On a Family of Decomposable Kernels on Sequences

Andrea Baisero andrea.baisero@ipvs.uni-stuttgart.de
Institute for Parallel and Distributed Systems
Machine Learning and Robotics, University of Stuttgart
Stuttgart, Germany

Florian T. Pokorny fpokorny@csc.kth.se
Carl Henrik Ek chek@csc.kth.se
School of Computer Science and Communication
Royal Institute of Technology, KTH
Stockholm, Sweden

Abstract

In many applications data is naturally presented in terms of orderings of some basic elements or symbols. Reasoning about such data requires a notion of similarity capable of handling sequences of different lengths. In this paper we describe a family of Mercer kernel functions for such sequentially structured data. The family is characterized by a decomposable structure in terms of symbol-level and structure-level similarities, representing a specific combination of kernels which allows for efficient computation. We provide an experimental evaluation on sequential classification tasks comparing kernels from our family of kernels to a state of the art sequence kernel called the Global Alignment kernel which has been shown to outperform Dynamic Time Warping.

Keywords: Kernel, Sequences

1. Introduction

Many types of data have inherent sequential structure. Sequences of letters in computational linguistics, series of images in computer vision or cell structures in computational biology and arbitrary data sets depending on a parameter such as time provide familiar examples of such data. It is hence not surprising that there exists a significant amount of work focused on representing such data. In (Rieck, 2011) the author reviews and broadly categorizes sequential similarity measures into three main categories: *bag-of-words*, *edit-distance* and *string-kernel* based methods. Bag-of-words (Harris, 1970) based similarity measures translate the notion of a sequence to a distribution over certain sub-sequences (*i.e.* words in natural language processing) of the sequence itself, meaning that such measures only encode the sequential structure up to the length of the sub-sequence and disregard information about word order. As such, Bag-of-words methods require us to be able to identify significant sub-sequences (the words), which is not always obvious for sequences arising outside natural language. Nevertheless, this approach captures some structure and, as the sequential data is translated into a vector space whose basis consists of elementary subsequences, it allows us to interpret the data and enables us to use well-developed learning methods for such vectorial data. Techniques based on edit distances (Damerau, 1964; Levenshtein,

1966) relate sequences by defining a transformation from one sequence to the other and associating a cost to the transformation. Edit distances can be very useful if the notion of cost with respect to different transformations is well grounded. The third category refers to (dis)similarity measures defined by implicitly specifying an inner-product space through a kernel function between sequences. String kernels (Lodhi et al., 2002; Rousu and Shawe-Taylor, 2006) were proposed in Computational Linguistics, where data consists of sequences (text) of discrete symbols (letters). The (dis)similarity measure is defined in terms of "gaps" between symbols in the two sequences. String kernels are a specific instance of a larger class of kernel functions referred to as rational kernels (Cortes et al., 2004). Rational kernels are related to weighted automata (Mohri, 2009) and define inner products from the specific sequential structure described by the automata. In this paper, we will focus on a new family of kernel based (dis)similarity measures.

The contributions of this work are in particular: *a)* A generic approach for the construction of sequence kernels which scales $\mathcal{O}(nm)$ in the lengths n, m of the input strings. *b)* The kernel decomposes intuitively into structure-level and symbol-level similarities. Compared to previous approaches, the structure of the symbol space can be encoded by any Mercer kernel. *c)* We show that a recently proposed intuitive (dis)similarity measure on sequences (Baisero et al., 2013), is positive definite kernel and falls into our class. *d)* We compare and evaluate several kernels from our family which perform favourably against the state of the art Global Alignment kernel.

2. Related work

The three main sequence similarity approaches discussed above are all based on the concept that sequence similarity is defined in terms of discrete unordered symbols, and the similarity between two symbols $a, b, \in \Sigma$ is typically is defined by zero if $a \neq b$ and one otherwise. However, for many types of data the symbol space Σ might be continuous, and we might in fact have a natural similarity measure on Σ itself. As an example, consider the problem of matching two discretized waveforms $\alpha = [\alpha_1, \ldots, \alpha_n]$, $\beta = [\beta_1, \ldots, \beta_m]$ where $\alpha_i, \beta_i \in \mathbb{R} = \Sigma$ and where there exists a natural distance $\|a - b\|$ for $a, b \in \Sigma = \mathbb{R}$. A popular similarity measure closely related to edit distances is Dynamic Time Warping (Sakoe and Chiba, 1978; Müller, 2007). It provides a similarity measure based on the cost of aligning two sequences such that the sum of matching each element is minimized. This measure does not by itself correspond to a positive definite kernel function (Bahlmann et al., 2002) and hence lacks a geometrical interpretation. One approach has been to use the dynamic time warping distance inside a radial basis exponential kernel function (Lei and Sun, 2007; Bahlmann et al., 2002). However this still suffers from the drawback that dynamic time warping is not a kernel itself. Even though non-positive kernels have been shown to be useful (Haasdonk, 2005) in practice, they lack a geometrical interpretation and the mathematical justification which makes the use of kernel methods so appealing.

Motivated by the intuition for the definition of dynamic time warping, (Cuturi et al., 2007) developed a related similarity measure which in fact corresponds to a valid kernel function for sequences. Here a (dis)similarity function is defined by summarizing all possible alignments between two sequences through a 'soft-min' rather than using only the minimal cost alignment as in dynamic time warping. Importantly, compared to previous kernels on

sequences, this kernel is capable of incorporating a structured non-discrete symbol space Σ. The resulting kernel is referred to as the Global Alignment kernel and was shown to outperform Dynamic Time Warping for sequence classification. However, to be a valid Mercer kernel, the structure of the symbol space Σ have to be induced by a specific class of kernel functions. Further, it strongly favors small sequence perturbations over larger perturbations which reduces the ability of the kernel to generalize example data. Some of these issues have been addressed in (Cuturi, 2010) where only a subset of the possible alignments contributes to the inner-product.

Another approach was taken in (Baisero et al., 2013), where we proposed a (dis)similarity measure called the Path kernel. Just like the Global Alignment kernel, this kernel is defined by reasoning about the (dis)similarity of all possible alignments of two sequences. In experiments, the Path kernel performed better than the Global Alignment kernel for a set of experiments both with respect to accuracy and computational cost. We will show that this kernel naturally falls into a class of kernels that we will define in this work, thus proving that it is positive semi-definite[1].

3. On the construction of sequence kernels

We are interested in finite sequences $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_{|\mathbf{s}|})$, with symbols $\mathbf{s}_i \in \Sigma$, belonging to a symbol space Σ which can be discrete or continuous. We denote the set of such finite sequences by $Seq(\Sigma)$ and are interested in studying combinations of Mercer kernel functions on symbols $k_\Sigma : \Sigma \times \Sigma \to \mathbb{R}$ that yield valid Mercer kernels on a sequential level $Seq(\Sigma)$. We follow the convention of calling a kernel $k : X \times X \to \mathbb{R}$ positive definite if $\sum_{i,j=1}^{n} c_i k(x_i, x_j) c_j \geqslant 0$ for any finite subset $\{x_1, \ldots, x_n\} \subset X$, $n \in \mathbb{N}$ and any $\{c_1, \ldots, c_n\} \subset \mathbb{R}$. Let us now describe a novel general approach towards the construction of such kernels for sequences belonging to $Seq(\Sigma)$:

Lemma 3.1 *Let $k_\Sigma : \Sigma \times \Sigma \to \mathbb{R}$ be a continuous positive definite kernel on Σ, where Σ is a separable metric space and let $k_S : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ be a positive definite kernel on integers. Then the kernel*

$$k(\mathbf{s}, \mathbf{t}) = \sum_{i=1}^{|\mathbf{s}|} \sum_{j=1}^{|\mathbf{t}|} k_\Sigma(\mathbf{s}_i, \mathbf{t}_j) k_S(i, j), \tag{1}$$

defined for any finite sequences $\mathbf{s}, \mathbf{t} \in Seq(\Sigma)$, $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_{|\mathbf{s}|})$ and $\mathbf{t} = (\mathbf{t}_1, \ldots, \mathbf{t}_{|\mathbf{t}|})$ is also positive definite.

Proof Observe that both k_Σ and k_S can be trivially extended to kernels on $\Sigma \times \mathbb{N}$ by $K_1((\mathbf{s}, i), (\mathbf{t}, j)) = k_\Sigma(\mathbf{s}_i, \mathbf{t}_j)$, $K_2((\mathbf{s}, i), (\mathbf{t}, j)) = k_S(i, j)$ for $\mathbf{s}_i, \mathbf{t}_j \in \Sigma$ and $i, j \in \mathbb{N}$. Now $K((\mathbf{s}, i), (\mathbf{t}, j)) = K_1((\mathbf{s}, i), (\mathbf{t}, j)) K_2((\mathbf{s}, i), (\mathbf{t}, j))$ is a positive kernel on $U = \Sigma \times \mathbb{N}$. Let X, Y be finite subsets of U. According to Lemma 1, (Haussler, 1999), the kernel

$$L(X, Y) = \sum_{x \in X, y \in Y} K(x, y)$$

1. Note that, the Path kernel defined in (Baisero et al., 2013) is not related to the special Rational kernel proposed in (Takimoto and Warmuth, 2003), which is also referred to as a Path kernel

is then also positive definite. Note that a sequence $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n)$ corresponds to a subset $X = \{(\mathbf{s}_1, 1), (\mathbf{s}_2, 2), \ldots, (\mathbf{s}_n, n)\}$, and thus the above kernel is positive definite. ■

Note that any countable discrete space and any finite dimensional vector space can be given the structure of a separable metric space. If Σ is discrete and countable, any kernel on Σ is trivially continuous with respect to the discrete topology. Note also that, while the above result readily follows from the work on convolution kernels by (Haussler, 1999), the above natural class of kernels has – to the best of our knowledge – not been studied or formulated in this manner. This might be partially, because classical kernels coming from natural language processing often only consider similarity measures on the symbol space Σ directly.

We observe that the family of kernels described above relates all pairs of the input sequences' symbols using k_Σ and adjusts these values according to similarity of positions of the symbols within the sequences, as measured by k_S. An added benefit of the proposed family of kernels is their relative computational simplicity, since kernel evaluations scale like $\mathcal{O}(|s||t|)$ in the length of the input strings $\mathbf{s}, \mathbf{t}$. Noting that,

$$k(\mathbf{s}, \mathbf{t}) = tr(K_\Sigma(\mathbf{s}, \mathbf{t})^T K_S(|\mathbf{s}|, |\mathbf{t}|)), \tag{2}$$

where $[K_\Sigma(\mathbf{s}, \mathbf{t})]_{ij} = k_\Sigma(\mathbf{s}_i, \mathbf{t}_j)$, $[K_S(|\mathbf{s}|, |\mathbf{t}|)]_{ij} = k_S(i, j)$, $i = 1, \ldots, |\mathbf{s}|$, $j = 1, \ldots, |\mathbf{t}|$ and tr denotes the trace, we observe that the matrix K_S can be pre-computed once the maximal length of any sequence in a data-set is known. The evaluation of the kernel is then just a trace of a matrix product which can be efficiently implemented.

In a typical scenario k_S and k_Σ might also depend on parameters $\theta_S \in \mathbb{R}^n$ and $\theta_\Sigma \in \mathbb{R}^m$. These parameters can be set through cross-validation but they can also be learned if the gradients of the kernel functions with respect to these parameters can be computed. If we wish to use the kernel to represent a functional relationship $f : \mathbf{s} \mapsto \mathbf{y}$, where $\mathbf{y} \in \mathbb{R}^d$, we can encode a preference over the mapping f by a Gaussian process (Rasmussen and Williams, 2006). If the co-variance in the output space is encoded by a sequence kernel k, the parameters, θ_Σ and θ_S can then be learned by maximizing the marginal likelihood of the model. In order to accommodate classification in a Gaussian process framework, the regression noise is usually squashed (Rasmussen and Williams, 2006) rendering the integration required to reach the marginal likelihood infeasible. However, it has been observed that learning the parameters for a classification task with $1 - C$ encoding, where each class is encoded using a binary variable, and a Gaussian noise assumption works well in practice (Kapoor et al., 2009).

4. Examples of Sequence Kernels

In this paper we will focus on kernels from the family in Lemma 3.1. A straightforward approach to formulate such a sequence kernel would be to pick a familiar kernel k_S, where $|i - j|$ determines the impact on k. This can be implemented by a stationary kernel k_S such as an exponential kernel:

Corollary 4.1 *For $\alpha > 0$, the function $k_e : Seq(\Sigma) \times Seq(\Sigma) \to \mathbb{R}$ given by,*

$$k_e(\mathbf{s}, \mathbf{t}) = \sum_{i=1}^{|\mathbf{s}|} \sum_{j=1}^{|\mathbf{t}|} k_\Sigma(\mathbf{s}_i, \mathbf{t}_j) e^{-\frac{\|i-j\|^2}{\alpha}}, \tag{3}$$

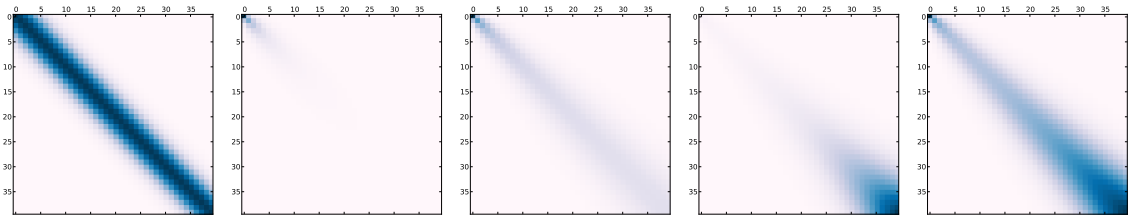

Figure 1: *The above figure shows five different structure kernel matrices. The left-most image depicts the exponential kernel while the remaining four show the structure kernel k_Γ for the path sequence kernel for varying parameter values. The values of $C_d = \{0.3, 0.35, 0.35, 0.3\}$ and $C_{hv} = \{0.3, 0.33, 0.37, 0.37\}$. Changing the parameters C_d and C_{hv} emphasizes different parts of the alignment highlighting the non-stationary structure of how the contribution of different parts of a sequences is accumulated.*

is a valid kernel on $Seq(\Sigma)$ corresponding to the exponential structure kernel on $\mathbb{N}$.

Similarly, we can now define kernels by defining a kernel k_S on integers by restricting any known kernel on $\mathbb{R}$ to the integers. Examples include the polynomial kernels $k_S(i,j) = (ij + c)^d$, the perceptron kernel, *etc.* While the above kernel follows readily from the definition of our class of sequence kernels, we would now like to focus on a secondary viewpoint which, as we will show, also leads to a sequence kernel as in Lemma 3.1. In (Baisero et al., 2013), the authors proposed a novel (dis)similarity measure $k_p : Seq(\Sigma) \times Seq(\Sigma) \to \mathbb{R}$ which can be defined most elegantly in a recursive fashion as,

$$k_p(\mathbf{s}, \mathbf{t}) = \begin{cases} k_\Sigma(\mathbf{s}_1, \mathbf{t}_1) & \\ \quad + C_{\mathrm{hv}} k_p(\mathbf{s}_{2:}, \mathbf{t}) & |\mathbf{s}| \geqslant 1 |\mathbf{t}| \geqslant 1 \\ \quad + C_{\mathrm{hv}} k_p(\mathbf{s}, \mathbf{t}_{2:}) & C_{\mathrm{d}} \geqslant 0 \\ \quad + C_{\mathrm{d}} k_p(\mathbf{s}_{2:}, \mathbf{t}_{2:}) & C_{\mathrm{hv}} \geqslant 0 \\ 0 & \text{otherwise,} \end{cases} . \tag{4}$$

where $\mathbf{t}_{\mathbf{2}:}$ denotes the sequence obtained by removing the first symbol from $\mathbf{t} \in Seq(\Sigma)$. The recursive formulation above can be interpreted as accumulating information from all possible alignments of two strings. An alignment of two strings $\mathbf{s}$ and $\mathbf{t}$ is defined by a path γ through a matrix $\mathbf{M}$ of size $|\mathbf{s}| \times |\mathbf{t}|$ from element $[\mathbf{M}]_{11}$ to $[\mathbf{M}]_{|\mathbf{s}||\mathbf{t}|}$. Each path defines a different alignment in terms of "stretches" of a sequence see Figure 2. Each path is decomposed into series of simple operations which have a different effect, parametrized by C_{d} and C_{hv}, on the final similarity measure. The cardinality of the set of paths, and therefore of the alignments, for two sequences $\mathbf{s}$ and $\mathbf{t}$ is the Delannoy number $D(|\mathbf{s}|, |\mathbf{t}|)$. In addition to the recursive formulation above, (Baisero et al., 2013) also showed that the resulting function can be written in a similar form to Eq. 1 where $k_S = k_\Gamma$ and,

$$k_\Gamma(i,j) = \sum_{d=0}^{\min(i,j)-1} C_{\mathrm{hv}}^{i+j-2-2d} C_{\mathrm{d}}^d \frac{(i+j-2-d)!}{(i-1-d)!(j-1-d)!d!}. \tag{5}$$

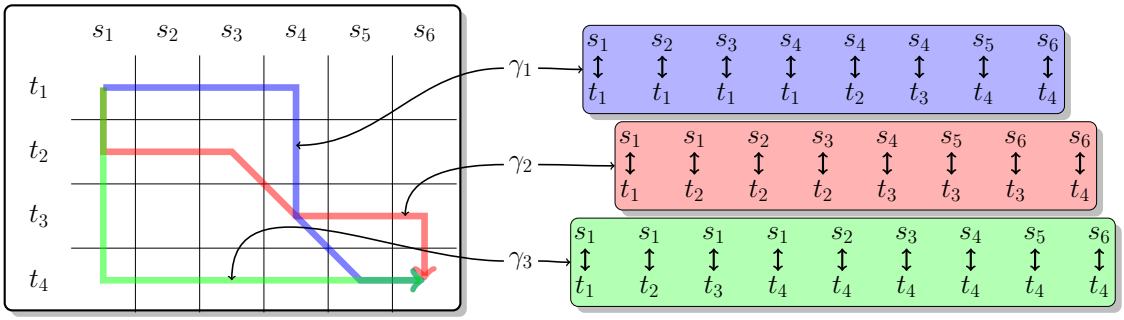

Figure 2: *The image above depicts the Path kernel interpreted as an alignments of paths. Two sequences* $\mathbf{t} = [\mathbf{t}_1, \dots, \mathbf{t}_4]$ *and* $\mathbf{s} = [\mathbf{s}_1, \dots, \mathbf{s}_6]$ *and three separate alignments* $\gamma_{1,2,3}$ *are shown. The matrix to the left shows how a subset of the possible alignments between* $\mathbf{s}$ *and* $\mathbf{t}$ *can be constructed as* paths *between the top-left to the bottom-right element of the matrix to the right. On the right, the three different alignments generated from the paths on the left are shown.*

While the recursive definition of k_p is natural, since it assigns a cost for diagonal and off-diagonal moves in a matrix, the above formula seems rather unintuitive. While (Baisero et al., 2013) did not provide a proof of positive definiteness, we will now show that the path kernel k_p does in fact define a positive definite kernel and that k_p naturally falls into the class of kernels considered here.

In order to prove that k_p is a positive definite kernel, we now need to show that $k_\Gamma : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ is indeed a kernel on integers. First lets recall that the Gamma function $\Gamma : \mathbb{C} \to \mathbb{R}$ is defined by,

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t} \mathrm{d}t, \tag{6}$$

for $z \in \mathbb{C}$, $\mathcal{R}e(z) > 0$ and that we have $\Gamma(n) = (n-1)!$ for $n \in \mathbb{N}$. Let us now think of the factorial as a curious example of a positive Mercer kernel on integers:

Lemma 4.2 *Let* $d \in \mathbb{Z}$ *and* $X_d = \{x \in \mathbb{N} : x \geqslant \frac{d}{2}\}$. *The function* $k : X_d \times X_d \to \mathbb{R}$ *defined by,*

$$k(x, x') = (x + x' - d)!,$$

is a positive definite kernel on X_d *corresponding to the feature mapping* $\psi_d : X_d \to L^1(\mathbb{R}_{\geqslant 0})$ *mapping* $x \in X_d$ *to the function* $f_x(t) = t^{x - \frac{d}{2}} e^{-\frac{t}{2}}$. *I.e., considering the standard inner product on* $L^1(\mathbb{R}_{\geqslant 0})$ *given by* $\langle f, g \rangle = \int_0^\infty f(t) g(t) \mathrm{d}t$ *for two integrable functions* $f, g \in L^1(\mathbb{R}_{\geqslant 0})$, *we have* $k(x, x') = \langle f_x, f_y \rangle$.

Proof The result follows directly from the above integral formula for $\Gamma(x + x' - d + 1)$. ■

Note that, if $C_\mathrm{d} \geqslant 0$, we can use the idea of the above lemma and write $k_\Gamma(i, j) =$

$\sum_{d=0}^{min(i,j)-1}\langle\varphi_d(i),\varphi_d(j)\rangle$, where $\varphi_d : X_{2+2d} \to L^1(\mathbb{R}_{\geqslant 0})$ is the feature map mapping integers to functions which is given by,

$$(\varphi_d(i))(t) = \frac{C_{\mathrm{d}}^{\frac{d}{2}} C_{\mathrm{hv}}^{i-1-d}}{(i-1-d)!} t^{i-1-\frac{d}{2}} e^{-\frac{t}{2}},$$

and $\langle f, g\rangle$ is again the inner-product of functions obtained by integration. $k_d(i,j) = \langle\varphi_d(i),\varphi_d(j)\rangle$ is a kernel on $X_{2d+2} = \{x \in \mathbb{N} : x > d\}$. Note that the condition $d \leqslant \min(i,j) - 1$, $i, j \in \mathbb{N}$ in the summation appearing in the definition of k_Γ is equivalent to $d < i$ and $d < j$, i.e. $i, j \in X_{2d+2}$. Let us call the extension of k_d to $\mathbb{N} \times \mathbb{N}$ $\hat{k}_d$, so that $\hat{k}_d(i,j) = 0$ if i or $j \notin X_{2d+2}$ and $\hat{k}_d(i,j) = k_d(i,j)$ if $i, j \in X_{2d+2}$. Then $\hat{k}_d : \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ is a positive definite kernel by construction and we have,

$$\begin{aligned} k_\Gamma(i,j) &= \sum_{d=0}^{\min(i,j)-1} C_{\mathrm{hv}}^{i+j-2-2d} C_{\mathrm{d}}^{d} \frac{(i+j-2-d)!}{(i-1-d)!(j-1-d)!d!} \\ &= \sum_{d=0}^{\min(i,j)-1} k_d(i,j) = \sum_{d=0}^{\min(i,j)-1} \delta_{d<i} k_d(i,j) \delta_{d<j} \\ &= \sum_{d=0}^{\min(i,j)-1} \hat{k}_d(i,j) = \sum_{d=0}^{\infty} \hat{k}_d(i,j), \end{aligned}$$

where $\delta_{d<i} = 1$ if $d < i$ and zero otherwise. For any finite set of integers, only finitely many terms in the sum above are non-zero and the kernel k_S is clearly positive since it is a sum of positive kernels.

Corollary 4.3 *Let $k_\Sigma : \Sigma \times \Sigma \to \mathbb{R}$ be a continuous positive definite kernel on Σ, where Σ is a separable metric space. Then the associated path kernel $k_p : Seq(\Sigma) \times Seq(\Sigma) \to \mathbb{R}$ is a positive definite kernel for any $C_d \geqslant 0$ and $C_{hv} \in \mathbb{R}$.*

5. Experiments

In this section we will experimentally evaluate the performance of the path sequence kernel on a set of real sequential classification data-sets. However, to provide intution for the approach, we will first show how the proposed path sequence kernel represent two sets of toy-data. One of the motivations behind this work is to provide a vectorial embedding of sequences of different length. To evaluate this we, generate 10 noisy sine and cosine curves of different length as shown in Figure 3. We now wish to find an embedding that separates the two classes of curves. By formulating the classification task as a regression problem to a $1-C$ encoding and placing a Gaussian Process prior (Rasmussen and Williams, 2006) over the mapping, we can learn the kernel parameters through a maximum-likelihood approach. In Figure 3, the resulting embedding is displayed, clearly showing how the kernel manages to separate the two classes. It is important to note that both the sine and the cosine curves are generated over a full period which means that the first order statistics for the curves are the same. The discriminating information in the data is hence contained in the sequential structure which the Path sequence kernel extracts.

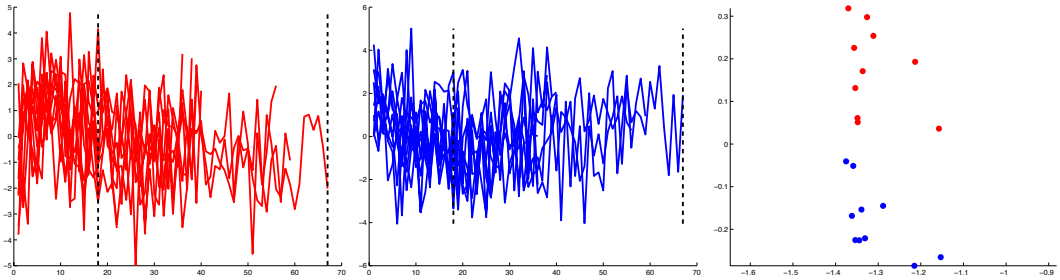

Figure 3: *The above figure depicts the the first toy data-set onto which we have applied the path sequence kernel. The left-most image shows the first class which consists of* 10 *noisy sine curves of different length while the middle image shows the other class which consists of noisy cosine curves. The dashed line indicate the length of the longest and the shortest curve in each data-set respectively. The right-most image shows the embedding defined by the first two principal components given by the trained kernel.*

One of the benefits of the proposed decomposable kernel is that, by learning its parameters, we can determine if the discriminating information is contained in the structure or symbol level of the sequences. To evaluate this we generated a second toy data set Figure 4. The data-set consists of 10 noisy sine and square waves. Half of the sequences from each class have been altered such that 5 symbols at random places take the value 4. We will conduct two experiments on this data. In the first we will learn the parameters of the kernel as to separate the sine from the square waveform, while for the second experiment, we want to separate the sequences containing the randomly positioned symbol 4 irrespective of waveform. In the first experiment the structure of the symbols are much more important while in the second the only distinguishing aspect is that the sequence contains the symbol 4. This is reflected by the learned parameters, for the first experiment the coefficent for making diagonal moves, C_D, which reflects the importance of the sequences being aligned, is much higher compared to the second experiment where there is little difference between diagonal and horizontal move reflecting that the information is contained at the symbol level of the sequences. In Figure 4 the embedding and the kernel matrices are shown.

We will now proceed to evaluate the performance of the different kernels on a set of well known sequential classification data-sets from the UCI Machine Learning repository (Bache and Lichman, 2013) with varying length, dimension and number of classes, see Table 1. We compared three different kernels: the exponential, the path sequence kernels and the global alignment kernelCuturi et al. (2007). In each of the experiments the same exponential kernel is used to represent the symbol space such that the kernels only differ in how they represent the structural component of each sequence. Classification is performed by applying a support vector machine Chang and Lin (2011) to the space induced by the various kernel. The parameters of the kernels and the classifier are learned using nested-cross validation with 3 outer and inner folds and 3 and 20 repetitions respectively. The outer cross-validation iterates through different divisions of the data into training and test

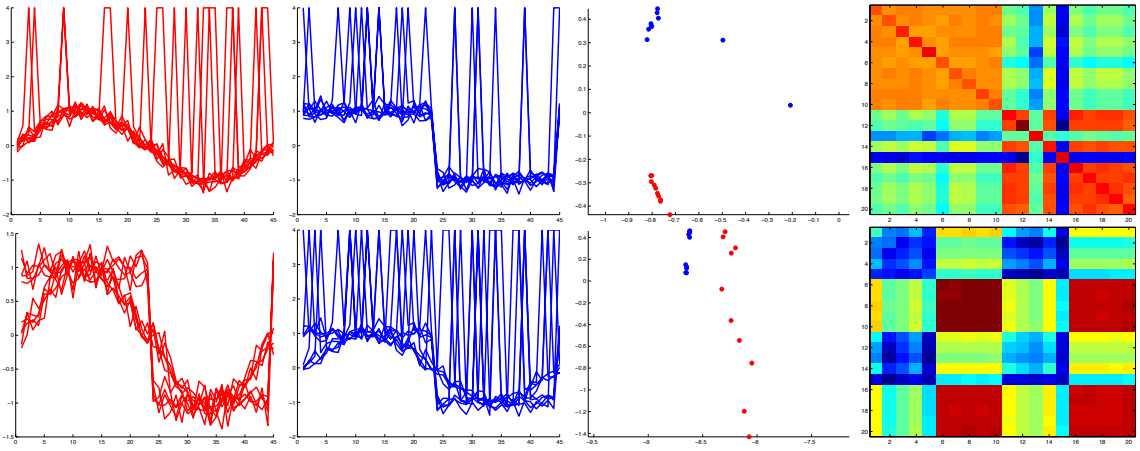

Figure 4: *The figure shows the second toy data-set presented in the paper. Each row corresponds to a specific experiment setting. The first two columns show the two input classes while the third shows the embedding defined by the two first principal components and the last column depicts the learned kernel matrix evaluated on the input data.*

sets. The inner cross-validation uses the training set to perform parameter selection, using the established number of folds and repetitions. The chosen parameters are finally used to test the model on the test set and the results are measured and averaged over the previously described number of folds and repetitions.

To make sure that the discriminative information in the data resides in the structure and not only in the first order statistics of the symbol space we tried to classify the fixed length data by assuming each dimension to be independent. To do so we concatenated each symbol in a sequence and used an exponential kernel and a Euclidean distance as a similarity measure. The results are shown in Table 6. As can be seen, the performance for the Libras and the different PEMS data-sets are roughly random indicating that the structure is indeed important. Comparing the three different kernels, we can see that the path sequence kernel is consistently outperforming the other two kernels. It is interesting to see the significant difference in performance between the exponential and the path sequence kernels. We argue that the difference in performance is due to the stationary characteristics of the exponential kernel where the influence of each symbol match only depends on the difference in position. The path kernel has a more fine-grained characteristic where the actual position of a match incluences the similarity score. Both the path and the global alignment kernel take all possible alignments into account. As we see, the path kernel significantly outperforms the global alignment kernel. This can be explained by the dominating influence of the "best" alignment in the global alignment kernel compared to the path kernel which more gracefully accumulates information from all possible alignments into the final kernel value. This behavior also means that there is a strong preference towards sequences of the same length for the global alignment kernel which is something that can explain the big difference

	dim	length	#classes	#N
AUSLAN	22	45-136 (55)	95	2565
Libras	2	45	15	945
PEMS100	963	144	7	440
PEMS95	335	144	7	440
PEMS90	171	144	7	440
Vowels	12	7-29 (15)	9	640
Characters	3	60-182 (122)	20	2858

Table 1: *The above table charaterizes the UCI (Bache and Lichman, 2013) data-sets on which our experimental evaluation is performed. From top to bottom the data sets where presented in (Kadous, 2002; Dias et al., 2009; Cuturi, 2010; Kudo et al., 1999; Williams et al., 2006). The column from left to right show the dimensionality of the symbol space, the range of lengths of the sequences and their median length within brackets, number of classes and the number of sequences. The three different PEMS data-sets are projections of the data onto its principal components such that 100,95 and 90 percent of the variance in the data is retained.*

in performance compared with the path kernel on the AUSLAN data-set. We believe this shows the value of a non-stationary structured kernel for representing sequences. The path perspective provides an intuitive, rigorous and interpretable formulation for designing such new kernels.

6. Conclusion

In this paper we have presented an approach to combine kernels in a structured manner such that the resulting measurement represent a Mercer kernel. This leads to kernels that provides a (dis)similarity measure between sequences of different length. In particular we proved that a recently proposed (dis)similarity measure (Baisero et al., 2013) falls within this family which allows us to adopt the intuitive notion of alignments which we believe will be provide useful insights for designing new kernels. We showed experimentally how the path sequence kernel significantly outperforms previous state-of-the-art methods. In future work we aim to further establish the path perspective and discuss how new novel kernels which include higher-order paths which takes more than simple moves into account can be designed.

References

K Bache and M Lichman. UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2013.

C Bahlmann, B Haasdonk, and H Burkhardt. Online handwriting recognition with support vector machines - a kernel approach. In *8th International Workshop on Frontiers in*

	AUSLAN	Libras	PEMS100	PEMS95	PEMS90	Vowels	Characters
RBF	-	$6.67 \pm 0.41\%$	$14.13 \pm 1.45\%$	$14.74 \pm 1.42\%$	$14.11 \pm 1.29\%$	-	-
k_{ga}	$68.08 \pm 8.40\%$	$84.18 \pm 3.99\%$	$39.16 \pm 17.10\%$	$39.27 \pm 19.45\%$	$36.44 \pm 19.44\%$	$96.51 \pm 1.67\%$	$98.40 \pm 0.11\%$
k_e	$67.06 \pm 1.99\%$	$43.55 \pm 3.83\%$	$15.65 \pm 6.17\%$	$12.73 \pm 0.9\%$	$12.58 \pm 1.10\%$	$96.87 \pm 0.78\%$	-
k_p	$92.40 \pm 0.63\%$	$87.37 \pm 2.86\%$	$61.32 \pm 15.17\%$	$60.26 \pm 16.19\%$	$54.05 \pm 17.17\%$	$97.97 \pm 0.55\%$	$98.17 \pm 0.70\%$

Table 2: *The above table shows the average classification performance and the standard deviation over the multiple runs. The symbols k_{ga},k_e and k_p refers to the global alignment, exponential sequence and the path sequence kernel respectively.*

Handwriting Recognition, pages 49–54, August 2002.

Andrea Baisero, Florian T Pokorny, Danica Kragic, and Carl Henrik Ek. The Path Kernel. In *International Conference on Pattern Recognition Applications and Methods*, February 2013.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

Corinna Cortes, Patrick Haffner, and Mehryar Mohri. Rational kernels: Theory and algorithms. *The Journal of Machine Learning Research*, 5:1035–1062, 2004.

Marco Cuturi. Fast Global Alignment Kernels. In *International Conference on Machine Learning*, 2010.

Marco Cuturi, Jean-Philippe Vert, Oystein Birkenes, and Tomoko Matsui. A Kernel for Time Series Based on Global Alighments. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages II–413–II–416. IEEE, 2007.

Fred J Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, March 1964.

D B Dias, R C B Madeo, T Rocha, H H Biscaro, and S M Peres. Hand movement recognition for Brazilian Sign Language: A study using distance-based neural networks. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 697–704, 2009.

B Haasdonk. Feature space interpretation of SVMs with indefinite kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):482–492, 2005.

Zelig Harris. Distributional structure. In *Papers in Structural and Transformational Linguistics*, pages 775–794. D. Reidel Publishing Company, Dordrecht, Holland, 1970.

David Haussler. Convolution kernels on discrete structures. Technical report, University of California at Santa Cruz, 1999.

M W Kadous. *Temporal Classification: Extending the Classification Paradigm to Multivariate Time Series*. PhD thesis, University of New South Wales, 2002.

Ashish Kapoor, Kirsten Grauman, Raquel Urtasun, and Trevor Darrell. Gaussian Processes for Object Categorization. *International journal of computer vision*, 88(2):169–188, July 2009.

Mineichi Kudo, Jun Toyama, and Masaru Shimbo. Multidimensional curve classification using passing-through regions. *Pattern Recognition Letters*, 20(11-13):1103–1111, November 1999.

Hansheng Lei and Bingyu Sun. A Study on the Dynamic Time Warping in Kernel Machines. In *Third International IEEE Conference on Signal-Image Technologies and Internet-Based System SITIS*, pages 839–845. IEEE, 2007.

V I Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet physics doklady*, 10:707, February 1966.

H Lodhi, C Saunders, J Shawe-Taylor, N Cristianini, and C Watkins. Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444, 2002.

Mehryar Mohri. Weighted Automata Algorithms. In *Handbook of weighted automata*, pages 213–254. Springer Berlin Heidelberg, Berlin, Heidelberg, September 2009.

Meinard Müller. Dynamic Time Warping. In *Information retrieval for music and motion*, pages 69–84. Springer Berlin Heidelberg, 2007.

Carl Edward Rasmussen and Christopher K I Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2006.

Konrad Rieck. Similarity measures for sequential data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(4):296–304, 2011.

J Rousu and J Shawe-Taylor. Efficient Computation of gapped substring kernels for large alphabets. *Journal of Machine Learning Research*, 2006.

H Sakoe and S Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(1):43–49, 1978.

Eiji Takimoto and Manfred K Warmuth. Path kernels and multiplicative updates. *The Journal of Machine Learning Research*, 4:773–818, 2003.

Ben H Williams, Marc Toussaint, and Amos J Storkey. Extracting Motion Primitives from Natural Handwriting Data. In *Artificial Neural Networks*, pages 634–643, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.