D2KE: From Distance to Kernel and Embedding via Random Features For Structured Inputs

Anonymous authors
Paper under double-blind review

Abstract

We present a new methodology that constructs a family of *positive definite kernels* from any given dissimilarity measure on structured inputs whose elements are either real-valued time series or discrete structures such as strings, histograms, and graphs. Our approach, which we call D2KE (from Distance to Kernel and Embedding), draws from the literature of Random Features. However, instead of deriving random feature maps from a user-defined kernel to approximate kernel machines, we build a kernel from a random feature map, that we specify given the distance measure. We further propose use of a finite number of random objects to produce a random feature embedding of each instance. We provide a theoretical analysis showing that D2KE enjoys better generalizability than universal Nearest-Neighbor estimates. On one hand, D2KE subsumes the widely-used *representative-set method* as a special case, and relates to the well-known *distance substitution kernel* in a limiting case. On the other hand, D2KE generalizes existing *Random Features methods* applicable only to vector input representations to complex structured inputs of variable sizes. We conduct classification experiments over such disparate domains as time series, strings, and histograms (for texts and images), for which our proposed framework compares favorably to existing distance-based learning methods in terms of both testing accuracy and computational time.

1 Introduction

In many problem domains, it is easier to specify a reasonable dissimilarity (or similarity) function between instances than to construct a feature representation. This is particularly the case with structured inputs whose elements are either real-valued time series or discrete structures such as strings, histograms, and graphs, where it is typically less than clear how to construct the representation of entire structured inputs with potentially widely varying sizes, even when given a good feature representation of each individual component. Moreover, even for complex structured inputs, there are many well-developed dissimilarity measures, such as the Dynamic Time Warping measure between time series, Edit Distance between strings, Hausdorff distance between sets, and Wasserstein distance between distributions.

However, standard machine learning methods are designed for vector representations, and classically there has been far less work on distance-based methods for either classification or regression on structured inputs. The most common distance-based method is Nearest-Neighbor Estimation (NNE), which predicts the outcome for an instance using an average of its nearest neighbors in the input space, with nearness measured by the given dissimilarity measure. Estimation from nearest neighbors, however, is unreliable, specifically having high variance when the neighbors are far apart, which is typically the case when the intrinsic dimension implied by the distance is large.

To address this issue, a line of research has focused on developing global distance-based (or similarity-based) machine learning methods (Pkkalska & Duin, 2005; Duin & Pękalska, 2012; Balcan et al., 2008a; Cortes et al., 2012), in large part by drawing upon connections to kernel methods (Scholkopf et al., 1999) or directly learning with similarity functions (Balcan et al., 2008a; Cortes et al., 2012; Balcan et al., 2008b; Loosli et al., 2016); we refer the reader in particular to the survey in (Chen et al., 2009a). Among these, the most direct approach treats the data similarity matrix (or transformed dissimilarity matrix) as a kernel Gram matrix, and then uses standard kernel-based methods such as

Support Vector Machines (SVM) or kernel ridge regression with this Gram matrix. A key caveat with this approach however is that most similarity (or dissimilarity) measures do not provide a *positive-definite (PD)* kernel, so that the empirical risk minimization problem is not well-defined, and moreover becomes non-convex (Ong et al., 2004; Lin & Lin, 2003).

A line of work has therefore focused on estimating a positive-definite (PD) Gram matrix that merely approximates the similarity matrix. This could be achieved for instance by clipping, or flipping, or shifting eigenvalues of the similarity matrix (Pekalska et al., 2001), or explicitly learning a PD approximation of the similarity matrix (Chen & Ye, 2008; Chen et al., 2009b). Such modifications of the similarity matrix however often leads to a loss of information; moreover, the enforced PD property is typically guaranteed to hold only on the training data, resulting in an inconsistency between the set of testing and training samples (Chen et al., 2009a) [1].

Another common approach is to select a subset of training samples as a held-out representative set, and use distances or similarities to structured inputs in the set as the feature function (Graepel et al., 1999; Pekalska et al., 2001). As we will show, with proper scaling, this approach can be interpreted as a special instance of our framework. Furthermore, our framework provides a more general and richer family of kernels, many of which significantly outperform the representative-set method in a variety of application domains.

To address the aforementioned issues, in this paper, we propose a novel general framework that constructs a family of PD kernels from a dissimilarity measure on structured inputs. Our approach, which we call D2KE (from Distance to Kernel and Embedding), draws from the literature of Random Features (Rahimi & Recht, 2008), but instead of deriving feature maps from an existing kernel for approximating kernel machines, we build novel kernels from a random feature map specifically designed for a given distance measure. The kernel satisfies the property that functions in the corresponding Reproducing Kernel Hilbert Space (RKHS) are Lipschitz-continuous w.r.t. the given distance measure. We also provide a tractable estimator for a function from this RKHS which enjoys much better generalization properties than nearest-neighbor estimation. Our framework produces a feature embedding and consequently a vector representation of each instance that can be employed by any classification and regression models. In classification experiments in such disparate domains as strings, time series, and histograms (for texts and images), our proposed framework compares favorably to existing distance-based learning methods in terms of both testing accuracy and computational time, especially when the number of data samples is large and/or the size of structured inputs is large.

We highlight our main contributions as follows:

- From the perspective of distance kernel learning, we propose for the first time a methodology that constructs a family of PD kernels via Random Features from a given distance measure for structured inputs, and provide theoretical and empirical justifications for this framework.
- From the perspective of Random Features (RF) methods, we generalize existing Random Features methods applied only to vector input representations to complex structured inputs of variable sizes. To the best of our knowledge, this is the first time that a generic RF method has been used to accelerate kernel machines on structured inputs across a broad range of domains such as time-series, strings, and the histograms.

2 Related Work

Distance-Based Kernel Learning. Existing approaches either require strict conditions on the distance function (e.g. that the distance be isometric to the square of the Euclidean distance) (Haasdonk & Bahlmann, 2004; Schölkopf, 2001), or construct empirical PD Gram matrices that do not necessarily generalize to the test samples (Pekalska et al., 2001; Pkkalska & Duin, 2005; Pekalska & Duin, 2006; 2008; Duin & Pękalska, 2012). Haasdonk & Bahlmann (2004) and Schölkopf (2001) provide conditions under which one can obtain a PD kernel through simple transformations of the distance measure, but which are not satisfied for many commonly used dissimilarity measures such as Dynamic Time Warping, Hausdorff distance, and Earth Mover's distance (Haasdonk & Bahlmann,

[1] A generalization error bound was provided for the *similarity-as-kernel* approach in (Chen et al., 2009a), but only for a *positive-definite* similarity function.

Table 1: Comparison between D2KE and different random features methods.

Methods	Inputs format	Distance Metric	Random Feature	Build new kernel
D2KE	Time-series, strings, sets	DTW, EditDist, HD	From any distribution	Yes
Rahimi's RF and its variants	Vector-form	Euclidean	User defined	No

2004). Equivalently, one could also find a Euclidean embedding (also known as dissimilarity representation) approximating the dissimilarity matrix as in Multidimensional Scaling (Pekalska et al., 2001; Pkkalska & Duin, 2005; Pekalska & Duin, 2006; 2008; Duin & Pękalska, 2012) [2]. Differently, Loosli et al. (2016) presented a theoretical foundation for an SVM solver in Krein spaces and directly evaluated a solution that uses the original (indefinite) similarity measure.

There are also some specific approaches dedicated to building a PD kernel on some structured inputs such as text and time-series (Collins & Duffy, 2002; Cuturi, 2011), that modify a distance function over sequences to a kernel by replacing the minimization over possible alignments into a summation over all possible alignments. This type of kernel, however, results in a diagonal-dominance problem, where the diagonal entries of the kernel Gram matrix are orders of magnitude larger than the off-diagonal entries, due to the summation over a huge number of alignments with a sample itself.

Random Features Methods. Interest in approximating non-linear kernel machines using randomized feature maps has surged in recent years due to a significant reduction in training and testing times for kernel based learning algorithms (Dai et al., 2014). There are numerous explicit nonlinear random feature maps that have been constructed for various types of kernels, including Gaussian and Laplacian Kernels (Rahimi & Recht, 2008; Wu et al., 2016), intersection kernels (Maji & Berg, 2009), additive kernels Vedaldi & Zisserman (2012), dot product kernels (Kar & Karnick, 2012; Pennington et al., 2015), and semigroup kernels (Mukuta et al., 2018). Among them, the Random Fourier Features (RFF) method, which approximates a Gaussian Kernel function by means of multiplying the input with a Gaussian random matrix, and its fruitful variants have been extensively studied both theoretically and empirically (Sriperumbudur & Szabó, 2015; Felix et al., 2016; Rudi & Rosasco, 2017; Bach, 2017; Choromanski et al., 2018). To accelerate the RFF on input data matrix with high dimensions, a number of methods have been proposed to leverage structured matrices to allow faster matrix computation and less memory consumption (Le et al., 2013; Hamid et al., 2014; Choromanski & Sindhwani, 2016).

However, all the aforementioned RF methods merely consider inputs with vector representations, and compute the RF by a linear transformation that is either a matrix multiplication or an inner product under Euclidean distance metric. In contrast, D2KE takes structured inputs of potentially different sizes and computes the RF with a structured distance metric (typically with dynamic programming or optimal transportation). Another important difference between D2KE and existing RF methods lies in the fact that existing RF work assumes a user-defined kernel and then derives a random-feature map, while D2KE constructs a new PD kernel through a random feature map and makes it computationally feasible via RF. The table 1 lists the differences between D2KE and existing RF methods.

A very recent piece of work (Wu et al., 2018) has developed a kernel and a specific algorithm for computing embeddings of single-variable real-valued time-series. However, despite promising results, this method cannot be applied on discrete structured inputs such as strings, histograms, and graphs. In contrast, we have an unified framework for various structured inputs beyond the limits of (Wu et al., 2018) and provide a general theoretical analysis w.r.t KNN and other generic distance-based kernel methods.

3 Problem Setup

We consider the estimation of a target function $f : \mathcal{X} \to \mathbb{R}$ from a collection of samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathcal{X}$ is the structured input object, and $y_i \in \mathcal{Y}$ is the output observation associated with the target function $f(\boldsymbol{x}_i)$. For instance, in a regression problem, $y_i \sim f(\boldsymbol{x}_i) + \omega_i \in \mathbb{R}$ for some random noise ω_i, and in binary classification, we have $y_i \in \{0, 1\}$ with $P(y_i = 1|\boldsymbol{x}_i) = f(\boldsymbol{x}_i)$. We are given a dissimilarity measure $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ between input objects instead of a feature representation of $\boldsymbol{x}$.

[2] A proof of the equivalence between PD of similarity matrix and Euclidean of dissimilarity matrix can be found in (Borg & Groenen, 1997).

Note that the size structured inputs $\boldsymbol{x}_i$ may vary widely, e.g. strings with variable lengths or graphs with different sizes. For some of the analyses, we require the dissimilarity measure to be a *metric* as follows.

Assumption 1 (Distance Metric)**.** *$d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a distance metric, that is, it satisfies (i) $d(\boldsymbol{x}_1, \boldsymbol{x}_2) \geq 0$, (ii) $d(\boldsymbol{x}_1, \boldsymbol{x}_2) = 0 \iff \boldsymbol{x}_1 = \boldsymbol{x}_2$, (iii) $d(\boldsymbol{x}_1, \boldsymbol{x}_2) = d(\boldsymbol{x}_2, \boldsymbol{x}_1)$, and (iv) $d(\boldsymbol{x}_1, \boldsymbol{x}_2) \leq d(\boldsymbol{x}_1, \boldsymbol{x}_3) + d(\boldsymbol{x}_3, \boldsymbol{x}_2)$.*

3.1 Function Continuity and Space Covering

An ideal feature representation for the learning task is (i) compact and (ii) such that the target function $f(\boldsymbol{x})$ is a simple (e.g. linear) function of the resulting representation. Similarly, an ideal dissimilarity measure $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$ for learning a target function $f(\boldsymbol{x})$ should satisfy certain properties. On one hand, a small dissimilarity $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$ between two objects should imply small difference in the function values $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)|$. On the other hand, we want a small expected distance among samples, so that the data lies in a compact space of small intrinsic dimension. We next build up some definitions to formalize these properties.

Assumption 2 (Lipschitz Continuity)**.** *For any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}$, there exists some constant $L > 0$ such that*

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq L\, d(\boldsymbol{x}_1, \boldsymbol{x}_2), \tag{1}$$

We would prefer the target function to have a small Lipschitz-continuity constant L with respect to the dissimilarity measure $d(.,.)$. Such Lipschitz-continuity alone however might not suffice. For example, one can simply set $d(\boldsymbol{x}_1, \boldsymbol{x}_2) = \infty$ for any $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$ to satisfy Eq. equation 1. We thus need the following quantity that measures the size of the space implied by a given dissimilarity measure.

Definition 1 (Covering Number)**.** *Assuming d is a* metric*. A δ-cover of $\mathcal{X}$ w.r.t. $d(.,.)$ is a set $\mathcal{E}$ s.t.*

$$\forall \boldsymbol{x} \in \mathcal{X}, \exists \boldsymbol{x}_i \in \mathcal{E}, d(\boldsymbol{x}, \boldsymbol{x}_i) \leq \delta.$$

Then the covering number $N(\delta; \mathcal{X}, d)$ is the size of the smallest δ-cover for $\mathcal{X}$with respect to d.

Assuming the input domain $\mathcal{X}$ is compact, the covering number $N(\delta; \mathcal{X}, d)$ measures its size w.r.t. the distance measure d. We show how the two quantities defined above affect the estimation error of a Nearest-Neighbor Estimator.

3.2 Effective Dimension and Nearest Neighbor Estimation

We extend the standard analysis of the estimation error of k-nearest-neighbor from finite-dimensional vector spaces to any structured input space $\mathcal{X}$, with an associated distance measure d, and a finite covering number $N(\delta; \mathcal{X}, d)$, by defining the *effective dimension* as follows.

Assumption 3 (Effective Dimension)**.** *Let the effective dimension $p_{\mathcal{X},d} > 0$ be the minimum p satisfying*

$$\exists c > 0, \forall \delta : 0 < \delta < 1, \quad N(\delta; \mathcal{X}, d) \leq c \left(\frac{1}{\delta}\right)^p.$$

Here we provide an example of effective dimension in case of measuring the space of *Multiset*.

Multiset with Hausdorff Distance. A multiset is a set that allows duplicate elements. Consider two multisets $\boldsymbol{x}_1 = \{\boldsymbol{u}_i\}_{i=1}^M$, $\boldsymbol{x}_2 = \{\boldsymbol{v}_j\}_{j=1}^N$. Let $\Delta(\boldsymbol{u}_i, \boldsymbol{v}_j)$ be a *ground distance* that measures the distance between two elements $\boldsymbol{u}_i, \boldsymbol{v}_j \in \mathcal{V}$ in a set. The (modified) *Hausdorff Distance* (Dubuisson & Jain, 1994) can be defined as $d(\boldsymbol{x}_1, \boldsymbol{x}_2) :=$

$$\max\{\frac{1}{N}\sum_{i=1}^{N} \min_{j \in [M]} \Delta(\boldsymbol{u}_i, \boldsymbol{v}_j), \frac{1}{M}\sum_{j=1}^{M} \min_{i \in [N]} \Delta(\boldsymbol{v}_j, \boldsymbol{u}_i)\} \tag{2}$$

Let $N(\delta; \mathcal{V}, \Delta)$ be the covering number of $\mathcal{V}$ under the ground distance Δ. Let $\mathcal{X}$ denote the set of all sets of size bounded by L. By constructing a covering of $\mathcal{X}$ containing any set of size less or equal than L with its elements taken from the covering of $\mathcal{V}$, we have $N(\delta; \mathcal{X}, d) \leq N(\delta; \mathcal{V}; \Delta)^L$. Therefore, $p_{\mathcal{X},d} \leq L \log N(\delta; \mathcal{V}, \Delta)$. For example, if $\mathcal{V} := \{\boldsymbol{v} \in \mathbb{R}^p \mid \|\boldsymbol{v}\|_2 \leq 1\}$ and Δ is Euclidean distance, we have $N(\delta; \mathcal{V}, \Delta) = (1 + \frac{2}{\delta})^p$ and $p_{\mathcal{X},d} \leq Lp$.

Equipped with the concept of *effective dimension*, we can obtain the following bound on the estimation error of the k-Nearest-Neighbor estimate of $f(\boldsymbol{x})$.

Theorem 1. *Let $Var(y|f(x)) \leq \sigma^2$, and $\hat{f}_n$ be the k-Nearest Neighbor estimate of the target function f constructed from a training set of size n. Denote $p := p_{\mathcal{X},d}$. We have*

$$\mathbb{E}_{\boldsymbol{x}}\left[\left(\hat{f}_n(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2\right] \leq \frac{\sigma^2}{k} + cL^2\left(\frac{k}{n}\right)^{2/p}$$

for some constant $c > 0$. For $\sigma > 0$, minimizing RHS w.r.t. the parameter k, we have

$$\mathbb{E}_{\boldsymbol{x}}\left[\left(\hat{f}_n(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2\right] \leq c_2\sigma^{\frac{4}{p+2}}L^{\frac{2p}{2+p}}\left(\frac{1}{n}\right)^{\frac{2}{2+p}} \tag{3}$$

for some constant $c_2 > 0$.

Proof. The proof is almost the same to a standard analysis of k-NN's estimation error in, for example, (Györfi et al., 2006), with the *space partition number* replaced by the *covering number*, and *dimension* replaced by the *effective dimension* in Assumption 3. □

When $p_{\mathcal{X},d}$ is reasonably large, the estimation error of k-NN decreases quite slowly with n. Thus, for the estimation error to be bounded by ϵ, requires the number of samples to scale exponentially in $p_{\mathcal{X},d}$. In the following sections, we develop an estimator $\hat{f}$ based on a RKHS derived from the distance measure, with a considerably better sample complexity for problems with higher effective dimension.

4 From Distance to Kernel for Structured Inputs

We aim to address the long-standing problem of how to convert a distance measure into a positive-definite kernel. Here we introduce a simple but effective approach *D2KE* that constructs a family of *positive-definite* kernels from a given distance measure. Given an structured input domain $\mathcal{X}$ and a distance measure $d(.,.)$, we construct a family of kernels as

$$k(\boldsymbol{x},\boldsymbol{y}) := \int p(\boldsymbol{\omega})\phi_{\boldsymbol{\omega}}(\boldsymbol{x})\phi_{\boldsymbol{\omega}}(\boldsymbol{y})d\boldsymbol{\omega}, \text{where } \ \phi_{\boldsymbol{\omega}}(\boldsymbol{x}) := \exp(-\gamma d(\boldsymbol{x},\boldsymbol{\omega})), \tag{4}$$

where $\boldsymbol{\omega} \in \Omega$ is a random structured object whose elements could be real-valued time-series, strings, and histograms, $p(\boldsymbol{\omega})$ is a distribution over Ω, and $\phi_{\boldsymbol{\omega}}(\boldsymbol{x})$ is a feature map derived from the distance of $\boldsymbol{x}$ to all random objects $\boldsymbol{\omega} \in \Omega$. The kernel is parameterized by both $p(\boldsymbol{\omega})$ and γ.

Relationship to Distance Substitution Kernel. An insightful interpretation of the kernel in Equation (4) can be obtained by expressing the kernel in Equation (4) as

$$\exp\left(-\gamma softmin_{p(\boldsymbol{\omega})}\{d(\boldsymbol{x},\boldsymbol{\omega}) + d(\boldsymbol{\omega},\boldsymbol{y})\}\right) \tag{5}$$

where the soft minimum function, parameterized by $p(\boldsymbol{\omega})$ and γ, is defined as

$$softmin_{p(\boldsymbol{\omega})}\ f(\boldsymbol{\omega}) := -\frac{1}{\gamma}\log\int p(\boldsymbol{\omega})e^{-\gamma f(\boldsymbol{\omega})}d\boldsymbol{\omega}. \tag{6}$$

Therefore, the kernel $k(\boldsymbol{x},\boldsymbol{y})$ can be interpreted as a soft version of the *distance substitution kernel* (Haasdonk & Bahlmann, 2004), where instead of substituting $d(\boldsymbol{x},\boldsymbol{y})$ into the exponent, it substitutes a soft version of the form

$$softmin_{p(\boldsymbol{\omega})}\{d(\boldsymbol{x},\boldsymbol{\omega}) + d(\boldsymbol{\omega},\boldsymbol{y})\}. \tag{7}$$

Note when $\gamma \to \infty$, the value of Equation (7) is determined by $\min_{\boldsymbol{\omega}\in\Omega}\ d(\boldsymbol{x},\boldsymbol{\omega}) + d(\boldsymbol{\omega},\boldsymbol{y})$, which equals $d(\boldsymbol{x},\boldsymbol{y})$ if $\mathcal{X} \subseteq \Omega$, since it cannot be smaller than $d(\boldsymbol{x},\boldsymbol{y})$ by the triangle inequality. In other words, when $\mathcal{X} \subseteq \Omega$,

$$k(\boldsymbol{x},\boldsymbol{y}) \to \exp(-\gamma d(\boldsymbol{x},\boldsymbol{y})) \ \text{ as } \ \gamma \to \infty.$$

On the other hand, unlike the distance-substituion kernel, our kernel in Equation (5) is always PD by construction.

Algorithm 1 Random Feature Approximation of function in RKHS with the kernel in Equation 4

1: Draw R samples from $p(\boldsymbol{\omega})$ to get $\{\boldsymbol{\omega}_j\}_{j=1}^{R}$.

2: Set the R-dimensional feature embedding as

$$\hat{\boldsymbol{\phi}}_j(\boldsymbol{x}) = \frac{1}{\sqrt{R}} \exp(-\gamma d(\boldsymbol{x}, \boldsymbol{\omega}_j)), \ \forall j \in [R]$$

3: Solve the following problem for some $\mu > 0$:

$$\hat{\boldsymbol{w}} := \underset{\boldsymbol{w} \in \mathbb{R}^R}{argmin} \ \frac{1}{n} \sum_{i=1}^{n} \ell(\boldsymbol{w}^T \hat{\boldsymbol{\phi}}(\boldsymbol{x}_i), y_i) + \frac{\mu}{2} \|\boldsymbol{w}\|^2$$

4: Output the estimated function $\tilde{f}_R(\boldsymbol{x}) := \hat{\boldsymbol{w}}^T \hat{\boldsymbol{\phi}}(\boldsymbol{x})$.

Random Feature Approximation. The reader might have noticed that the kernel in Equation (4) cannot be evaluated analytically in general. However, this does not prohibit its use in practice, so long as we can approximate it via *Random Features (RF)* (Rahimi & Recht, 2008), which in our case is particularly natural as the kernel itself is defined via a random feature map. Thus, our kernel with the RF approximation can not only be used in small problems but also in large-scale settings with a large number of samples, where standard kernel methods with $O(n^2)$ complexity are no longer efficient enough and approximation methods, such as Random Features, must be employed (Rahimi & Recht, 2008). Given the RF approximation, one can then directly learn a target function as a linear function of the RF feature map, by minimizing a domain-specific empirical risk. It is worth noting that a recent work (Sinha & Duchi, 2016) that learns to select a set of random features by solving an optimization problem in an supervised setting is orthogonal to our D2KE approach and could be extended to develop a supervised D2KE method. We outline this overall *RF* based empirical risk minimization for our class of D2KE kernels in Algorithm 1. It is worth pointing out that in line 2 of Algorithm 1 the random feature embeddings are computed by a structured distance measure between the original structured inputs and the generated random structured inputs, followed by the application of the exponent function parameterized by γ. This is in contrast with traditional RF methods that translate the input data matrix into the embedding matrix via a matrix multiplication with random Gaussian matrix followed by a non-linearity. We will provide a detailed analysis of our estimator in Algorithm 1 in Section 5, and contrast its statistical performance to that of K-nearest-neighbor.

Relationship to Representative-Set Method. A naive choice of $p(\boldsymbol{\omega})$ relates our approach to the *representative-set method (RSM)*: setting $\Omega = \mathcal{X}$, with $p(\boldsymbol{\omega}) = p(\boldsymbol{x})$. This gives us a kernel Equation (4) that depends on the data distribution. One can then obtain a Random-Feature approximation to the kernel in Equation (4) by holding out a part of the training data $\{\hat{\boldsymbol{x}}_j\}_{j=1}^{R}$ as samples from $p(\boldsymbol{\omega})$, and creating an R-dimensional feature embedding of the form:

$$\hat{\phi}_j(\boldsymbol{x}) := \frac{1}{\sqrt{R}} \exp\left(-\gamma d(\boldsymbol{x}, \hat{\boldsymbol{x}}_j)\right), \ j \in [R], \tag{8}$$

as in Algorithm 1. This is equivalent to a $1/\sqrt{R}$-scaled version of the embedding function in the *representative-set method* (or *similarity-as-features method*) (Graepel et al., 1999; Pekalska et al., 2001; Pkkalska & Duin, 2005; Pekalska & Duin, 2006; 2008; Chen et al., 2009a; Duin & Pękalska, 2012), where one computes each sample's similarity to a set of representatives as its feature representation. However, here by interpreting Equation (8) as a random-feature approximation to the kernel in Equation (4), we obtain a much nicer generalization error bound even in the case $R \to \infty$. This is in contrast to the analysis of RSM in (Chen et al., 2009a), where one has to keep the size of the representative set small (of the order $O(n)$) in order to have reasonable generalization performance.

Effect of $p(\boldsymbol{\omega})$. The choice of $p(\boldsymbol{\omega})$ plays an important role in our kernel. Surprisingly, we found that many "close to uniform" choices of $p(\boldsymbol{\omega})$ in a variety of domains give better performance than for instance the choice of the data distribution $p(\boldsymbol{\omega}) = p(\boldsymbol{x})$ (as in the representative-set method). Here are some examples from our experiments: i) In the *time-series* domain with dissimilarity computed via Dynamic Time Warping (DTW), a distribution $p(\boldsymbol{\omega})$ corresponding to random time series of length uniform in $\in [2, 10]$, and with Gaussian-distributed elements, yields much better performance than

the Representative-Set Method (RSM); ii) In *string* classification, with edit distance, a distribution $p(\boldsymbol{\omega})$ corresponding to random strings with elements uniformly drawn from the alphabet Σ yields much better performance than RSM; iii) When classifying sets of vectors with the Hausdorff distance in Equation (2), a distribution $p(\boldsymbol{\omega})$ corresponding to random sets of size uniform in $\in [3, 15]$ with elements drawn uniformly from a unit sphere yields significantly better performance than RSM.

We conjecture two potential reasons for the better performance of the chosen distributions $p(\boldsymbol{\omega})$ in these cases, though a formal theoretical treatment is an interesting subject we defer to future work. Firstly, as $p(\boldsymbol{\omega})$ is synthetic, one can generate unlimited number of random features, which results in a much better approximation to the exact kernel in Equation (4). In contrast, RSM requires held-out samples from the data, which could be quite limited for a small data set. Second, in some cases, even with a small or similar number of random features to RSM, the performance of the selected distribution still leads to significantly better results. For those cases we conjecture that the selected $p(\boldsymbol{\omega})$ generates objects that capture semantic information more relevant to the estimation of $f(\boldsymbol{x})$, *when coupled* with our feature map under the dissimilarity measure $d(\boldsymbol{x}, \boldsymbol{\omega})$.

5 Analysis

In this section, we analyze the proposed framework from the perspectives of error decomposition. Let $\mathcal{H}$ be the RKHS corresponding to the kernel in Equation (4). Let

$$f_C := \underset{f \in \mathcal{H}}{argmin} \mathbb{E}[\ell(f(\boldsymbol{x}), y)] \quad s.t. \|f\|_{\mathcal{H}} \leq C \tag{9}$$

be the population risk minimizer subject to the RKHS norm constraint $\|f\|_{\mathcal{H}} \leq C$. And let

$$\hat{f_n} := \underset{f \in \mathcal{H}}{argmin} \frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{x}_i), y_i) \quad s.t. \|f\|_{\mathcal{H}} \leq C \tag{10}$$

be the corresponding empirical risk minimizer. In addition, let $\tilde{f_R}$ be the estimated function from our *random feature approximation* (Algorithm 1). Then denote the population and empirical risks as $L(f)$ and $\hat{L}(f)$ respectively. We have the following risk decomposition $L(\tilde{f_R}) - L(f) =$

$$\underbrace{(L(\tilde{f_R}) - L(\hat{f_n}))}_{randomfeature} + \underbrace{(L(\hat{f_n}) - L(f_C))}_{estimation} + \underbrace{(L(f_C) - L(f))}_{approximation}$$

In the following, we will discuss the three terms from the rightmost to the leftmost.

Function Approximation Error. The RKHS implied by the kernel in Equation (4) is

$$\mathcal{H} := \left\{ f \;\middle|\; f(\boldsymbol{x}) = \sum_{j=1}^{m} \alpha_j k(\boldsymbol{x}_j, \boldsymbol{x}),\ \boldsymbol{x}_j \in \mathcal{X}, \forall j \in [m],\ m \in \mathbb{N} \right\},$$

which is a smaller function space than the space of Lipschitz-continuous function w.r.t. the distance $d(\boldsymbol{x}_1, \boldsymbol{x}_2)$. As we show, any function $f \in \mathcal{H}$ is Lipschitz-continous w.r.t. the distance $d(.,.)$.

Proposition 1. *Let $\mathcal{H}$ be the RKHS corresponding to the kernel in Equation (4) derived from some metric $d(.,.)$. For any $f \in \mathcal{H}$,*

$$|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq L_f d(\boldsymbol{x}_1, \boldsymbol{x}_2)$$

where $L_f = \gamma C$.

We refer readers to the detailed proof in Appendix A.1. While any f in the RKHS is Lipschitz-continuous w.r.t. the given distance $d(.,.)$, we are interested in imposing additional smoothness via the RKHS norm constraint $\|f\|_{\mathcal{H}} \leq C$, and by the kernel parameter γ. The hope is that the best function f_C within this class approximates the true function f well in terms of the approximation error $L(f_C) - L(f)$. The stronger assumption made by the RKHS gives us a qualitatively better estimation error, as discussed below.

Estimation Error. Define D_λ as

$$D_\lambda := \sum_{j=1}^{\infty} \frac{1}{1 + \lambda/\mu_j}$$

where $\{\mu_j\}_{j=1}^{\infty}$ is the eigenvalues of the kernel in Equation (5) and λ is a tuning parameter. It holds that for any $\lambda \geq D_\lambda/n$, with probability at least $1-\delta$, $L(\hat{f}_n) - L(f_C) \leq c(\log\frac{1}{\delta})^2 C^2 \lambda$ for some universal constant c (Zhang, 2005). Here we would like to set λ as small as possible (as a function of n). By using the following kernel-independent bound: $D_\lambda \leq 1/\lambda$, we have $\lambda = 1/\sqrt{n}$ and thus a bound on the estimation error

$$L(\hat{f}_n) - L(f_C) \leq c(\log\frac{1}{\delta})^2 C^2 \sqrt{\frac{1}{n}}. \tag{11}$$

The estimation error is quite standard for a RKHS estimator. It has a much better dependency w.r.t. n (i.e. $n^{-1/2}$) compared to that of *k-nearest-neighbor method* (i.e. $n^{-2/(2+p_{\mathcal{X},d})}$) especially for higher effective dimension. A more careful analysis might lead to tighter bound on D_λ and also a better rate w.r.t. n. However, the analysis of D_λ for our kernel in Equation (4) is much more difficult than that of typical cases as we do not have an analytic form of the kernel.

Random Feature Approximation. Denote $\hat{L}(.)$ as the empirical risk function. The error from RF approximation $L(\tilde{f}_R) - L(\hat{f}_n)$ can be further decomposed as

$$(L(\tilde{f}_R) - \hat{L}(\tilde{f}_R)) + (\hat{L}(\tilde{f}_R) - \hat{L}(\hat{f}_n)) + (\hat{L}(\hat{f}_n) - L(\hat{f}_n))$$

where the first and third terms can be bounded via the same estimation error bound in Equation (11), as both $\tilde{f}_R$ and $\hat{f}_n$ have RKHS norm bounded by C. Therefore, in the following, we focus only on the second term of empirical risk. We start by analyzing the approximation error of the kernel $\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2) = \tilde{k}_R(\boldsymbol{x}_1, \boldsymbol{x}_2) - k(\boldsymbol{x}_1, \boldsymbol{x}_2)$ where

$$\tilde{k}_R(\boldsymbol{x}_1, \boldsymbol{x}_2) := \frac{1}{R}\sum_{j=1}^{R} \phi_j(\boldsymbol{x}_1)\phi_j(\boldsymbol{x}_2). \tag{12}$$

Proposition 2. *Let $\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2) = k(\boldsymbol{x}_1, \boldsymbol{x}_2) - \tilde{k}(\boldsymbol{x}_1, \boldsymbol{x}_2)$, we have uniform convergence of the form*

$$P\left\{\max_{\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}} |\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2)| > 2t\right\} \leq 2\left(\frac{12\gamma}{t}\right)^{2p_{\mathcal{X},d}} e^{-Rt^2/2},$$

where $p_{\mathcal{X},d}$ is the effective dimension of $\mathcal{X}$ under metric $d(.,.)$. In other words, to guarantee $|\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2)| \leq \epsilon$ with probability at least $1-\delta$, it suffices to have

$$R = \Omega\left(\frac{p_{\mathcal{X},d}}{\epsilon^2}\log(\frac{\gamma}{\epsilon}) + \frac{1}{\epsilon^2}\log(\frac{1}{\delta})\right).$$

We refer readers to the detailed proof in Appendix A.2. Proposition 2 gives an approximation error in terms of kernel evaluation. To get a bound on the empirical risk $\hat{L}(\tilde{f}_R) - \hat{L}(\hat{f}_n)$, consider the optimal solution of the empirical risk minimization. By the Representer theorem we have $\hat{f}_n(\boldsymbol{x}) = \frac{1}{n}\sum_i \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x})$ and $\tilde{f}_R(\boldsymbol{x}) = \frac{1}{n}\sum_i \tilde{\alpha}_i \tilde{k}(\boldsymbol{x}_i, \boldsymbol{x})$. Therefore, we have the following corollary.

Corollary 1. *To guarantee $\hat{L}(\tilde{f}_R) - \hat{L}(\hat{f}_n) \leq \epsilon$, with probability $1-\delta$, it suffices to have*

$$R = \Omega\left(\frac{p_{\mathcal{X},d} M^2 A^2}{\epsilon^2}\log(\frac{\gamma}{\epsilon}) + \frac{M^2 A^2}{\epsilon^2}\log(\frac{1}{\delta})\right).$$

where M is the Lipschitz-continuous constant of the loss function $\ell(., y)$, and A is a bound on $\|\boldsymbol{\alpha}\|_1/n$.

We refer readers to the detailed proof in Appendix A.3. For most of loss functions, A and M are typically small constants. Therefore, Corollary 1 states that it suffices to have number of Random Features proportional to the *effective dimension* $O(p_{\mathcal{X},d}/\epsilon^2)$ to achieve an ϵ approximation error.

Combining the three error terms, we can show that the proposed framework can achieve ϵ-suboptimal performance.

Claim 1. *Let $\tilde{f}_R$ be the estimated function from our* random feature approximation *based ERM estimator in Algorithm 1, and let f^* denote the desired target function. Suppose further that for some absolute constants $c_1, c_2 > 0$ (up to some logarithmic factor of $1/\epsilon$ and $1/\delta$):*

1. *The target function f^* lies close to the population risk minimizer f_C lying in the RKHS spanned by the D2KE kernel: $L(f_C) - L(f) \leq \epsilon/2$.*

2. *The number of training samples $n \geq c_1 C^4/\epsilon^2$.*

3. *The number of random features $R \geq c_2 p_{\chi,d}/\epsilon^2$.*

We then have that: $L(\tilde{f}_R) - L(f^*) \leq \epsilon$ with probability $1 - \delta$.

6 Experiments

We evaluate the proposed method in four different domains involving time-series, strings, texts, and images. First, we discuss the dissimilarity measures and data characteristics for each set of experiments. Then we introduce comparison among different distance-based methods and report corresponding results.

Distance Measures. We have chosen three well-known dissimilarity measures: 1) Dynamic Time Warping (DTW), for time-series (Berndt & Clifford, 1994); 2) Edit Distance (Levenshtein distance), for strings (Navarro, 2001); 3) Earth Mover's distance (Rubner et al., 2000) for measuring the semantic distance between two Bags of Words (using pretrained word vectors), for representing documents. 4) (Modified) Hausdorff distance (Huttenlocher et al., 1993; Dubuisson & Jain, 1994) for measuring the semantic closeness of two Bags of Visual Words (using SIFT vectors), for representing images. Note that Bag of (Visual) Words in 3) and 4) can also be regarded as a histogram. Since most distance measures are computationally demanding, having quadratic complexity, we adapted or implemented C-MEX programs for them; other codes were written in Matlab.

Datasets. For each domain, we selected 4 datasets for our experiments. For time-series data, all are multivariate time-series and the length of each time-series varies from 2 to 205 observations; three are from the UCI Machine Learning repository (Frank & Asuncion, 2010), the other is generated from the IQ (In-phase and Quadrature components) samples from a wireless line-of-sight communication system from GMU. For string data, the size of alphabet is between 4 and 8 and the length of each string ranges from 34 to 198; two of them are from the UCI Machine Learning repository and the other two from the LibSVM Data Collection (Chang & Lin, 2011). For text data, all are chosen partially overlapped with these in (Kusner et al., 2015). The length of each document varies from 9.9 to 117. For image data, all of datasets were derived from Kaggle; we computed a set of SIFT-descriptors to represent each image and the size of SIFT feature vectors of each image varies from 1 to 914. We divided each dataset into 70/30 train and test subsets (if there was no predefined train/test split). Properties of these datasets are summarized in Table 6 in Appendix B.

Table 2: Classification performance comparison on multi-variate Time-series with variable lengths

Methods	D2KE		KNN		DSK_RBF		DSK_ND		KSVM		RSM	
Datasets	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time
Auslan	**92.60**	42.4s	70.26	52.1s	92.47	43.4s	89.74	44.6s	85.58	95.6s	88.96	68.6s
pentip	**99.88**	4.5s	98.37	27.3s	98.02	125.4s	70.40	126.6s	98.37	125.3s	98.48	13.6s
ActRecog	**64.72**	43.4s	53.43	85.5s	55.58	64.9s	45.31	68.0s	51.65	75.2s	62.44	64.5s
IQ_radio	**86.87**	469.3s	60.25	3734s	77.41	13381s	47.31	12251s	80.52	10084s	70.84	1275.9s

Table 3: Classification performance comparison on Strings with variable lengths.

Methods	D2KE		KNN		DSK_RBF		DSK_ND		KSVM		RSM	
Datasets	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time
bit-str4	**90.00**	2.3s	80.00	3.2s	88.33	3.9s	86.67	3.5s	83.33	2.8s	86.67	2.6s
splice	**90.03**	46.9s	79.41	164.2s	87.88	204.9s	85.89	208.2s	67.29	169.9s	86.10	87.3s
dna3	**89.65**	125.1s	85.79	859.6s	86.75	1005.2s	87.15	1025.2s	46.10	991.8s	87.04	213.5s
mnist-str8	**98.49**	2196s	96.58	13207s	97.5	18666s	92.66	18604s	96.80	84684s	97.31	4608.6s

Table 4: Classification performance comparison on Bag of Words Vectors for Documents.

Methods	D2KE		KNN		DSK_RBF		DSK_ND		KSVM		RSM	
Datasets	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time
Bbcsport	**98.11**	90.0s	95.4	157.2s	97.15	514.5s	96.52	512.5s	96.27	511.0s	97.0	263.2s
Twitter	**74.2**	15.1s	71.3	65.4s	72.05	73.5s	71.93	73.0s	70.60	73.0s	71.84	17.9s
Recipe	**61.5**	257s	57.4	478.1s	58.51	1508.5s	57.53	1502.2s	52.63	2364.1s	58.55	480.2s
Ohsumed	**64.4**	532.3s	55.5	3650.5s	59.89	6236.1s	58.51	6229.2s	46.50	6108.1s	59.91	841.5s

Table 5: Classification performance comparison on Bag of Visual Words for images.

Methods	D2KE		KNN		DSK_RBF		DSK_ND		KSVM		RSM	
Datasets	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time	Accu	Time
flower	**46.03**	22.0s	33.33	96.4s	36.51	103.5s	36.51	102.4s	38.10	85.1s	33.33	38.6s
letters2	**55.45**	64.5s	42.52	90.9s	54.55	101.9s	53.27	99.7s	42.45	205.2s	53.34	89.8s
decor	70.06	150.3s	61.81	1017.3s	**70.83**	1225.1s	70.14	1221.9s	52.78	987.1s	67.25	425.2s
style	**40.29**	20.5s	36.57	348.0s	38.06	450.3s	30.59	449.2s	25.74	443.1s	37.68	352.6s

Baselines. We compare D2KE against 5 state-of-the-art baselines, including 1) *KNN*: a simple yet universal method to apply any distance measure to classification tasks; 2) *DSK_RBF* (Haasdonk & Bahlmann, 2004): distance substitution kernels, a general framework for kernel construction by substituting a problem specific distance measure in ordinary kernel functions. We use a Gaussian RBF kernel; 3) *DSK_ND* (Haasdonk & Bahlmann, 2004): another class of distance substitution kernels with negative distance; 4) *KSVM* (Loosli et al., 2016): learning directly from the similarity (indefinite) matrix followed in the original Krein Space; 5) *RSM* (Pekalska et al., 2001): building an embedding by computing distances from randomly selected representative samples.

Among these baselines, KNN, DSK_RBF, DSK_ND, and KSVM have quadratic complexity $O(N^2L^2)$ in both the number of data samples and the length of the sequences, while RSM has computational complexity $O(NRL^2)$, linear in the number of data samples but still quadratic in the length of the sequence. These compare to our method, D2KE, which has complexity $O(NRL)$, linear in both the number of data samples and the length of the sequence. For each method, we search for the best parameters on the training set by performing 10-fold cross validation. For our new method D2KE, since we generate random samples from the distribution, we can use as many as needed to achieve performance close to an exact kernel. We report the best number in the range $R = [4, 4096]$ (typically the larger R is, the better the accuracy). We employ a linear SVM implemented using LIBLINEAR (Fan et al., 2008) for all embedding-based methods (RSM and D2KE) and use LIBSVM (Chang & Lin, 2011) for precomputed dissimilairty kernels (DSK_RBF, DSK_ND, and KSVM). More details of experimental setup are provided in Appendix B.

Results. As shown in Tables 2, 3, 4, and 5, D2KE can consistently outperform or match the baseline methods in terms of classification accuracy while requiring far less computation time. There are several observations worth making here. First, D2KE performs much better than KNN, supporting our claim that D2KE can be a strong alternative to KNN across applications. Second, compared to the two distance substitution kernels DSK_RBF and DSK_ND and the KSVM method operating directly on indefinite similarity matrix, our method can achieve much better performance, suggesting that a representation induced from a truly PD kernel makes significantly better use of the data than indefinite kernels. Among all methods, RSM is closest to our method in terms of practical construction of the feature matrix. However, the random objects (time-series, strings, or sets) sampled by D2KE perform significantly better, as we discussed in section 4. More detailed discussions of the experimental results for each domain are given in Appendix C.

7 Conclusion and Future Work

In this work, we have proposed a general framework for deriving a *positive-definite* kernel and a feature embedding function from a given dissimilarity measure between input objects. The framework is especially useful for structured input domains such as sequences, time-series, and sets, where many well-established dissimilarity measures have been developed. Our framework subsumes at least two existing approaches as special or limiting cases, and opens up what we believe will be a useful new direction for creating embeddings of structured objects based on distance to random objects. A promising direction for extension is to develop such distance-based embeddings within a deep architecture to support use of structured inputs in an end-to-end learning system.

REFERENCES

Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(21):1–38, 2017.

Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. A theory of learning with similarity functions. *Machine Learning*, 72(1-2):89–112, 2008a.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 671–680. ACM, 2008b.

Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. *KDD workshop*, 10(16):359–370, 1994.

I Borg and P Groenen. Modern multidimensional scaling. series in statistics, 1997.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011.

Jianhui Chen and Jieping Ye. Training svm with indefinite kernels. In *Proceedings of the 25th international conference on Machine learning*, pp. 136–143. ACM, 2008.

Yihua Chen, Eric K Garcia, Maya R Gupta, Ali Rahimi, and Luca Cazzanti. Similarity-based classification: Concepts and algorithms. *Journal of Machine Learning Research*, 10(Mar):747–776, 2009a.

Yihua Chen, Maya R Gupta, and Benjamin Recht. Learning kernels from indefinite similarities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 145–152. ACM, 2009b.

Krzysztof Choromanski and Vikas Sindhwani. Recycling randomness with structure for sublinear time kernel expansions. *arXiv preprint arXiv:1605.09049*, 2016.

Krzysztof Choromanski, Mark Rowland, Tamas Sarlos, Vikas Sindhwani, Richard Turner, and Adrian Weller. The geometry of random features. In *International Conference on Artificial Intelligence and Statistics*, pp. 1–9, 2018.

Michael Collins and Nigel Duffy. Convolution kernels for natural language. In *Advances in neural information processing systems*, pp. 625–632, 2002.

Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research*, 13(Mar):795–828, 2012.

Marco Cuturi. Fast global alignment kernels. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 929–936, 2011.

Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pp. 3041–3049, 2014.

M-P Dubuisson and Anil K Jain. A modified hausdorff distance for object matching. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pp. 566–568. IEEE, 1994.

Robert PW Duin and Elżbieta Pękalska. The dissimilarity space: Bridging structural and statistical pattern recognition. *Pattern Recognition Letters*, 33(7):826–832, 2012.

X Yu Felix, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pp. 1975–1983, 2016.

Andrew Frank and Arthur Asuncion. Uci machine learning repository [http://archive. ics. uci. edu/ml]. irvine, ca: University of california. *School of information and computer science*, 213, 2010.

Yue Gao, Meng Wang, Dacheng Tao, Rongrong Ji, and Qionghai Dai. 3-d object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing*, 21(9):4290–4303, 2012.

Thore Graepel, Ralf Herbrich, Peter Bollmann-Sdorra, and Klaus Obermayer. Classification on pairwise proximity data. In *Advances in neural information processing systems*, pp. 438–444, 1999.

László Györfi, Michael Kohler, Adam Krzyzak, and Harro Walk. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Bernard Haasdonk and Claus Bahlmann. Learning with distance substitution kernels. In *Joint Pattern Recognition Symposium*, pp. 220–227. Springer, 2004.

Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis DeCoste. Compact random feature maps. In *International Conference on Machine Learning*, pp. 19–27, 2014.

Daniel P. Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*, 15(9): 850–863, 1993.

Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *Artificial Intelligence and Statistics*, pp. 583–591, 2012.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *International Conference on Machine Learning*, pp. 957–966, 2015.

Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, volume 85, 2013.

Hsuan-Tien Lin and Chih-Jen Lin. A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods. *submitted to Neural Computation*, 3:1–32, 2003.

Gaelle Loosli, Stephane Canu, and Cheng Soon Ong. Learning svm in krein spaces. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1204–1216, 2016.

Subhransu Maji and Alexander C Berg. Max-margin additive classifiers for detection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 40–47. IEEE, 2009.

Yusuke Mukuta, Yoshitaka Ushiku, and Tatsuya Harada. Alternating circulant random features for semigroup kernels. In *AAAI*, 2018.

Gonzalo Navarro. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88, 2001.

Cheng Soon Ong, Xavier Mary, Stéphane Canu, and Alexander J Smola. Learning with non-positive kernels. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 81. ACM, 2004.

Elzbieta Pekalska and Robert PW Duin. Dissimilarity-based classification for vectorial representations. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pp. 137–140. IEEE, 2006.

Elzbieta Pekalska and Robert PW Duin. Beyond traditional kernels: Classification in two dissimilarity-based representation spaces. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(6):729–744, 2008.

Elzbieta Pekalska, Pavel Paclik, and Robert PW Duin. A generalized kernel approach to dissimilarity-based classification. *Journal of machine learning research*, 2(Dec):175–211, 2001.

Jeffrey Pennington, X Yu Felix, and Sanjiv Kumar. Spherical random features for polynomial kernels. In *Advances in neural information processing systems*, pp. 1846–1854, 2015.

E Pkkalska and R Duin. The dissimilarity representation for pattern recognition. *World Scientific*, 2005.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pp. 3215–3225, 2017.

Bernhard Schölkopf. The kernel trick for distances. In *Advances in neural information processing systems*, pp. 301–307, 2001.

Bernhard Scholkopf, Sebastian Mika, Chris JC Burges, Philipp Knirsch, K-R Muller, Gunnar Ratsch, and Alexander J Smola. Input space versus feature space in kernel-based methods. *IEEE transactions on neural networks*, 10(5):1000–1017, 1999.

Mehmet Sezgin and Bülent Sankur. Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic imaging*, 13(1):146–166, 2004.

Aman Sinha and John C Duchi. Learning kernels with random features. In *Advances in Neural Information Processing Systems*, pp. 1298–1306, 2016.

Bharath Sriperumbudur and Zoltán Szabó. Optimal rates for random fourier features. In *Advances in Neural Information Processing Systems*, pp. 1144–1152, 2015.

Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. *IEEE transactions on pattern analysis and machine intelligence*, 34(3):480–492, 2012.

Lingfei Wu, Ian EH Yen, Jie Chen, and Rui Yan. Revisiting random binning features: Fast convergence and strong parallelizability. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1265–1274. ACM, 2016.

Lingfei Wu, Ian En-Hsu Yen, Jinfeng Yi, Fangli Xu, Qi Lei, and Michael Witbrock. Random warping series: A random features method for time-series embedding. In *International Conference on Artificial Intelligence and Statistics*, pp. 793–802, 2018.

Tong Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17(9):2077–2098, 2005.

A Proof of Theorem 1 and Theorem 2

A.1 Proof of Theorem 1

Proof. Note the function $g(t) = exp(-\gamma t)$ is Lipschitz-continuous with Lipschitz constant γ. Therefore,

$$\begin{aligned}
&|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| = |\langle f, \phi(\boldsymbol{x}_1) - \phi(\boldsymbol{x}_2)\rangle| \\
&\leq \|f\|_{\mathcal{H}} \|\phi(\boldsymbol{x}_1) - \phi(\boldsymbol{x}_2)\|_{\mathcal{H}} \\
&= \|f\|_{\mathcal{H}} \sqrt{\int_{\boldsymbol{\omega}} p(\boldsymbol{\omega})(\phi_{\boldsymbol{\omega}}(\boldsymbol{x}_1) - \phi_{\boldsymbol{\omega}}(\boldsymbol{x}_2))^2 d\boldsymbol{\omega}} \\
&\leq \|f\|_{\mathcal{H}} \sqrt{\int_{\boldsymbol{\omega}} p(\boldsymbol{\omega})\gamma^2 |d(\boldsymbol{x}_1, \boldsymbol{\omega}) - d(\boldsymbol{x}_2, \boldsymbol{\omega})|^2 d\boldsymbol{\omega}} \\
&\leq \gamma \|f\|_{\mathcal{H}} \sqrt{\int_{\boldsymbol{\omega}} p(\boldsymbol{\omega}) d(\boldsymbol{x}_1, \boldsymbol{x}_2)^2 d\boldsymbol{\omega}} \\
&\leq \gamma \|f\|_{\mathcal{H}} d(\boldsymbol{x}_1, \boldsymbol{x}_2) \leq \gamma C d(\boldsymbol{x}_1, \boldsymbol{x}_2)
\end{aligned}$$

□

A.2 Proof of Theorem 2

Proof. Our goal is to bound the magnitude of $\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2) = \tilde{k}_R(\boldsymbol{x}_1, \boldsymbol{x}_2) - k(\boldsymbol{x}_1, \boldsymbol{x}_2)$. Since $E[\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2)] = 0$ and $|\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2)| \leq 1$, from Hoefding's inequality, we have

$$P\left\{|\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2)| \geq t\right\} \leq 2\exp(-Rt^2/2)$$

a given input pair $(\boldsymbol{x}_1, \boldsymbol{x}_2)$. To get a unim bound that holds $\forall(\boldsymbol{x}_1, \boldsymbol{x}_2) \in \mathcal{X} \times \mathcal{X}$, we find an ϵ-covering $\mathcal{E}$ of $\mathcal{X}$ w.r.t. $d(.,.)$ of size $N(\epsilon, \mathcal{X}, d)$. Applying union bound over the ϵ-covering $\mathcal{E}$ for $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, we have

$$P\left\{\max_{\boldsymbol{x}_1' \in \mathcal{E}, \boldsymbol{x}_2' \in \mathcal{E}} |\Delta_R(\boldsymbol{x}_1', \boldsymbol{x}_2')| > t\right\} \leq 2|\mathcal{E}|^2 \exp(-Rt^2/2). \tag{13}$$

Then by the definition of $\mathcal{E}$ we have $|d(\boldsymbol{x}_1, \boldsymbol{\omega}) - d(\boldsymbol{x}_1', \boldsymbol{\omega})| \leq d(\boldsymbol{x}_1, \boldsymbol{x}_1') \leq \epsilon$. Together with the fact that $\exp(-\gamma t)$ is Lipschitz-continuous with parameter γ for $t \geq 0$, we have

$$|\phi_{\boldsymbol{\omega}}(\boldsymbol{x}_1) - \phi_{\boldsymbol{\omega}}(\boldsymbol{x}_1')| \leq \gamma\epsilon$$

and thus

$$|\tilde{k}_R(\boldsymbol{x}_1, \boldsymbol{x}_2) - \tilde{k}_R(\boldsymbol{x}_1', \boldsymbol{x}_2')| \leq 3\gamma\epsilon,$$

$$|k(\boldsymbol{x}_1, \boldsymbol{x}_2) - k(\boldsymbol{x}_1', \boldsymbol{x}_2')| \leq 3\gamma\epsilon$$

for $\gamma\epsilon$ chosen to be ≤ 1. This gives us

$$|\Delta_R(\boldsymbol{x}_1, \boldsymbol{x}_2) - \Delta_R(\boldsymbol{x}_1', \boldsymbol{x}_2')| \leq 6\gamma\epsilon \tag{14}$$

Combining equation 13 and equation 14, we have

$$\begin{aligned}
&P\left\{\max_{\boldsymbol{x}_1' \in \mathcal{E}, \boldsymbol{x}_2' \in \mathcal{E}} |\Delta_R(\boldsymbol{x}_1', \boldsymbol{x}_2')| > t + 6\gamma\epsilon\right\} \\
&\leq 2\left(\frac{2}{\epsilon}\right)^{2p_{\mathcal{X},d}} \exp(-Rt^2/2).
\end{aligned} \tag{15}$$

Choosing $\epsilon = t/6\gamma$ yields the result. □

A.3 Proof for Corollary 1

Proof. First of all, we have

$$\frac{1}{n}\sum_{i=1}^{n}\ell(\frac{1}{n}\sum_{j=1}^{n}\tilde{\alpha}_j\tilde{k}(\boldsymbol{x}_j,\boldsymbol{x}_i),y_i)$$
$$\leq \frac{1}{n}\sum_{i=1}^{n}\ell(\frac{1}{n}\sum_{j=1}^{n}\alpha_j\tilde{k}(\boldsymbol{x}_j,\boldsymbol{x}_i),y_i)$$

by the optimality of $\{\tilde{\alpha}_j\}_{j=1}^n$ w.r.t. the objective using the approximate kernel. Then we have

$$\begin{aligned}
&\hat{L}(\tilde{f}_R)-\hat{L}(\hat{f}_n)\\
&\leq \frac{1}{n}\sum_{i=1}^{n}\ell(\frac{1}{n}\sum_{j=1}^{n}\alpha_j\tilde{k}(\boldsymbol{x}_j,\boldsymbol{x}_i),y_i)-\ell(\frac{1}{n}\sum_{j=1}^{n}\alpha_j k(\boldsymbol{x}_j,\boldsymbol{x}_i),y_i)\\
&\leq M\frac{\|\boldsymbol{\alpha}\|_1}{n}\left(\max_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{X}}|\tilde{k}(\boldsymbol{x}_1,\boldsymbol{x}_2)-k(\boldsymbol{x}_1,\boldsymbol{x}_2)|\right)\\
&\leq MA\left(\max_{\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{X}}|\tilde{k}(\boldsymbol{x}_1,\boldsymbol{x}_2)-k(\boldsymbol{x}_1,\boldsymbol{x}_2)|\right)
\end{aligned}$$

where A is a bound on $\|\boldsymbol{\alpha}\|_1/n$. Therefore to guarantee

$$\hat{L}(\tilde{f}_R)-\hat{L}(\hat{f}_n)\leq\epsilon$$

we would need $\left(\max_{i,j\in[n]}|\Delta_R(\boldsymbol{x}_1,\boldsymbol{x}_2)|\right)\leq\hat{\epsilon}:=\epsilon/MA$. Then applying Theorem 2 leads to the result. □

B General Experimental Settings

Table 6: Properties of the datasets. TS, Str, Text, and Img stand for Time-Series, String, text, and Image respectively. Var stands for the number of variables for time-series, word embeddings, and image SIFT-descriptors while Alpb stands for the size of the alphabet for strings. Note that all data samples have quite different range of lengths in different datasets.

Domain	Name	Var/Alpb	Classes	Train	Test	length
TS	Auslan	22	95	1795	770	45-136
TS	pentip	3	20	2000	858	109-205
TS	ActRecog	3	7	1837	788	2-151
TS	IQ_radio	4	5	6715	6715	512
Str	bit-str4	4	10	140	60	44/158
Str	splice	4	3	2233	957	60
Str	dna3	4	2	3620	1555	147
Str	mnist-str8	8	10	60000	10000	17/99
Text	Bbcsport	300	5	517	220	43-469
Text	Twitter	300	3	2176	932	1-26
Text	Recipe	300	15	3059	1311	1-340
Text	Ohsumed	300	10	3999	5153	11-166
Img	flower	128	10	147	63	66/429
Img	decor	128	7	340	144	35/914
Img	style	128	7	625	268	6/530
Img	letters2	128	33	3277	1404	1/22

General Setup. For each method, we search for the best parameters on the training set by performing 10-fold cross validation. Following (Haasdonk & Bahlmann, 2004), we use an exact RBF kernel for DSK_RBF while choosing squared distance for DSK_ND. We use the Matlab implementation provided by Loosli et al. (2016) to run experiments for KSVM. Similarly, we adopted a simple method – random selection – to obtain $R = [4, 512]$ data samples as the representative set for RSM (Pekalska et al., 2001). For our new method D2KE, since we generate random samples from the distribution, we can use as many as needed to achieve performance close to an exact kernel. We

report the best number in the range $R = [4, 4096]$ (typically the larger R is, the better the accuracy). We employ a linear SVM implemented using LIBLINEAR (Fan et al., 2008) for all embedding-based methods (RSM, and D2KE) and use LIBSVM (Chang & Lin, 2011) for precomputed dissimilairty kernels (DSK_RBF, DSK_ND, and KSVM).

All datasets are collected from popular public websites for Machine Learning and Data Science research, including the UCI Machine Learning repository (Frank & Asuncion, 2010), the LibSVM Data Collection (Chang & Lin, 2011), and the Kaggle Datasets, except one time-series dataset IQ that is shared from researchers from George Mason University. Table 6 lists the detailed properties of the datasets from four different domains. All computations were carried out on a DELL dual-socket system with Intel Xeon processors at 2.93GHz for a total of 16 cores and 250 GB of memory, running the SUSE Linux operating system. To accelerate the computation of all methods, we used multithreading with 12 threads total for various distance computations in all experiments.

C Detailed Experimental Results on Time-Series, Strings, and Images

C.1 Results on multivariate time-series

Setup. For time-series data, we employed the most successful distance measure - DTW - for all methods. For all datasets, a Gaussian distribution was found to be applicable, parameterized by its bandwidth σ. The best values for σ and for the length of random time series were searched in the ranges [1e-3 1e3] and [2 50], respectively.

Results. As shown in Table 2, D2KE can consistently outperform or match all other baselines in terms of classification accuracy while requiring far less computation time for multivariate time-series. The first interesting observation is that our method performs substantially better than KNN, often by a large margin, i.e., D2KE achieves 26.62% higher performance than KNN on IQ_radio. This is because KNN is sensitive to the data noise common in real-world applications like IQ_radio, and has notoriously poor performance for high-dimensional data sets like Auslan. Moreover, compared to the two distance substitution kernels DSK_RBF and DSK_ND, and KSVM operating directly on indefinite similarity matrix, our method can achieve much better performance, suggesting that a representation induced from a truly p.d. kernel makes significantly better use of the data than indefinite kernels. Among all methods, RSM is closest to our method in terms of practical construction of the feature matrix. However, the random time series sampled by D2KE performs significantly better, as we discussed in section 4. First, RSM simply chooses a subset of the original data points and computes the distances between the whole dataset and this representative set; this may suffer significantly from noise or redundant information in the time-series. In contrast, our method samples a short random sequence that could both denoise and find the patterns in the data. Second, the number of data points that can be sampled is limited by the total size of the data while the number of possible random sequences drawn from the distribution is unlimited, making the feature space much more abundant. Third, RSM may incur significant computational cost for long time-series, due to its quadratic complexity in length.

C.2 Results on strings

Setup. For string data, there are various well-known edit distances. Here, we choose Levenshtein distance as our distance measure since it can capture global alignments of the underlying strings. We first compute the alphabet from the original data and then uniformly sample characters from this alphabet to generate random strings. We search for the best parameters for γ in the range [1e-5 1], and for the length of random strings in the range [2 50], respectively.

Results. As shown in Table 3, D2KE consistently performs better than or similarly to other distance-based baselines. Unlike the previous experiments where DTW is not a distance metric, Levenshtein distance is indeed a distance metric; this helps improve the performance of our baselines. However, D2KE still offers a clear advantage over baseline. It is interesting to note that the performance of DSK_RBF is quite close to our method's, which may be due to DKS_RBF with Levenshtein distance producing a c.p.d. kernel which can essentially be converted into a p.d. kernel. Notice that on relatively large datasets, our method, D2KE, can achieve better performance, and often with far less computation than other baselines with quadratic complexity in both number and length of data samples. For instance, on mnist-str8 D2KE obtains higher accuracy with an order of magnitude less

runtime compared to DSK_RBF and DSK_ND, and two orders of magnitude less than KSVM, due to higher computational costs both for kernel matrix construction and for eigendecomposition.

C.3 Results on sets of Word Vectors for text

Setup. For text data, following (Kusner et al., 2015) we use the earth mover's distance as our distance measure between two documents, since this distance has recently demonstrated a strong performance when combining with KNN for document classifications. We first compute the Bag of Words for each document and represent each document as a histogram of word vectors, where google pretrained word vectors with dimension size 300 is used. We generate random documents consisting of each random word vectors uniformly sampled from the unit sphere of the embedding vector space $\mathbb{R}^{300}$. We search for the best parameters for γ in the range [1e-2 1e1], and for length of random document in the range [3 21].

Results. As shown in Table 4, D2KE outperforms other baselines on all four datasets. First of all, all distance based kernel methods perform better than KNN, illustrating the effectiveness of SVM over KNN on text data. Interestingly, D2KE also performs significantly better than other baselines by a notiably margin, in large part because document classification mainly associates with "topic" learning where our random documents of short length may fit this task particularly well. For the datasets with large number of documents and longer length of document, D2KE achieves about one order of magnitude speedup compared with other exact kernel/similarity methods, thanks to the use of random features in D2KE.

C.4 Results on sets of SIFT-descriptors for images

Setup. For image data, following (Pekalska et al., 2001; Haasdonk & Bahlmann, 2004) we use the modified Hausdorff distance (MHD) (Dubuisson & Jain, 1994) as our distance measure between images, since this distance has shown excellent performance in the literature (Sezgin & Sankur, 2004; Gao et al., 2012). We first applied the open-source OpenCV library to generate a sequence of SIFT-descriptors with dimension 128, then MHD to compute the distance between sets of SIFT-descriptors. We generate random images of each SIFT-descriptor uniformly sampled from the unit sphere of the embedding vector space $\mathbb{R}^{128}$. We search for the best parameters for γ in the range [1e-3 1e1], and for length of random SIFT-descriptor sequence in the range [3 15].

Results. As shown in Table 5, D2KE performance outperforms or matches other baselines in all cases. First, D2KE performs best in three cases while DSK_RBF is the best on dataset decor. This may be because the underlying SIFT features are not good enough and thus random features is not effective to find the good patterns quickly in images. Nevertheless, the quadratic complexity of DSK_RBF, DSK_ND, and KSVM in terms of both the number of images and the length of SIFT descriptor sequences makes it hard to scale to large data. Interestingly, D2KE still performs much better than KNN and RSM, which again supports our claim that D2KE can be a strong alternative to KNN and RSM across applications.