w does not result in competitive predictive performance.

III. RELATED WORK

There is a plethora of TSC approaches, so we refer the reader to Bagnall et al. [12] for a comprehensive overview of methods. The first kernel-based classification approaches comprise standard SVM kernels (linear, RBF) between whole time series [13]. For periodic patterns, several cross-correlation kernels are available [14]. Furthermore, some methods are based on DTW kernels [15] or general alignments of time series [16]. A lack of positive definiteness in such kernels prompted an investigation into the impact of indefinite kernels on classification performance [17]. Closest to our current method is KEMD [18], which uses the Earth Mover’s Distance [19] on histograms of the time series.

IV. OUR METHOD

A. Optimal transport

Optimal transport refers to a set of methods for comparing probability distributions by geometrical means. One of its commonly used methods is the *Wasserstein distance*. Given the probability distributions on some metric space, the Wasserstein distance defines a metric between them. More precisely, let σ and μ be two probability distributions defined on a metric space \mathcal{M} with some metric $\text{dist}(\cdot, \cdot)$.

Definition 1. Given $p \in \mathbb{R}_{>0}$, the p^{th} Wasserstein distance is defined as

$$W_p(\sigma, \mu) := \left(\inf_{\gamma \in \Gamma(\sigma, \mu)} \int_{\mathcal{M} \times \mathcal{M}} \text{dist}(x, y)^p d\gamma(x, y) \right)^{\frac{1}{p}}, \quad (3)$$

where $\Gamma(\sigma, \mu)$ is the set of all transportation plans $\gamma \in \Gamma(\sigma, \mu)$ over $\mathcal{M} \times \mathcal{M}$ with marginals σ and μ on the first and second factors, respectively.

The Wasserstein distance satisfies the axioms of a metric, as long as $\text{dist}(\cdot, \cdot)$ is a metric [20]. This is also used for other classification problems [21]. We will focus on the 1st Wasserstein distance, i.e. $p = 1$, and refer to it as *the* Wasserstein distance. The Wasserstein distance is related to optimal transport problems [20], where the general goal is to find the most ‘inexpensive’ (in terms of predefined costs) way to transport the probability mass of one probability distribution σ to another probability distribution μ . It is possible to reformulate the Wasserstein distance as an optimisation problem between two matrices [19].

Definition 2. Let $X \in \mathbb{R}^{n \times m}$ and $Y \in \mathbb{R}^{n' \times m}$ be two matrices. We consider X and Y to represent sets of feature vectors of dimension m , but with potentially different cardinalities n and n' . The 1st Wasserstein distance between X and Y is defined as

$$W_1(X, Y) := \min_{P \in \Gamma(X, Y)} \langle D, P \rangle_{\text{F}}, \quad (4)$$

where D is an $n \times n'$ matrix containing the pairwise distances $\text{dist}(x, y)$ for $(x, y) \in X \times Y$, P is the transport matrix, and $\langle \cdot, \cdot \rangle_{\text{F}}$ is the Frobenius inner product.

The transport matrix P contains the fractions indicating how to transport the values from X to Y with the lowest transport effort. If we assume that the total mass to be transported is 1 and is evenly distributed across the elements of X and Y , then the values for the rows and columns of P must sum up to $1/n$ and $1/n'$, respectively.

B. A subsequence-based Wasserstein kernel

We now define our novel subsequence-based Wasserstein kernel. Let $w \in \mathbb{N}_{>0}$ refer to a window width (i.e. a subsequence length). Given a set of n time series $\mathcal{T} := \{T_1, \dots, T_n\}$ we denote their length- w subsequences as $\mathcal{S} := \{\mathcal{S}_1, \dots, \mathcal{S}_n\}$.

Definition 3 (Wasserstein time series kernel). Let T_i and T_j be two time series, and s_{i1}, \dots, s_{iU} as well as s_{j1}, \dots, s_{jV} be their respective subsequences. Moreover, let D be a $U \times V$ matrix that contains the pairwise distances of all subsequences, such that $D_{uv} := \text{dist}(s_{iu}, s_{jv})$, where $\text{dist}(\cdot, \cdot)$ denotes the usual Euclidean distance. Following Definition 2, we must solve the optimisation problem

$$W_1(T_i, T_j) := \min_{P \in \Gamma(T_i, T_j)} \langle D, P \rangle_{\text{F}}, \quad (5)$$

which yields the optimal transport cost to transform T_i into T_j by means of their subsequences. Then, given $\lambda \in \mathbb{R}_{>0}$, we can define

$$\text{WTK}(T_i, T_j) := \exp(-\lambda W_1(T_i, T_j)), \quad (6)$$

which we refer to as our Wasserstein-based subsequence kernel; we will discuss its theoretical properties in Section IV-D.

Since we consider a time series T_i to be represented by its set of subsequences \mathcal{S}_i , we will also write $W_1(\mathcal{S}_i, \mathcal{S}_j) := W_1(T_i, T_j)$ and $\text{WTK}(\mathcal{S}_i, \mathcal{S}_j) := \text{WTK}(T_i, T_j)$ to simplify the notation. Note that in the calculation of WTK, the expression in Eq. 5, i.e. $W_1(T_i, T_j)$, is a metric [20]. Hence, it is also possible to use it in a k -nearest neighbour (k -NN) classifier.

C. Intuition

Similar to the Matrix Profile [22] and other shapelet-based methods, our method WTK makes use of the descriptive power of the subsequences of a time series. Figure 1 depicts the individual steps of our method, i.e. length- w subsequence

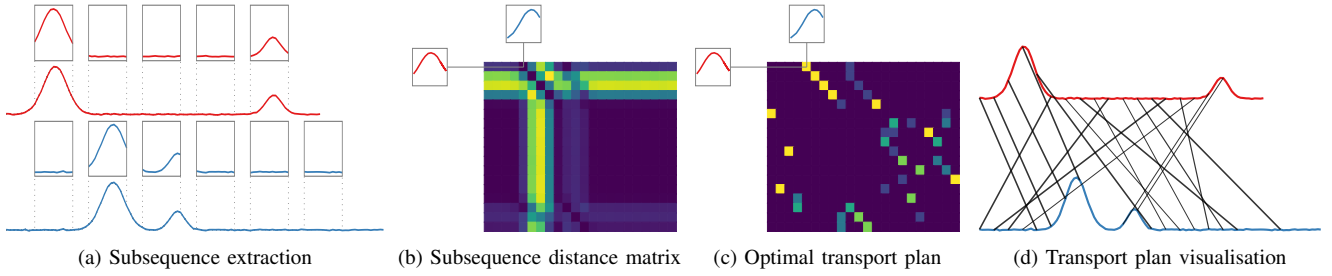


Figure 1. To measure the dissimilarity between two time series, our method proceeds in several steps. (a) First, all subsequences of the two time series are extracted based on a sliding window approach (here, not all subsequences are shown because their windows overlap). (b) Second, the pairwise distance matrix between all subsequences is calculated. Yellow indicates large distances, whereas blue indicates small distances. On its own, this matrix is not sufficient to assess the dissimilarity between the two time series, because it is not clear which subsequences correspond to each other. (c) Calculating the optimal transport plan makes correspondences more readily visible. Yellow indicates a large fraction of transported mass (high similarity), whereas blue indicates a small fraction (low similarity). For example, the two highlighted subsequences are matched with each other in the plan. Since the two time series have different lengths, some rows of the transport plan contain fractional matchings, making it possible to detect fine-grained differences in the distributions of subsequences. (d) A visualisation of the transport plan. Each line indicates a match between two subsequences, with the line being anchored at the respective beginning of the corresponding subsequences. The thickness of the line indicates the transport value. For clarity, only the largest transport values are shown.

extraction (Figure 1a), followed by pairwise distance calculations (Figure 1b), which is finally followed by the calculation of the optimal transport plan (Figure 1c).

The transport plan P that results from solving the optimisation problem in Eq. 5 can be seen as a map that assigns each length- w subsequence of the first time series (columns) to *at least* one length- w subsequence of the other time series (rows). Ultimately, all subsequences from the first time series must be ‘transported’ to some subsequences of the second time series. Figure 1d shows how the obtained transport plan values map on the example time series. The formulation of the optimisation problem already accounts for different cardinalities in the respective sets, which makes our method applicable to time series of varying lengths, as depicted in the figure. In case the time series have the same length (such as the time series data sets in the ‘UCR Time Series Archive’), this mapping is a bijection.

Finally, the Wasserstein distance is computed by summing the entries obtained by the Hadamard product of the two matrices shown in Figure 1. The obtained distance value is capable of better capturing the difference between the time series in terms of subsequence distributions compared with merely summing the values of the transport plan. In the example, we can observe that the optimisation procedure selects the lowest distances between subsequences and aligns the respective peaks of the time series correctly.

D. Theoretical Properties

Before using the similarity measure defined in Eq. 6, we have to prove that it satisfies the properties of a kernel: if we want to have a kernel that belongs to an RKHS, we must prove that it is positive definite. According to Feragen et al. [23, Theorem 5], this is equivalent to stating that for any data set, the symmetric matrix \mathcal{D} with $\mathcal{D}_{i,j} = W_1(T_i, T_j)$ is (conditionally) negative definite, which implies that it has at most *one* positive eigenvalue. Our empirical results indicate that for some data sets and some configurations, we observe *more* than one positive eigenvalue in \mathcal{D} ; the kernel matrix \mathcal{K}

obtained via Eq. 6 is therefore not positive definite. This leaves us with several options:

- (a) We can *enforce* the eigenvalue condition by calculating $\mathcal{L} := \mathcal{K} \cdot \mathcal{K}^\top$, where \mathcal{K} refers to the $n \times n$ matrix with entries according to Eq. 6. Letting $y := \mathcal{K}^\top x$ for $x \in \mathbb{R}^n$, we then have $x^\top \mathcal{K} \mathcal{K}^\top x = x^\top \mathcal{K} y = y^\top y = \sum_{i=1}^n y_i^2 \geq 0$, so \mathcal{L} is positive definite. This is also known as the *empirical kernel*. It is computationally the easiest, requiring only an additional matrix multiplication. However, it changes the values between individual time series, and we observed in our experiments that the predictive performance suffers when compared with other options.
- (b) We can *simplify* the matrix by subtracting all negative eigenvalues, leading to $\mathcal{L} := \mathcal{K} - \sum_i \lambda_i v_i v_i^\top$, where i ranges over the indices of the negative eigenvalues and v_i denotes their corresponding unit eigenvectors. By construction, this will set negative eigenvalues to zero, leaving us with a positive definite matrix. This is computationally harder, requiring a full eigendecomposition.
- (c) We can *generalise* the Wasserstein distance to a ‘softmin’ of all possible transportation plans, which ensures that we obtain a positive definite kernel. However, it scales exponentially with the number of subsequences and is infeasible for all practical purposes [24].
- (d) We can *sidestep* the eigenvalue condition by using algorithms that are capable of handling these *indefinite* matrices [7].

The majority of all data sets in our experiments resulted in a positive definite kernel matrix \mathcal{K} , making the options outlined above unnecessary. Nevertheless, to ensure classifier convergence, we employ a Kreĭn SVM [25] (following Option d), which is capable of handling positive definite and indefinite matrices. Unlike the other options, this one does not have to modify the kernel values at testing time. Hence, we refer to WTK as a *kernel*, with the added caveat that for some data sets, *the kernel matrix is indefinite*. We also tried Options (a) and (b) but none of them exhibited a significantly better performance.

Algorithm 1 Wasserstein Time Series Kernel

Input: Time series for training and testing $\mathcal{T}_{\text{train}}, \mathcal{T}_{\text{test}}$; subsequence length w ; kernel weight factor λ
Output: $\mathcal{K}^{\text{train}}, \mathcal{K}^{\text{test}}$

```
1: // Extract subsequences
2:  $\mathcal{S}^{\text{train}} \leftarrow \text{SUBSEQUENCES}(\mathcal{T}_{\text{train}}, w)$ 
3:  $\mathcal{S}^{\text{test}} \leftarrow \text{SUBSEQUENCES}(\mathcal{T}_{\text{test}}, w)$ 
4: for  $T_i \in \mathcal{T}_{\text{train}}$  do
5:   for  $T_j \in \mathcal{T}_{\text{train}}$  do
6:     // Wasserstein distance calculation (train)
7:      $\mathcal{D}_{ij}^{\text{train}} \leftarrow W_1(\mathcal{S}_i^{\text{train}}, \mathcal{S}_j^{\text{train}})$ 
8:   end for
9:   for  $T_k \in \mathcal{T}_{\text{test}}$  do
10:    // Wasserstein distance calculation (test)
11:     $\mathcal{D}_{ik}^{\text{test}} \leftarrow W_1(\mathcal{S}_i^{\text{train}}, \mathcal{S}_k^{\text{test}})$ 
12:   end for
13: end for
14: // Kernel matrix calculation
15:  $\mathcal{K}^{\text{train}} \leftarrow \exp(-\lambda \mathcal{D}^{\text{train}})$ 
16:  $\mathcal{K}^{\text{test}} \leftarrow \exp(-\lambda \mathcal{D}^{\text{test}})$ 
17: return  $\mathcal{K}^{\text{train}}, \mathcal{K}^{\text{test}}$ 
```

Comparison to KEMD: Our method differs substantially from the KEMD method [18], which can be considered a histogram intersection kernel that treats each time series as a one-dimensional distribution. By contrast, our approach measures the distance between high-dimensional distributions of *subsequences* and it is therefore better suited to capture long-distance similarities of subsequences and time series.

E. Complexity and implementation

Our method comprises the following parts: (a) subsequence extraction, (b) subsequence distance calculation, and (c) Wasserstein metric calculation. Letting n refer to the number of time series, we have at most $s := m - w + 1$ subsequences per time series. The extraction process is dominated by m , the length of the time series, leading to a total complexity of $\mathcal{O}(nm)$. This is a preprocessing step that we share with other methods such as [3] and [26]. Then, the following operations are performed for each pair of time series. Computing the distances between subsequences of two time series requires s^2 distance calculations, each of which has to process a sequence of length w . Hence, in the worst case, this calculation has a complexity of $\mathcal{O}(s^2w)$; it is possible to reduce this quite significantly, at least in the case of Euclidean distances, by reusing calculations. Finally, evaluating Eq. 5 for two time series has dimensions $s \times s$ [27]. Asymptotically, the runtime of all parts can be summarised as $\mathcal{O}(n^2m^3 \log m)$, because m is an upper bound on the number of subsequences of a fixed length. This worst-case approximation can be sped up using approximation schemes such as Sinkhorn iterations [27]. In our experimental setup, which relied on a basic entropic regularisation scheme without hyperparameter search, we could observe a slight speed improvement, but the accuracies obtained via a straightforward Sinkhorn approximation were not competitive with the results of the exact distance computation, which is why we do not discuss it further.

Algorithm 1 depicts a pseudocode description of our method. Our implementation uses Python 3.7 and POT [28]. Our code is publicly available².

V. EXPERIMENTS

We perform all experiments on the 85 ‘UCR Time Series Archive’ data sets [11]. Each data set consists of a predefined train/test split of varying sizes and time series of varying lengths, though in each data set the time series length is fixed.³ We perform different experiments for evaluating the performance of our proposed approach. Specifically, we are interested in evaluating the accuracies obtained using WTK and compare it with (a) different subsequence-based methods, (b) established baselines such as DTW, and (c) state-of-the-art methods for TSC.

a) Comparison partners: The compared methods range from neural network architectures over shapelet-based classifiers such as Shapelet Transform (ST) [29] and Learned Shapelets (LS) [30] to ensemble methods such as Elastic Ensemble (EE) [31], FLAT-COTE [32], and HIVE-COTE [33]. Other algorithms include dictionary-based classifiers such as BOSS [34], a classifier based on a combination of DTW distances and SAX histograms (DTW_F) [35], and SAXVSM [36]. Furthermore, a rotation forest [37] (RotF) and a random forest are considered. Other baselines include a 1-nearest neighbour based on Euclidean Distance (E-1NN) and a Bayesian network (BN).

b) Training and evaluation: We evaluate the classification accuracy on the test set and select the parameters on the training set via 5-fold cross-validation using a Kreĭn SVM classifier [25] with the following parameter grid: (a) $\gamma \in \{10^{-5}, 10^{-4}, \dots, 10^3\}$ for the RBF kernel, (b) $\lambda \in \{10^{-4}, 10^{-3}, \dots, 10\}$ for WTK, (c) $C \in \{10^{-3}, 10^{-2}, \dots, 10^3\}$ for the SVM classifier. We also vary length w of the subsequences used for each subsequence-based method by checking potential values of w as 10%, 30%, and 50% of the original time series length. We initially included k -NN classifiers based on Eq. 5 in our experimental setup, but they perform more than 3% worse on average, so we do not discuss them any further.

A. Comparison to other kernels

We compare our method with a linear kernel and an RBF kernel, trained on subsequences of the same length. For the linear kernel, we expect bad predictive performance since it effectively degrades into a comparison of the means of two time series, as outlined in Section II. By contrast, the RBF kernel has already shown favourable performance [13], but we are the first to include it in a large-scale comparison. The RBF kernel compares each pair of subsequences *independently*, whereas WTK is able to capture similarities between the *entire distributions* of subsequences of two time series. Figure 2a depicts the results for all UCR data sets. Our kernel outperforms the simple linear kernel in *all* cases; a straightforward application

²<https://github.com/BorgwardtLab/WTK>

³Please refer to <http://www.timeseriesclassification.com> for details.

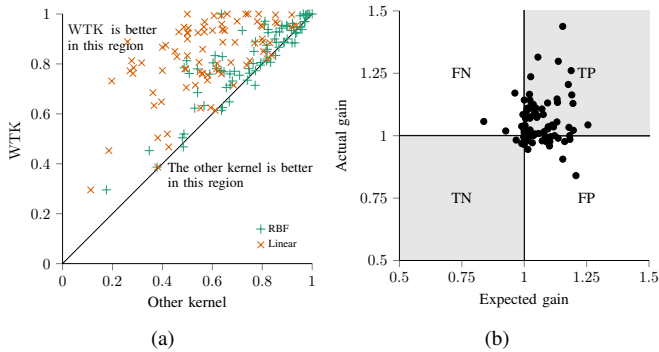


Figure 2. (a) Comparison of the predictive accuracy of our method with the linear and the RBF kernels for the ‘UCR Time Series Archive’ data sets. (b) A ‘Texas Sharpshooter’ plot, comparing the *expected* gains of our method with the *actual* gains, relative to DTW-1-NN. TP (TN) points indicate correct prediction of superior (inferior) method performance. FP (FN) points overestimate (underestimate) method performance.

of simple instances of the \mathcal{R} -convolution framework with linear subsequence kernels is indeed meaningless for TSC. Moreover, we outperform the RBF kernel on all but 12 data sets. The accuracy difference for the points below the diagonal is negligible ($\approx 2.2\%$). Hence, the performance of our method is not caused by considering subsequences per se, but by considering the distribution of subsequence similarities. In terms of computational runtime, we observe that the linear kernel, the RBF kernel, and our method WTK, each of which uses distances between subsequences, perform asymptotically the same. Therefore, computing the Wasserstein distance is not the driving factor in complexity.

B. Comparison to DTW-1-NN

Following best practices in TSC literature [11], we also compare our method with a DTW-1-NN baseline and create a ‘Texas Sharpshooter’ plot, which shows the *expected* gain (estimated from the training data set via 5-fold cross-validation) and the *actual* gain (calculated on the test data set). Figure 2b depicts the results: most points fall into the TP or TN quadrants, implying that we either correctly predicted that we outperform DTW-1-NN (TP), or that we are outperformed by DTW-1-NN (TN). Points in FN are a ‘happy surprise’ because we predicted that our method would do worse than it actually does. The problematic quadrant, the FP quadrant, only contains few points; moreover, the accuracy differences in this quadrant are relatively minor. Hence, the plot supports our claim that WTK is better than the DTW-1-NN baseline as most points are in the TP quadrant.

C. Comparison to the state of the art (SOTA)

Next, we compare our method with the SOTA. To this end, we collected the accuracies of *all* published methods of [12], as well as two neural network baselines [38] with their classification performances from [39]. This resulted in 40 methods, although the availability of comparison partners depends on the data set. For each data set, we picked the best-performing method (using the published train/test

split), referring to it as the respective state-of-the-art method for the data set. In total, we compare our method with the best of 40 other methods to ensure the most honest and comprehensive testing scenario. We observe that our method outperforms *all* state-of-the-art methods on six data sets, i.e. `DistalPhalanxTW`, `DistalPhalanxOutlineAgeGroup`, `MiddlePhalanxOutlineAgeGroup`, `Earthquakes`, `ECG5000`, and `FordB`. In total, we exhibit at least *equal* accuracy as any state-of-the-art method on 12 data sets: we are on a par with the SOTA on `BeetleFly`, `Coffee`, `ECGFiveDays`, `Plane`, `ShapeletSim`, and `Trace`.

Performance breakdown: In numerous other data sets, we are almost as good as the SOTA. Table I gives a more precise breakdown of these performance differences. We compare our method with HIVE-COTE, the best-performing ensemble method, and with KEMD, a conceptually similar competitor. Each table entry shows the fraction of data sets for which the condition of the first column is fulfilled. We observe that, while we do not match the performance of HIVE-COTE, an ensemble method, for the majority of all data sets, our performance difference is less than 5%. At the same time, we see that the performance of KEMD is highly variable, leading to favourable performance on some data sets, while being completely outperformed on the majority of them.

Statistical analysis: To strengthen our claims, Figure 3 shows a *critical difference plot* [40] that depicts our method and other selected methods. For a significance level of $\alpha = 0.05$, the plot shows that there is no statistically significant difference in the performance of our method and these best-performing classifiers. The best-performing classifiers are either deep neural networks or large ensembles and therefore heavily parametrised, thus showing a good promise for the generalisation performance of our method.

VI. CONCLUSION

We developed a novel subsequence-based kernel that uses the Wasserstein distance as an effective similarity measure for time series classification. To prove the benefits of our method, we performed a large-scale evaluation on the ‘UCR Time Series Archive’ data sets that showed that our method outperforms some of the state-of-the-art time series classification algorithms while also displaying favourable generalisation properties. Currently, our method only considers fixed-size subsequence lengths and we plan to include varying-length subsequences in future work. However, there is a computational burden in their selection, as well as additional

Table I
DIFFERENCE (Δ) IN MEAN ACCURACY WITH RESPECT TO THE SOTA

Δ	WTK	HIVE-COTE	KEMD
$\Delta \geq 0$	14.1 %	36.5 %	4.7 %
$0\% > \Delta \geq -5\%$	44.7 %	34.1 %	15.3 %
$-5\% > \Delta \geq -10\%$	24.7 %	18.8 %	7.1 %
$-10\% > \Delta \geq -15\%$	8.2 %	1.2 %	16.5 %
$-15\% > \Delta \geq -20\%$	4.7 %	7.1 %	9.4 %
$-20\% > \Delta$	3.5 %	2.4 %	47.1 %

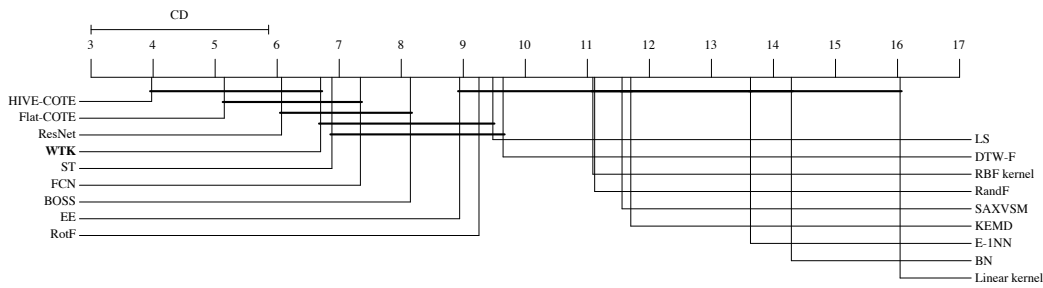


Figure 3. Critical difference plot, comparing our method (shown in bold) with several other methods. The scale indicates the average rank of each method in terms of test accuracy for all data sets. The classification performances of methods sharing horizontal bars are not significantly different. We observe that there is no statistically significant difference between the performance of our method and state-of-the-art ensemble methods.

constraints because no commonly-used metric for comparing subsequences of varying lengths exists. We also plan to extend our method to multivariate time series. Finally, effective subsequence preselection techniques will reduce the computational burden while potentially increasing the predictive performance. In conclusion, this work demonstrates the merits of subsequence-based kernels for time series classification, and the potential of optimal transport measures in data mining.

ACKNOWLEDGMENTS

This study was funded in part by the Alfried Krupp Prize for Young University Teachers of the Alfried Krupp von Bohlen und Halbach-Stiftung (K.B.). EG was funded by Horizon 2020 project CDS-QUAMRI, Grant No. 634541.

REFERENCES

- [1] U. Rebbapragada *et al.*, “Finding anomalous periodic time series,” *Machine learning*, vol. 74, no. 3, pp. 281–313, 2009.
- [2] C. Bock *et al.*, “Association mapping in biomedical time series via statistically significant shapelet mining,” *Bioinformatics*, vol. 34, no. 13, pp. i438–i446, 2018.
- [3] L. Ye and E. Keogh, “Time series shapelets: A new primitive for data mining,” in *KDD*, (New York, NY, USA), pp. 947–956, ACM, 2009.
- [4] D. Haussler, “Convolution kernels on discrete structures,” tech. rep., University of California at Santa Cruz, 1999.
- [5] G. Peyré *et al.*, “Computational optimal transport,” *Foundations and Trends® in Machine Learning*, vol. 11, no. 5–6, pp. 355–607, 2019.
- [6] J.-P. Vert, “The optimal assignment kernel is not positive definite,” *arXiv e-prints*, 2008.
- [7] D. Oglic and T. Gärtner, “Learning in reproducing kernel Kreĭn spaces,” in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 3859–3867, PMLR, 2018.
- [8] B. Haasdonk, “Feature space interpretation of SVMs with indefinite kernels,” *IEEE TPAMI*, vol. 27, no. 4, pp. 482–492, 2005.
- [9] T. Rakthanmanon *et al.*, “Searching and mining trillions of time series subsequences under dynamic time warping,” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 262–270, 2012.
- [10] E. Keogh and J. Lin, “Clustering of time-series subsequences is meaningless: implications for previous and future research,” *Knowledge and Information Systems*, vol. 8, no. 2, pp. 154–177, 2005.
- [11] Y. Chen *et al.*, “The UCR Time Series Classification Archive,” 2015.
- [12] A. Bagnall *et al.*, “The great time series classification bake off,” *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.
- [13] S. Rüping, “SVM kernels for time series analysis,” Tech. Rep. 43, Technical University of Dortmund, 2001.
- [14] G. Wachman *et al.*, “Kernels for periodic time series arising in astronomy,” in *Machine Learning and Knowledge Discovery in Databases*, pp. 489–505, Springer, 2009.
- [15] M. Cuturi *et al.*, “A kernel for time series based on global alignments,” in *ICASSP*, vol. 2, pp. 413–416, 2007.
- [16] M. Cuturi, “Fast global alignment kernels,” in *28th International Conference on Machine Learning (ICML)*, pp. 929–936, 2011.
- [17] P.-F. Marteau and S. Gibet, “On recursive edit distance kernels with application to time series classification,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 6, pp. 1121–1133, 2015.
- [18] M. R. Daliri, “Kernel earth mover’s distance for EEG classification,” *Clinical EEG and Neuroscience*, vol. 44, no. 3, pp. 182–187, 2013.
- [19] Y. Rubner *et al.*, “The Earth Mover’s Distance as a metric for image retrieval,” *IJCV*, vol. 40, no. 2, pp. 99–121, 2000.
- [20] C. Villani, *Optimal transport: Old and new*. Springer, 2008.
- [21] M. Togninalli *et al.*, “Wasserstein Weisfeiler–Lehman graph kernels,” *arXiv e-prints*, 2019.
- [22] C.-C. M. Yeh *et al.*, “Matrix Profile I: All Pairs Similarity Joins for Time Series: A unifying view that includes motifs, discords and shapelets,” in *IEEE ICDM*, pp. 1317–1322, 2016.
- [23] A. Feragen *et al.*, “Geodesic exponential kernels: When curvature and linearity conflict,” in *IEEE CVPR*, pp. 3032–3042, 2015.
- [24] L. Wu *et al.*, “D2KE: From distance to kernel and embedding,” *arXiv e-prints*, 2018.
- [25] G. Loosli *et al.*, “Learning SVM in Kreĭn spaces,” *IEEE TPAMI*, vol. 38, no. 6, pp. 1204–1216, 2015.
- [26] S. Gharghabi *et al.*, “Matrix Profile XII: MPdist: A novel time series distance measure to allow data mining in more challenging scenarios,” in *IEEE ICDM*, pp. 965–970, 2018.
- [27] J. Altschuler *et al.*, “Near-linear time approximation algorithms for optimal transport via sinkhorn iteration,” in *NIPS*, pp. 1964–1974, 2017.
- [28] R. Flamary and N. Courty, “POT: Python Optimal Transport,” 2017.
- [29] A. Bostrom and A. Bagnall, “Binary Shapelet Transform for Multiclass Time Series Classification,” in *DaWaK*, pp. 257–269, Springer, 2015.
- [30] J. Grabocka *et al.*, “Learning time-series shapelets,” in *KDD*, pp. 392–401, 2014.
- [31] J. Lines and A. Bagnall, “Time series classification with ensembles of elastic distance measures,” *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.
- [32] A. Bagnall *et al.*, “Time-Series Classification with COTE,” *IEEE TKDE*, vol. 27, no. 9, pp. 2522–2535, 2015.
- [33] J. Lines *et al.*, “HIVE-COTE: The Hierarchical Vote Collective of Transformation-Based Ensembles for Time Series Classification,” in *IEEE ICDM*, pp. 1041–1046, Dec 2016.
- [34] P. Schäfer, “The BOSS is concerned with time series classification in the presence of noise,” *Data Mining and Knowledge Discovery*, vol. 29, no. 6, pp. 1505–1530, 2015.
- [35] R. J. Kate, “Using dynamic time warping distances as features for improved time series classification,” *Data Mining and Knowledge Discovery*, vol. 30, no. 2, pp. 283–312, 2016.
- [36] P. Senin and S. Malinchik, “SAX-VSM: Interpretable Time Series Classification Using SAX and Vector Space Model,” in *IEEE ICDM*, pp. 1175–1180, 2013.
- [37] J. J. Rodriguez *et al.*, “Rotation Forest: A New Classifier Ensemble Method,” *IEEE TPAMI*, vol. 28, no. 10, pp. 1619–1630, 2006.
- [38] Z. Wang *et al.*, “Time series classification from scratch with deep neural networks: A strong baseline,” in *IJCNN*, pp. 1578–1585, 2017.
- [39] H. I. Fawaz *et al.*, “Deep learning for time series classification: a review,” *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 1–47, 2019.
- [40] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.