Information Sciences 436–437 (2018) 214–226

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Data-independent Random Projections from the feature-map of the homogeneous polynomial kernel of degree two

Daniel López-Sánchez*, Juan Manuel Corchado, Angélica González Arrieta

Department of Computer Science, University of Salamanca, Salamanca, Spain

ARTICLE INFO

Article history:
Received 9 December 2016
Revised 28 August 2017
Accepted 13 January 2018
Available online 16 January 2018

MSC:
00-01
99-00

Keywords:
Random Projection
Non-linear dimensionality reduction
Polynomial kernel

ABSTRACT

This paper presents a novel non-linear extension of the Random Projection method based on the degree-2 homogeneous polynomial kernel. Our algorithm is able to implicitly map data points to the high-dimensional feature space of that kernel and from there perform a Random Projection to an Euclidean space of the desired dimensionality. Pairwise distances between data points in the kernel feature space are approximately preserved in the resulting representation. As opposed to previous kernelized Random Projection versions, our method is data-independent and preserves much of the computational simplicity of the original algorithm. This is achieved by focusing on a specific kernel function, what allowed us to analyze the effect of its associated feature mapping in the distribution of the Random Projection hyperplanes. Finally, we present empirical evidence that the proposed method outperforms alternative approaches in terms of pairwise distance preservation, while being significantly more efficient. Also, we show how our method can be used to approximate the accuracy of non-linear classifiers with efficient linear classifiers in some datasets.

1. Introduction

Random Projection (RP) is a simple yet computationally efficient dimensionality reduction method. While other dimensionality reduction methods compute the directions onto which the data will be projected by analyzing a training dataset, RP populates the projection matrix with a random normal distribution, thus requiring low computational effort. This technique is based on a theoretical result known as the Johnson–Lindenstrauss (JL) lemma: For any $0 < \epsilon < 1$ and $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ there is a map $f : \mathbb{R}^d \to \mathbb{R}^k$ for $k = \mathcal{O}(\epsilon^{-2}\log(n))$ such that:

$$\forall i, j \quad (1-\epsilon)||x_i - x_j||^2 \leqslant ||f(x_i) - f(x_j)||^2 \leqslant (1+\epsilon)||x_i - x_j||^2 \tag{1}$$

Furthermore, this map can be found in randomized polynomial time. For a simple proof of this lemma, refer to Dasgupta and Gupta [10]. Originally, the map f consisted of projecting the points from $\mathbb{R}^d$ to $\mathbb{R}^k$ by means of a $d \times k$ matrix whose elements were drawn from a Gaussian distribution. Later on, Achlioptas proved that the projection matrix could be populated according to a much simpler distribution [1]. Concretely, he showed that the entries of the projection matrix R can be computed as shown in Eq. (2); where Achlioptas used $s = 1$ and $s = 3$ (see [22]).

$$r_{ij} = \sqrt{s}\begin{cases} 1 & \text{with prob.} \quad 1/2s \\ 0 & \text{with prob.} \quad 1 - 1/s \\ -1 & \text{with prob.} \quad 1/2s \end{cases} \tag{2}$$

* Corresponding author.
E-mail address: lope@usal.es (D. López-Sánchez).

https://doi.org/10.1016/j.ins.2018.01.022
0020-0255/

In fact, Achlioptas proved that the only requirement to satisfy the JL lemma is that the entries of the projection matrix must be independent random variables with zero mean and unit variance. Once the $d \times k$ matrix R has been populated, an arbitrary set of n points represented as an $n \times d$ matrix X can be projected from $\mathbb{R}^d$ to $\mathbb{R}^k$ according to Eq. (3) (see [1]).

$$X'_{n\times k} = \frac{1}{\sqrt{k}} X_{n\times d} R_{d\times k} \tag{3}$$

Using the distribution proposed by Achlioptas to populate the projection matrix reduces the computational cost of the projection. This is because the multiplication by $\sqrt{s}$ present in Eq. (2) can be delayed, so the computation of the projection itself reduces to aggregate evaluation (i.e. summation and subtraction but no multiplication).

Over the years, various authors [3,32] have explored the possibility of performing random projections from the feature space associated to different kernel functions. This is of interest because of two main reasons: (1) if the JL-lemma is satisfied, the pairwise distances between samples in the kernel feature space will be preserved in the resulting representation, so the low-dimensional projected points will preserve most of the structure from the kernel feature-space; and (2), since Random Projection preserves separability margins [3], classification problems may become more lineally solvable after the Random Projection form a kernel feature space. If this is the case, the classification accuracy of scalable linear classifiers will increase. Motivated by these advantages, Zhao et al. recently proposed a kernelized variant of the original Random Projection Algorithm, capable of working with an arbitrary kernel function [32]. As a drawback, their proposal lost some of the beneficial features of the original Random Projection algorithm. In particular, the method proposed in [32] is data-dependent and, unlike the original Random Projection algorithm, requires an expensive training phase. In addition, this algorithm is not compatible with the database-friendly distribution proposed by Achlioptas [1] and its distance-preserving capabilities have not been analyzed empirically. Moreover, it has been theoretically proven that formulating an algorithm capable of performing a Random Projection from the feature space of an arbitrary kernel function, by just having black-box access to the kernel function but no training samples (i.e., in a data-independent manner) is not possible [3,5]. However, the question of whether this could be achieved for particular kernel functions such as the polynomial kernel remains open.

In this paper, we present a novel method to efficiently perform random projections from the feature space of the homogeneous polynomial kernel of degree two. By focusing on a specific kernel function, our method overcomes the limitations of previous kernelization attempts of the Random Projection algorithm (i.e., data-dependence and training inefficiency). In addition, our method is compatible with the database-friendly distribution proposed by Achlioptas. Our experimental results evidence that, by using our method, one can efficiently generate a low-dimensional representation of data samples that condenses the structure of data in the feature space of the degree-2 homogeneous polynomial kernel, approximately preserving the pairwise distances between samples in that space. Because of the data-independent nature of the proposed method, it can be applied in online-learning scenarios [13] where no data samples are initially available. In addition, this representation can be used to train efficient linear classifiers that approximate the accuracy of their non-linear counterparts.

Formally, we propose replacing the inner product that takes place during the matrix multiplication of Eq. (3) (i.e. projection of data points of X onto directions in R) by the kernel function (see Eq. (4)). By doing so, the columns of the projection matrix and the data samples are mapped by $\phi(\cdot)$ and the projection takes place in the kernel feature space:

$$x'_{ij} = \frac{1}{\sqrt{k}} K\Big([x_{i1}, x_{i2}, \ldots, x_{id}],\ [r_{1j}, r_{2j}, \ldots, r_{dj}]\Big) = \frac{1}{\sqrt{k}} \langle \phi(row_i X), \phi(col_j R)\rangle \tag{4}$$

where $\langle \cdot, \cdot \rangle$ denotes the scalar product. However, this is not in general equivalent to explicitly mapping the data points to the extended feature space by means of $\phi : \mathbb{R}^d \to \mathcal{H}$ and then performing a Random Projection. This is because we cannot guarantee that the distribution of the elements in R will be preserved by the embedding $\phi(\cdot)$. Therefore, our goal will be to define the projection matrix R in such a way that when its columns are mapped by $\phi(\cdot)$ to the implicit, high-dimensional feature space $\mathcal{H}$ the result will be a set of valid random projection directions. This is the reason why we must focus on a specific kernel function, so we can analyze how its associated mapping $\phi(\cdot)$ affects the columns of the projection matrix.

The rest of this paper is structured as follows. Section 2 reviews the related work available in the literature. Section 3 describes the proposed approach and how it manages to efficiently perform a valid Random Projection form the feature space of the degree-2 homogeneous polynomial kernel. An improved method to construct the implicit projection hyperplanes is then described in Section 4. In Section 5, we provide empirical evidence that our method approximates a Random Projection form the feature space of the degree-2 homogeneous polynomial kernel, so the pairwise distances between points in the feature space are approximately preserved in the reduced representation. Additional experimental results regarding the suitability of our method for the task of classification are reported in Section 6. Finally, Section 7 summarizes the conclusions and future research lines.

2. Related work

This paper focuses on the design of a novel method to perform a Random Projection from the feature space associated to a specific kernel function, namely the degree-2 homogeneous polynomial kernel. By performing this random projection, it is possible to efficiently generate a compact representation of samples that captures the structure of data in the kernel feature space. Conveniently, this is achieved without any explicit computation of the expensive mapping function $\phi(\cdot)$. By focusing on a specific kernel function, we were able to preserve some of the most important beneficial features of the original

Random Projection algorithm, namely its data-independence, efficiency and the possibility of implementing the projection in terms of aggregate evaluation, thanks to the distribution proposed by Achlioptas [1]. One important advantage of this kernel family is that the associated feature space is known and of finite (but very high) dimension. The intuition behind the homogeneous polynomial kernel is that it is often useful to construct new features as the product of original features. The feature space of the degree-2 homogeneous polynomial kernel consists of all the possible second order monomials of the original features of a sample, so it is able to account for interactions between the original features. For instance, in a text-processing problem where the features of samples represent the number of occurrences of a word in a document, the features in the kernel feature space correspond to co-occurrences of pairs of words, which may be more informative than individual frequencies. This explains why numerous machine learning problems are more easily solved in the feature space of polynomial kernels [8,20,29,30]. Moreover, the suitability polynomial kernels for various classification tasks is supported by the results presented in [7]. In this work, the authors analyzed the effectiveness of the polynomial kernel of degree two in the context of Support Vector Machine (SVM) classification. They propose an alternative to the standard kernel-SVM method. Specifically, they apply fast linear-SVM classification methods to data samples explicitly mapped by the mapping function $\phi(\cdot)$ associated to the polynomial kernel of degree two. Their results evidence that, using this approach, it is possible to achieve accuracy rates close to those achieved when using highly non-linear kernels (e.g. radial basis kernels) for various large-scale datasets. However, using the explicit feature map $\phi(\cdot)$ is highly inconvenient due to the size of the resulting representation. To overcome this limitation, the authors of [7] applied various techniques that rely on the sparsity of training data, but of course this is not always the case.

In [21], the authors developed a novel technique to generate a set of random hyperplanes over an implicit kernel feature space $\{r_1 \dots r_k\} \in \mathcal{H}$ which are restricted to be approximately Gaussian. The main idea of their proposal, often known as the Kulis–Grauman approach, is to construct the hyperplanes in the implicit feature space $\mathcal{H}$ as a weighted sum of a subset of the training-data points, thus being a data-dependent method. The Kulis–Grauman approach was recently adopted to perform a kernelized version of Random Projection in the context of image classification [2]. Although this version is compatible with an arbitrary kernel function, it loses much of the computational efficiency and conceptual simplicity of Random Projection. In addition, it makes use of the specific data to be projected during the training stage (as opposed to standard RP, in which the projection matrix R is completely data-independent and no training stage is required). Later on, this kernelized version of RP was extended and applied to the problem of image clustering [32]. The authors proposed three versions of the algorithm: Kernelised Gaussian Random Projection (KG-RP), Kernelised Orthonormal Random Projection (KORP), and KPCA-based Random Projection (KPCA-RP). The three versions differ in the manner in which the hyperplanes are generated over the implicit feature space, but the mentioned limitations (computational and conceptual complexity, data-dependence and training phase need) still apply. In addition, these studies do not provide empirical evidence that the JL-lemma still applies when using their proposed versions of the algorithm (i.e. pairwise distances between data points in $\mathcal{H}$ should be preserved in the resulting space). Our experimental results suggest that our method outperforms that of Alavi et al. [2] in terms of distance preservation while having a much lower computational cost.

In fact, Balcan et al. [3,5] showed that performing a Random Projection from the feature space of an arbitrary kernel function, by just having black-box access to the kernel function but no training samples to estimate the data-distribution is not possible. They work, however, left as an open problem whether this could be done for specific kernel functions such as the polynomial kernel. In that sense, our work answers that question positively, by providing a means to perform a Random Projection from the feature space of the degree-2 homogeneous polynomial kernel without the need of accessing the distribution of samples that will be projected.

As mentioned before, one of the motivations to design methods that can perform Random Projections from the feature space of kernel functions is the possibility of using the resulting representation as a basis for efficient categorization with linear classifiers. Since separability margins are approximately preserved after a Random Projection [5], a linear support vector machine classifier (SVM) trained on the resulting representation may be able to approximate the accuracy of its kernelized counterparts, with the advantage that linear SVMs are significantly more efficient [31]. In fact, a lot of effort has been recently put into designing more scalable versions of the traditional kernel-SVM. A notable example is the development of Least Squares Support Vector Machines (LS-SVM) [6,17,26], which replace the constrained quadratic programming problem in the formulation of kernel-SVMs by a set of linear equations, thus being applicable to large-scale problems.

3. Random Projection from the feature space of the homogeneous polynomial kernel

Given a set of n points from $\mathbb{R}^d$ represented as a $n \times d$ matrix X, the goal of our method is to map them to the high-dimensional, implicit feature space $\mathcal{H}$ of the degree-2 homogeneous polynomial kernel and then perform a random projection from $\mathcal{H}$ to $\mathbb{R}^k$ so the resulting points are represented as a $n \times k$ matrix X'. Preferably, these operations should be performed implicitly via the kernel function (as shown in Eq. (4)), avoiding any explicit computation of the mapping function $\phi(\cdot)$. As mentioned before, in order to ensure that Eq. (4) approximates a valid Random Projection in the kernel feature space of the degree-2 homogeneous polynomial kernel, we will define an appropriate distribution for the projection matrix R by analyzing the properties of that specific kernel function. Essentially, homogeneous polynomial kernels are a subset of the family of polynomial kernels, parametrized by the polynomial degree p and the constant term c:

$$K(x, y) = (x^T y + c)^p \tag{5}$$

When c is fixed to zero, the result is a homogeneous polynomial kernel. As opposed to other kernel functions, the implicit feature space associated to the homogeneous polynomial kernel of degree two can be easily determined. To do so, it suffices to write the kernel function in the extended from. The homogeneous polynomial kernel of degree two is defined as:

$$K(x,y) = (x^T y)^2 \tag{6a}$$

$$= \left(\sum_{i=1}^{d} x_i y_i\right)^2 = \sum_{i=1}^{d} (x_i)^2 (y_i)^2 + \sum_{i=1}^{d-1} \sum_{j=i+1}^{d} (\sqrt{2} x_i x_j)(\sqrt{2} y_i y_j) \tag{6b}$$

where $x, y \in \mathbb{R}^d$. Then, it is clear that the previous expression corresponds to the inner product of x and y in the feature space defined by the following embedding (see, for example [25]):

$$\phi(x) = (x_1^2, \ldots, x_d^2, \sqrt{2}x_1x_2, \ldots, \sqrt{2}x_1x_d, \sqrt{2}x_2x_3, \ldots, \sqrt{2}x_2x_d, \ldots, \sqrt{2}x_{d-1}x_d) \tag{7}$$

Once the feature space associated to the kernel is known, one might consider performing a standard Random Projection from this representation. Following this approach, data samples should be transformed by $\phi(\cdot)$ and then projected to a low-dimensional space by standard Random Projection. In fact, if the sparse distribution proposed by Achlioptas is used, the projection matrix R might be efficiently stored using a sparse matrix implementation. However, explicitly computing $\phi(x)$ might be a very expensive and almost intractable task. This is mainly due to the fact that the dimension of $\phi(x)$ grows as $\mathcal{O}(d^2)$. As a consequence, trying to explicitly transform a set of samples with $\phi(\cdot)$ requires intensive computations and a significant storage capacity, especially if data samples are dense and sparse matrix implementations cannot be used (which is often the case). For instance, in Section 5 we will see that storing the explicit form of $\phi(x)$ for 200 samples from a dataset of 96×96 colour images would require more than 284 GB of memory, which is far beyond the current capacity of most devices' main memory. Nevertheless, in Section 5 we empirically compare this approach to our proposed method both in terms of performance and efficiency. From now on, we will refer to the explicit transformation of data samples by $\phi(\cdot)$ followed by the application of standard Random Projection as the *explicit approach*.

Returning to our proposed method, once the feature space of the kernel has been determined, it becomes possible for us to define a distribution for the projection matrix R such that its columns will follow a valid Random Projection distribution when transformed by $\phi(\cdot)$. In this manner we will be able to perform the Random Projection via the kernel function, thus avoiding any explicit computation of $\phi(\cdot)$. Particularly, we propose populating the projection matrix R according to Eq. (8).

$$r_{ij} = \frac{\sqrt{s}}{\sqrt[4]{2}} \begin{cases} 1 & \text{with prob.} \quad 1/2s \\ 0 & \text{with prob.} \quad 1 - 1/s \\ -1 & \text{with prob.} \quad 1/2s \end{cases} \tag{8}$$

Given that the embedding associated to the kernel function is known, we can analyze its effect on the proposed distribution. Let $r \in \mathbb{R}^d$ be any of the columns of R; then, consider the distribution of features in $\phi(r)$ (i.e. one of the projection hyperplanes onto which data points will be projected). Two different distributions emerge:

$$\phi(r) = \left(\underbrace{r_1^2, \cdots, r_d^2}_{\text{distribution } A}, \underbrace{\sqrt{2}r_1r_2, \cdots, \sqrt{2}r_1r_d, \sqrt{2}r_2r_3, \cdots, \sqrt{2}r_2r_d, \cdots, \sqrt{2}r_{d-1}r_d}_{\text{distribution } B} \right) \tag{9}$$

$$A := \begin{cases} \frac{s}{\sqrt{2}} & \text{with prob.} \quad 1/s \\ 0 & \text{with prob.} \quad 1 - 1/s \end{cases} \qquad B := \sqrt{s'} \begin{cases} 1 & \text{with prob.} \quad 1/2s' \\ 0 & \text{with prob.} \quad 1 - 1/s' \\ -1 & \text{with prob.} \quad 1/2s' \end{cases} \tag{10}$$

where $s' = s^2$. Conveniently, distribution B corresponds to the original formula proposed by Achlioptas (see Eq. (2)). Notice that only the first d elements of $\phi(r)$ follow distribution A. The total number of features in $\phi(r)$ is $d + d((d-1)/2)$; hence, the percentage of components of $\phi(r)$ that follow distribution A tends to zero as d grows:

$$\lim_{d \to \infty} \frac{d}{d + d(\frac{d-1}{2})} = \lim_{d \to \infty} \frac{2}{d+1} = 0 \tag{11}$$

Therefore, as the original dimension of samples d increases, the computation defined in Eq. (13) with the projection matrix R populated according to Eq. (8) approximates the explicit mapping of points from $\mathbb{R}^d$ to $\mathcal{H}$ and its reduction by means of Random Projection from $\mathcal{H}$ to $\mathbb{R}^k$. Conveniently, this is done without any explicit computation of the embedding $\phi(\cdot)$.

In case one needs to work with very low-dimensional samples (i.e. low d values), it is possible to define a modified polynomial Kernel which ensures that all the components of $\phi(\cdot)$ follow distribution B. The implicit feature space associated to the kernel presented in Eq. (12) is that of the homogeneous polynomial kernel of degree two but without the first d components (precisely those following distribution A).

$$K(x,y) = (x^T y)^2 - (x^2)^T (y^2) \tag{12}$$

The experimental results reported in Section 5 show that pairwise distances between data points in $\mathcal{H}$ are approximately preserved in the reduced representation generated by the proposed method. However, if we compare the distance distortion induced by our method to the distortion induced by the explicit mapping of points to $\mathcal{H}$ and its reduction to $\mathbb{R}^k$ by means of standard Random Projection (i.e., the explicit approach), we see that the former is slightly higher. To explain this deviation, we must consider the requirements that the projection matrix must satisfy in order for the JL-lemma to hold. According to Achlioptas [1], the elements of the projection matrix must be independent random variables with zero mean and unit variance. Unfortunately, our implicit projection hyperplanes do not completely satisfy the independence condition. Consider for example the following elements of $\phi(r)$: $\sqrt{2}r_1r_2$, $\sqrt{2}r_2r_3$ and $\sqrt{2}r_1r_3$. Once the sign of the first two elements is known, the sign of the third is completely determined. This partial dependence between the elements of the projection directions causes the deviation in the distance preservation with respect to the explicit approach. In the next section, an alternative method to overcome this problem is proposed.

4. Using the Central Limit Theorem to construct the projection hyperplanes

In this section, a novel technique to construct a random projection matrix in the implicit feature space is proposed. The elements of this implicit projection matrix will be nearly independent variables drawn form a normal distribution with zero mean and unit variance. To populate this implicit matrix and perform the random projection in the implicit feature space (without any explicit computation of $\phi(\cdot)$), our method relies both on the properties of kernels and the Central Limit Theorem [18].

First, we populate m projection matrices each of dimension $d \times k$. We will refer to the l-th projection matrix as $R^{(l)}$. After this, an arbitrary set of n points represented as an $n \times d$ matrix X can be implicitly mapped from $\mathbb{R}^d$ to $\mathcal{H}$ and then randomly projected to $\mathbb{R}^k$ according to Eq. (13) (the resulting points are represented as an $n \times k$ matrix X').

$$x'_{ij} = \frac{1}{\sqrt{k}} \sum_{l=1}^{m} K\left(row_i X,\ col_j R^{(l)}\right) \tag{13}$$

To show that Eq. (13) actually approximates a valid Random Projection, we begin by applying the definition of the kernel function $K(x, y) = \langle \phi(x), \phi(y) \rangle$ to reveal the computations that take place in the feature space (Eq. (14)). Then, we apply a fundamental property of inner products, $\langle x, y + z \rangle = \langle x, y \rangle + \langle x, z \rangle$ [24], to rewrite the equation as a single inner product in Eq. (15), and use the summation notation to simplify the formula in Eq. (16).

$$x'_{ij} = \frac{1}{\sqrt{k}} \sum_{l=1}^{m} K\left(row_i X,\ col_j R^{(l)}\right) = \frac{1}{\sqrt{k}} \sum_{l=1}^{m} \langle \phi(row_i X), \phi\left(col_j R^{(l)}\right) \rangle \tag{14}$$

$$\begin{aligned} &= \frac{1}{\sqrt{k}} \langle \phi(row_i X),\ \phi\left(col_j R^{(1)}\right) \rangle + \cdots + \frac{1}{\sqrt{k}} \langle \phi(row_i X),\ \phi\left(col_j R^{(m)}\right) \rangle \\ &= \frac{1}{\sqrt{k}} \langle \phi(row_i X),\ \phi\left(col_j R^{(1)}\right) + \cdots + \phi\left(col_j R^{(m)}\right) \rangle \end{aligned} \tag{15}$$

$$= \frac{1}{\sqrt{k}} \langle \phi(row_i X),\ \sum_{l=1}^{m} \phi\left(col_j R^{(l)}\right) \rangle \tag{16}$$

At this point, we have shown that Eq. (13) corresponds to the mapping of the i-th sample in X to the kernel feature space $\mathcal{H}$ and its projection onto a vector of the form $\sum_{l=1}^{m} \phi(col_j R^{(l)})$. Therefore, in order for Eq. (13) to compute a valid Random Projection from the kernel feature space, we just need to ensure that $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ follows a valid RP-distribution. In [1], Achlioptas proved that any random distribution with zero mean and unit variance can be used to generate the projection vectors. In this case, we will use the Central Limit Theorem to ensure that the projection vectors follow a normal distribution with zero mean and unit variance. To this extent, we define the following distribution to populate the projection matrices $\{R^{(1)}, R^{(2)}, \cdots, R^{(m)}\}$:

$$r_{ij} = \frac{\sqrt{s}}{\sqrt[4]{2m}} \begin{cases} 1 & \text{with prob. } 1/2s \\ 0 & \text{with prob. } 1 - 1/s \\ -1 & \text{with prob. } 1/2s \end{cases} \tag{17}$$

Now, let us analyze the distribution of variables in the implicit feature space for the columns of the projection matrices. Let $r \in \mathbb{R}^d$ be a column of any of the m projection matrices populated according to Eq. (17), then:

$$\phi(r) = \left(\underbrace{r_1^2, \cdots, r_d^2}_{\text{distribution A}}, \underbrace{\sqrt{2}r_1r_2, \cdots, \sqrt{2}r_1r_d, \sqrt{2}r_2r_3, \cdots, \sqrt{2}r_2r_d, \cdots, \sqrt{2}r_{d-1}r_d}_{\text{distribution B}} \right) \tag{18}$$

$$A := \begin{cases} \frac{s}{\sqrt{2m}} & \text{with prob.} \quad 1/s \\ 0 & \text{with prob.} \quad 1-1/s \end{cases} \qquad B := \sqrt{\frac{s'}{m}} \begin{cases} 1 & \text{with prob.} \quad 1/2s' \\ 0 & \text{with prob.} \quad 1-1/s' \\ -1 & \text{with prob.} \quad 1/2s' \end{cases} \tag{19}$$

where $s' = s^2$. The mean and the variance of distribution B can be determined as follows [28]:

$$\mu = \mathbf{E}(B) = \sum_b b \cdot P(b) = \sqrt{\frac{s'}{m}} \cdot \frac{1}{2s'} - \sqrt{\frac{s'}{m}} \cdot \frac{1}{2s'} = 0 \tag{20}$$

$$\sigma^2 = \mathbf{E}\left([B-\mu]^2\right) = \sum_b (b-\mu)^2 \cdot P(b) = \left(\sqrt{\frac{s'}{m}} - 0\right)^2 \cdot \frac{1}{2s'} + \left(-\sqrt{\frac{s'}{m}} - 0\right)^2 \cdot \frac{1}{2s'} = \frac{s'}{m} \cdot \frac{1}{2s'} + \frac{s'}{m} \cdot \frac{1}{2s'} = \frac{1}{m} \tag{21}$$

where Σ_b denotes the sum over all of the possible values of B and $P(b)$ is the probability of the specific value b in the random distribution B. Once the mean and variance of B are known, we can apply the Central Limit Theorem (CLM) [18] to ensure that, for sufficiently large m, the majority of the elements of $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ will follow a normal distribution of zero mean and unit variance. Only the first d elements of the projection directions will not follow that distribution. Fortunately, as in the previous section the percentage of elements that follow distribution A with respect to the desired distribution B tends to zero as d grows larger.[1] Note that the m hyperparameter, which controls the number of vectors summed to form the final projection hyperplanes, must be specified manually. In the context of the CLM, a sample size of 25 or 30 is usually recommended to obtain a good approximation of the desired normal distribution [16]. The effect of choosing different m values in the proposed method is empirically studied in Section 5.

Since by the CLT most of the entries in $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ are distributed according to $\mathcal{N}(0,1)$, and we have shown that Eq. (13) corresponds to the mapping of a sample to the kernel feature space followed by a projection onto a vector of the form $\sum_{l=1}^{m} \phi(col_j R^{(l)})$, we can conclude that Eq. (13) approximates a valid Random Projection from the kernel feature space. Note that, due to the fact that the first d features in $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ do not follow a valid RP-distribution, we cannot categorically state that the JL-lemma will be satisfied. However, as we have seen in the previous section, the percentage of features that follow distribution A tends to zero as d grows larger. In fact, our experimental results evidence that the effect of these features is indeed neglectable, as in all cases our method was able to induce an average distortion in samples as low as that of the explicit approach by using a big enough m value. Also, Section 5.4 reports on the results of statistical tests which ensure that there is no significant difference between the distance preservation capabilities of the proposed approach and a standard Random Projection from the explicitly-computed kernel feature space. Nevertheless, we could also use the modified kernel function (see Eq. (12)), which removes these inconveniently distributed features from the kernel feature space. In that case, all the features in $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ would follow the desired standard normal distribution.

Finally, note that the distribution proposed in this section to populate the projection matrices is conveniently a generalization of the one proposed in the previous section. When the m hyperparameter is set to one (i.e. only one projection matrix is used) the distribution proposed in Eq. (17) is the same as in Eq. (8). Similarly, when $m = 1$, Eq. (13) is equivalent to the method described in the previous section. From now on, we will refer to our method as Polynomial-kernel Random Projection (P-RP), indicating in each case the value used for the m hyperparameter.

4.1. Computational complexity of the proposed approach

The original Random projection algorithm has a remarkably low computational complexity: the training step (i.e. populating the projection matrix R) takes $\mathcal{O}(dk)$ time, and projecting the $n \times d$ data matrix X into $\mathbb{R}^k$ can be done by means of Eq. (3) at the cost of time $\mathcal{O}(ndk)$ [4]. In the proposed variation of the algorithm, the training step consist of populating m projection matrices, thus having a complexity of $\mathcal{O}(mdk)$. Given that the polynomial kernel can be computed in $\mathcal{O}(d)$, the complexity of projecting the $n \times d$ data matrix X into k dimensions by means of Eq. (13) is of order $\mathcal{O}(mndk)$. However, since m is a hyperparameter whose value does not depend on the input data, it can be considered as a constant. By doing this, the complexities of the training and test phases turn out to be the same as in the original RP algorithm. This indicates that the scalability of P-RP when the number of training samples and its dimension grow is the same that the scalability of RP.

P-RP has a lower computational complexity than most common non-linear dimensionality reduction methods. For instance, Kernel PCA, Multidimensional scaling (MDS) and Isomap have a training computational complexity of $\mathcal{O}(n^3)$ [27]. This makes them scale poorly as the amount of training instances increase. Other methods (such as Autoencoders and Local Linear Coordination) depend on solving non-convex optimization problems. Hence, they might get stuck in local minima.

[1] Given that the first d elements of $\phi(col_j R^{(l)})$ are never negative, the first d elements of $\sum_{l=1}^{m} \phi(col_j R^{(l)})$ will grow larger as m increases, having a negative impact in the distance preservation capabilities of P-RP. To solve this problem, we multiply half of the elements in the summation of Eq. (13) by minus one. By doing so, the first d elements of the projection directions will exhibit a zero mean, and the distribution of the remaining entries will not be modified due to de symmetry of distribution B.

5. Experimental results on distance preservation

The proposed technique aims to implicitly project data samples from $\mathbb{R}^d$ to $\mathcal{H}$ and then project them to $\mathbb{R}^k$ in such a way that pairwise distances between samples in $\mathcal{H}$ are preserved in the resulting space. To evaluate the distance preservation properties of the different approaches, we compare the squared Euclidean distance between two dimensionality-reduced data vectors to their squared Euclidean distance in the kernel feature space. To do so, we use the following measure: let x, y be a couple of samples from $\mathbb{R}^d$ and let x', y' be their reduction to $\mathbb{R}^k$, then:

$$distortion_{x,y} = \frac{abs(||x\prime - y\prime||^2 - ||\phi(x) - \phi(y)||^2)}{||\phi(x) - \phi(y)||^2} \tag{22}$$

This measure can be easily interpreted. For example, if $distortion_{x,y} = 0.12$ we can conclude that the distance between both samples in $\mathcal{H}$ suffered a 12% distortion (increase or decrease) in the resulting feature space. Note that the computation of $||\phi(x) - \phi(y)||^2$ can be performed without any explicit evaluation of $\phi(\cdot)$, via the kernel function [23]:

$$||\phi(x) - \phi(y)||^2 = -2K(x, y) + K(x, x) + K(y, y) \tag{23}$$

To measure the distortion induced by a given method while reducing a set of n samples, the average distortion among all the $\binom{n}{2}$ possible pairs of different samples is computed. We shall use the average distortion measure to compare the different approaches proposed in this paper. To validate the effectiveness of the proposed method, the experiments were carried out using three datasets from different domains, namely artificial vision and speech recognition. We also evaluated the effectiveness of alternative methods such as the explicit approach (see Section 3) and Kernelised Gaussian Random Projection (KG-RP) [32]. In the case of KG-RP the p hyperparameter, which controls the number of samples used to estimate the distribution of data, must be set manually. By its nature, increasing its value can only cause the performance of the algorithm to improve, at the expense of longer running times. The authors of this method suggest using $p = \mathcal{O}(\sqrt{n})$ to achieve a balance between efficiency and performance [21,32], where n is the number of training samples. We followed this recommendation in our experiments and evaluated KG-RP with $p = 3\sqrt{n}$, $6\sqrt{n}$, and $9\sqrt{n}$. To support our claims about the scalability of the proposed method, each distance distortion result reported in this section is provided along with the corresponding training/embedding times.

5.1. Experiments on CIFAR-10

The CIFAR-10 dataset [19] consists of 60,000 32×32 colour images distributed among 10 different classes. It contains exactly 6000 images per class. The train/test split is usually arranged with 50,000 training images and 10,000 test images. Given that the images are of size 32×32 with three channels, the sample size d is 3072. The dimensionality of the implicit feature space $\mathcal{H}$ (for the homogeneous polynomial kernel of degree two) is 4,717,056. In this case, the explicit approach (i.e. explicit transformation of the samples by means of $\phi(\cdot)$ and reduction by linear Random Projection) is still tractable. However, it is extremely demanding both in terms of computational power and memory. For example, storing 200 samples from $\mathcal{H}$ as a matrix of 32-bit floats requires approximately 3.6 GB of memory.

To compare the different approaches, 200 samples were selected at random from the whole dataset. Then, the samples were reduced by means of the different methods proposed in this paper and the average pairwise distance distortion was measured. Fig. 1a) shows the average distortion induced by the different methods as the resulting dimension k grows.

As we can se in Fig. 1a), the method proposed in Section 3 (i.e., P-RP with $m = 1$) provides a reasonably low distortion given its simplicity. However, as explained before, a significant gap exits between the effectiveness of this method and the explicit approach. On the other hand, the method proposed in Section 4 to overcome that limitation shows a rapid decrease in the induced distortion as the hyper-parameter m grows. Table 1 compiles the resulting average distortions obtained using different methods and various values of k and m. To mitigate the stochastic nature of these methods, each experiment was performed ten times. The average result and the standard deviation are reported. Regarding the ruining times, P-RP is by far the fastest alternative, especially considering training times (which in the case of P-RP are solely due to the initialization of the random projection matrices). As expected, the explicit approach reports the longest running times due to the expensive explicit computation of the kernel feature space.

Finally, we analyze the effect of the hyperparameter m on the average distortion induced by P-RP. To this extent, the dimension of the resulting space k was fixed to 160 and the average distortion was evaluated for a wide range of m values. The results were compared to the average distortion induced by the explicit approach, also with $k = 160$. This comparison is shown in Fig. 1b). The results evidence that using values of m greater than 100 will ensure a performance of P-RP equivalent to that of the explicit approach. In addition, it is even possible to obtain a very close approximation to this performance with much lower m values.

5.2. Experiments on ISOLET

The ISOLET Spoken Letter Database is a dataset of letters from the English alphabet pronounced by native speakers under controlled conditions. The data consist of two productions of each letter by a total of 150 subjects (with uniform gender distribution). The dataset contains a total of 7800 samples, each corresponding to the features extracted from the

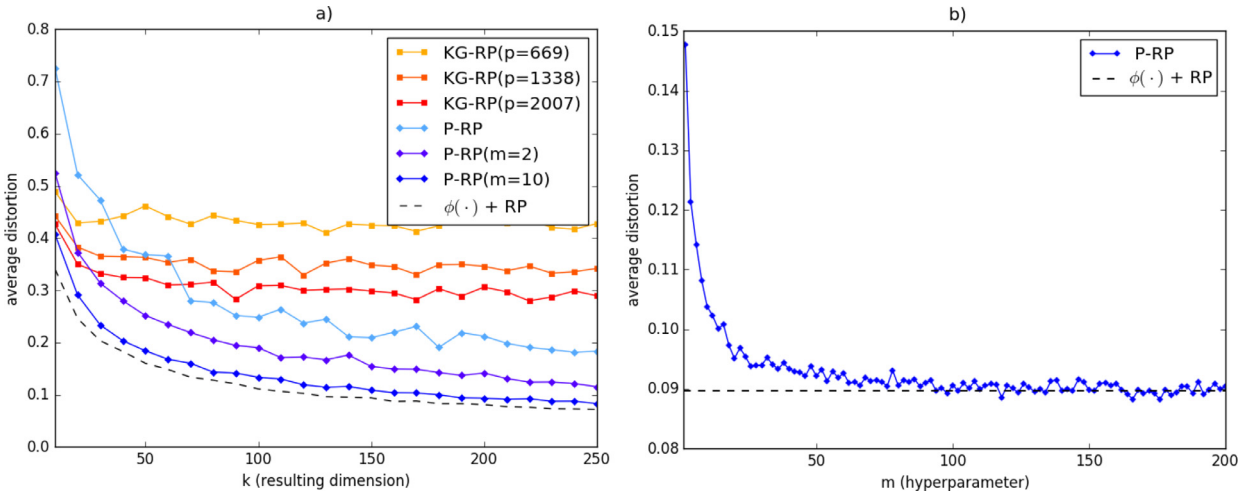

Fig. 1. a) Average distortion induced on CIFAR10 samples by using different methods as the resulting dimension grows. b) Effect of different values for the hyperparameter m on CIFAR10 samples transformed by P-RP; the resulting dimension was fixed to 160.

Table 1
Average pairwise distance distortion induced by different methods and hyperparameters on 200 samples from CIFAR10. Training/embedding times are also provided for all the experiments.

Method	k=40	k=80	k=120	k=160
$\phi(\cdot) + RP$	0.177 ± 0.007 7433.7/4837.7 ms	0.122 ± 0.003 9974.4/5263.4 ms	0.103 ± 0.004 18, 288.6/6142.6 ms	0.090 ± 0.003 36, 351.8/9226.9 ms
P-RP (m=1)	0.386 ± 0.040 3.6/0.41 ms	0.277 ± 0.022 5.6/0.61 ms	0.239 ± 0.035 10.6/0.81 ms	0.239 ± 0.033 16.6/0.95 ms
P-RP (m=2)	0.294 ± 0.025 2.14/0.81 ms	0.212 ± 0.018 5.05/1.29 ms	0.170 ± 0.009 6.85/1.36 ms	0.147 ± 0.006 12.5/1.66 ms
P-RP (m=10)	0.204 ± 0.008 11.6/3.58 ms	0.145 ± 0.007 28.7/3.66 ms	0.119 ± 0.006 38.9/7.8 ms	0.101 ± 0.005 47.4/8.11 ms
P-RP (m=30)	0.188 ± 0.016 35.2/9.9 ms	0.132 ± 0.005 70.0/16.9 ms	0.109 ± 0.004 116.8/18.5 ms	0.091 ± 0.002 148.7/23.4 ms
KG-RP (p=669)	0.438 ± 0.027 3311.0/5.02 ms	0.431 ± 0.030 3344.5/5.44 ms	0.430 ± 0.019 3430.0/6.59 ms	0.424 ± 0.010 3527.6/6.86 ms
KG-RP (p=1338)	0.357 ± 0.019 13, 473.4/9.38 ms	0.348 ± 0.024 13, 634/9.41 ms	0.335 ± 0.020 13, 878.2/11.61 ms	0.339 ± 0.013 14, 156.4/11.83 ms
KG-RP (p=2007)	0.323 ± 0.027 31, 532.1/15.52 ms	0.305 ± 0.018 32, 111.9/15.63 ms	0.292 ± 0.014 32, 607.4/16.31 ms	0.291 ± 0.015 33, 442.0/18.0 ms

pronunciation of a letter. For more details on the feature extraction process see [12]. Each sample consist of 617 features; therefore, the dimensionality of the associated implicit feature space $\mathcal{H}$ is 190,653. Note that the storage of 200 samples from $\mathcal{H}$ as a matrix of 32-bit floats requires approximately 145.5 MB of memory.

To compare the different approaches, 200 samples were selected at random from the whole dataset. Then, the samples were reduced by means of the different methods proposed in this paper and the average pairwise distance distortion was measured. Fig. 2a) shows the average distortion induced by the different methods as the resulting dimension k grows. Table 2 compiles the resulting average distortions obtained using different methods and various values of k and m. Each experiment was performed ten times; the average result and standard deviation of those ten runs are reported. To analyze the effect of the hyperparameter m on the average distortion induced by P-RP, the dimension of the resulting space k was fixed to 160 and the average distortion was evaluated for a wide range of m values. The results were compared to the average distortion induced by the explicit approach also with $k = 160$. This comparison is shown in Fig. 2b).

5.3. Experiments on STL-10

The STL-10 dataset [9], inspired by the CIFAR-10 dataset, is another very popular image recognition dataset. The two major differences with respect to CIFAR-10 are the lower amount of labelled images per class and the dimensions of the images in SLT-10 which are significantly bigger. The dataset consists of 500 training images, 800 test images per class and 100,000 unlabelled images for unsupervised learning. Each image is of size 96 $\times$ 96 with three colour channels; therefore each sample contains 27,648 features. The dimensionality of the implicit feature space $\mathcal{H}$ is then 382,219,776. Consequently, storing 200 samples from $\mathcal{H}$ as a matrix of 32-bit floats requires approximately 284.7 GB of memory, which is far beyond

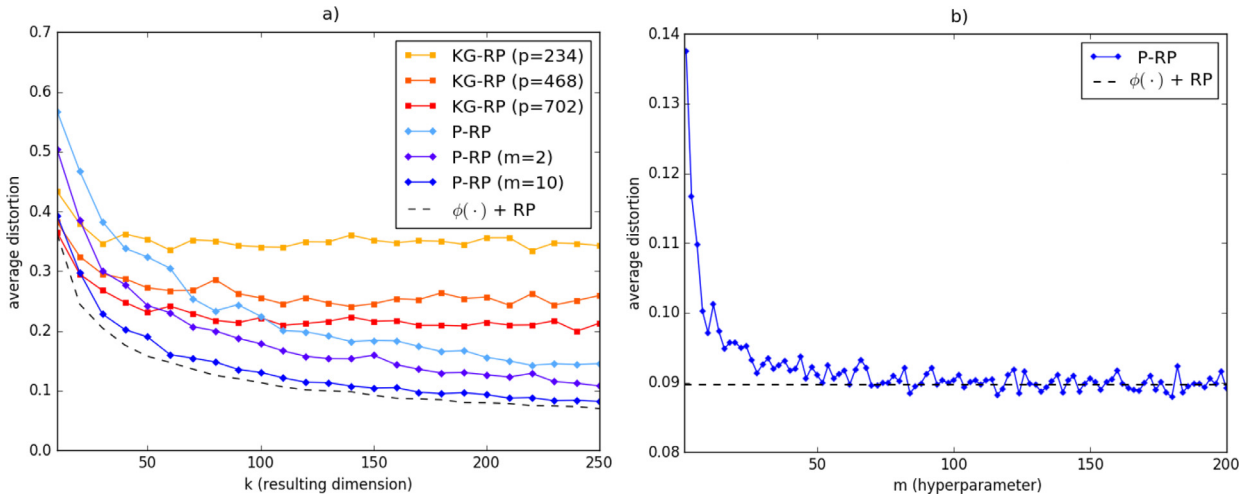

Fig. 2. a) Average distortion induced on ISOLET samples by using different methods as the resulting dimension grows. b) Effect of different values for the hyperparameter m on ISOLET samples transformed by P-RP; the resulting dimension was fixed to 160.

Table 2
Average pairwise distance distortion induced by different methods and hyperparameters on 200 samples from ISOLET. Training/embedding times are also provided for all the experiments.

Method	k=40	k=80	k=120	k=160
$\phi(\cdot) + RP$	0.174 ± 0.004 226.6/665.3 ms	0.123 ± 0.007 428.2/692.6 ms	0.102 ± 0.005 594.1/723.5 ms	0.089 ± 0.008 881.6/732.1 ms
P-RP (m=1)	0.361 ± 0.086 0.9/0.09 ms	0.245 ± 0.041 1.4/0.14 ms	0.215 ± 0.028 1.8/0.19 ms	0.179 ± 0.038 2.7/0.25 ms
P-RP (m=2)	0.279 ± 0.022 0.3/0.17 ms	0.192 ± 0.017 0.6/0.34 ms	0.155 ± 0.008 1.2/0.4 ms	0.139 ± 0.019 1.6/0.57 ms
P-RP (m=10)	0.196 ± 0.010 2.2/0.84 ms	0.139 ± 0.009 5.2/1.4 ms	0.114 ± 0.007 9.3/1.96 ms	0.100 ± 0.007 9.7/2.59 ms
P-RP (m=30)	0.181 ± 0.008 7.1/2.48 ms	0.132 ± 0.009 16.8/4.0 ms	0.107 ± 0.004 27.0/6.09 ms	0.092 ± 0.003 31.5/4.86 ms
KG-RP (p=234)	0.344 ± 0.026 448.7/0.8 ms	0.349 ± 0.028 454.8/0.86 ms	0.348 ± 0.017 482.6/0.96 ms	0.335 ± 0.02 492.9/1.25 ms
KG-RP (p=468)	0.283 ± 0.029 1668.9/1.45 ms	0.250 ± 0.017 1768.5/2.05 ms	0.257 ± 0.018 1845.9/2.13 ms	0.269 ± 0.02 1896.1/2.82 ms
KG-RP (p=702)	0.243 ± 0.025 3588.1/2.49 ms	0.222 ± 0.019 3730.68/2.56 ms	0.219 ± 0.017 3820.1/3.33 ms	0.204 ± 0.022 4012.4/3.41 ms

the current capacity of personal computers and a challenging volume even for high-end computing systems. This illustrates how rapidly (exponentially indeed) the explicit approach becomes intractable when the dimensionality of data samples grows. Working with this sample size renders the explicit approach intractable. For this reason, it was not evaluated in this case study. However, we could still measure the effectiveness of the other compared methods. To compare the different approaches, 500 samples were selected at random from the whole dataset. Then, the samples were reduced by means of the different methods proposed in this paper and the average pairwise distance distortion was measured. Fig. 3a) shows the average distortion induced by different methods as the resulting dimension k grows. Table 3 compiles the resulting average distortions obtained using different methods and various values of k and m. Each experiment was performed ten times; the average result and standard deviation of those ten runs are reported. Fig. 3b) analyzes the effect of the hyperparameter m on the average distortion induced by P-RP (k was fixed to 160).

5.4. Friedman test and post-hoc tests

This subsection reports on the results of several statistical tests that support our claim that the proposed method approximates a Random Projection from the feature space of the degree-2 homogeneous polynomial kernel. We applied the Friedman method with post-hoc tests as described in [14,15]. For all the tests, the performance measure used was $1 - avg.\ distortion$. Intuitively, big values of this performance measure correspond to small induced distortions in pairwise distances.

First of all, we analyzed whether a significant difference existed in the performance of the compared methods over the different datasets evaluated. Under the null-hypothesis, the Friedman test states that all the algorithms are equivalent, so a

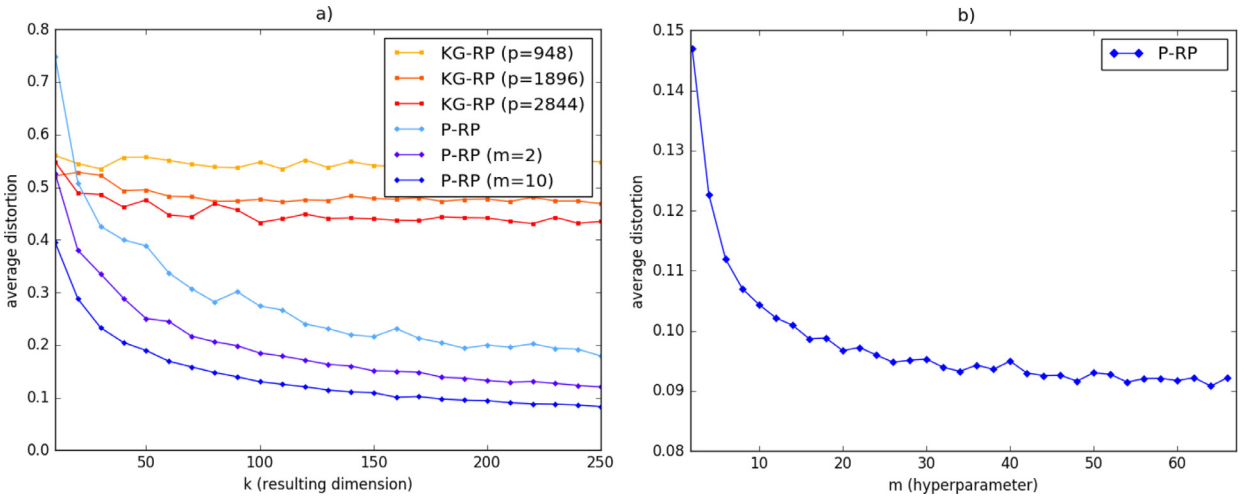

Fig. 3. a) Average distortion induced on 500 STL-10 samples by using different methods as the resulting dimension grows. b) Effect of different values for the hyperparameter m on STL-10 samples transformed by P-RP; the resulting dimension was fixed to 160.

Table 3
Average pairwise distance distortion induced by different methods and hyper-parameters on 500 samples from STL-10. Training/embedding times are also provided for all the experiments.

Method	k=40	k=80	k=120	k=160
P-RP (m=1)	0.360 ± 0.086 38.9/7.65 ms	0.254 ± 0.041 63.6/10.88 ms	0.215 ± 0.028 88.6/10.76 ms	0.179 ± 0.038 118.08/15.69 ms
P-RP (m=2)	0.279 ± 0.022 21.58/13.09 ms	0.192 ± 0.017 51.7/22.05 ms	0.155 ± 0.008 68.09/23.2 ms	0.139 ± 0.019 89.75/32.11 ms
P-RP (m=10)	0.196 ± 0.010 113.33/53.71 ms	0.139 ± 0.009 222.9/92.04 ms	0.114 ± 0.007 330.2/123.96 ms	0.100 ± 0.007 434.1/174.08 ms
P-RP (m=30)	0.181 ± 0.008 316.2/183.58 ms	0.132 ± 0.009 654.5/304.2 ms	0.107 ± 0.004 968.8/396.21 ms	0.092 ± 0.003 1295.6/487.22 ms
KG-RP (p=948)	0.559 ± 0.025 6934.6/215.1 ms	0.546 ± 0.018 6976.3/220.8 ms	0.544 ± 0.018 7123.3/224.5 ms	0.541 ± 0.004 7277.3/225.8 ms
KG-RP (p=1896)	0.495 ± 0.024 29,368.1/439.7 ms	0.485 ± 0.010 29,545/444.6 ms	0.479 ± 0.013 29,956.4/449.0 ms	0.482 ± 0.008 30,524.9/454.9 ms
KG-RP (p=2844)	0.469 ± 0.020 69,502.2/634.3 ms	0.452 ± 0.017 70,184.2/643.5 ms	0.445 ± 0.011 71,195.8/644.2 ms	0.434 ± 0.012 72,354.2/647.9 ms

Table 4
Adjusted p-values for the comparison of the control algorithm ($\phi(\cdot) + RP$) with the remaining algorithms (Bonferroni-Dunn, Holm and Hochberg tests).

i	Algorithm	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Hoch}
1	**KG-RP (p=3 · $\sqrt{n}$)**	4.62394×10^{-12}	3.23676×10^{-11}	3.23676×10^{-11}	3.23676×10^{-11}
2	**KG-RP (p=6 · $\sqrt{n}$)**	5.43308×10^{-9}	3.80316×10^{-8}	3.25985×10^{-8}	3.25985×10^{-5}
3	**KG-RP (p=9 · $\sqrt{n}$)**	3.06125×10^{-6}	2.142877×10^{-5}	1.53062×10^{-5}	1.53062×10^{-5}
4	**P-RP**	6.79534×10^{-6}	4.75674×10^{-5}	2.71813×10^{-5}	2.71813×10^{-5}
5	**P-RP (m=2)**	0.00204	0.014328	0.00614	0.00614
6	P-RP (m=10)	0.04550	0.31850	0.09100	0.09100
7	P-RP (m=30)	0.31731	2.22117	0.31731	0.31731

rejection of this hypothesis implies the existence of differences among the performance of the different methods. Using the Friedman statistic over the performance results previously reported in this section resulted in a value of 80.805 (distributed according to chi-square with 7 degrees of freedom) and a corresponding p-value of 4.3258×10^{-11}. As a consequence, we can reject the null-hypothesis and conclude that significant differences exist between the performances of the compared methods. Hence, a post-hoc statistical analysis must be performed. We selected the best performing method, namely $\phi(\cdot) + RP$ (i.e., the explicit approach), as the control. Then, the Bonferroni-Dunn, Holm and Hochbergs tests [15] were used to find whether the control method presents statistical differences when compared to the remaining approaches in terms of performance. The adjusted p-values for these tests are reported in Table 4, stressing in bold those methods that were worse than the control considering a level of significance $\alpha = 0.05$.

Table 5
Feature number, kernel feature space size, training sample number and test sample number of the datasets used in Section 6.

Dataset	Sample size d	Size of $\mathcal{H}$	#train	#test
ijcnn1	22	253	49,990	91,701
MNIST38	784	307,720	11,982	1984
covtype	54	1485	464,809	116,203
webspam	254	32,385	280,000	70,000

Table 6
Classification accuracy on various datasets obtained by using: a linear SVM over the original features, a Gaussian-kernel SVM, a linear SVM over the embedding defined by the polynomial kernel of degree two, a linear SVM trained on P-RP features and a linear SVM trained on KG-RP features.

Methods	IJCNN		MNIST38		covtype		webspam	
	Param.	Acc.	Param.	Acc.	Param.	Acc.	Param.	Acc.
Raw features Linear SVM	C=32	92.21%	C=0.03125	96.82%	C=0.0625	76.35%	C=32	93.15%
Raw features RBF-SVM	C=32	98.69%	C=2	99.70%	C=32	96.08%	C=8	99.20%
$\phi(\cdot):\mathbb{R}^d \to \mathcal{H}$ Linear SVM	C=0.125	97.84%	C=2	99.29%	C=2	80.09%	C=8	98.44%
P-RP (m=1)	C=4	97.24%	C=4	98.63%	C=0.1	79.77%	C=0.05	97.77%
Linear SVM	k=250	±0.05	k=2000	±0.17	k=500	±0.04	k=2000	±0.14
P-RP (m=2)	C=4	97.29%	C=4	98.80%	C=0.1	79.72%	C=0.05	97.56%
Linear SVM	k=250	±0.08	k=2000	±0.19	k=500	±0.01	k=2000	±0.13
P-RP (m=10)	C=4	97.31%	C=4	98.90%	C=0.1	79.72%	C=0.05	97.60%
Linear SVM	k=250	±0.07	k=2000	±0.2	k=500	±0.03	k=2000	±0.06
P-RP (m=20)	C=4	97.32%	C=4	98.94%	C=0.1	79.73%	C=0.05	97.65%
Linear SVM	k=250	±0.09	k=2000	±0.24	k=500	±0.01	k=2000	±0.14
P-RP (m=30)	C=4	97.33%	C=4	98.95%	C=0.1	79.73%	C=0.05	97.66%
Linear SVM	k=250	±0.06	k=2000	±0.2	k=500	±0.02	k=2000	±0.09
KG-RP (p=$9\sqrt{n}$)	C=4	97.36%	C=0.5	98.68%	C=0.1	79.43%	C=0.05	97.76%
Linear SVM	k=250	±0.10	k=2000	±0.07	k=500	±0.10	k=2000	±0.04

As we can see, while KG-RP and P-RP with $m <= 2$ perform worse than the control approach, no significant differences were detected by the tests when comparing the distance preservation performance of our method using $m >= 10$ and the explicit approach $\phi(\cdot) + RP$ (with a level of significance $\alpha = 0.05$). This supports our claim that, when a big enough value is selected for the m hyperparameter, our method approximates a Random Projection form the feature space of the degree-2 homogeneous polynomial kernel.

6. Experimental results on classification accuracy

Although Radial Basis Function (RBF) is the most widely used type of kernel in the context of support vector machine (SVM) classification, it suffers from some limitations. Mainly, the implicit feature map $\phi_{RBF}(\cdot)$ associated to the RBF kernel is infinite dimensional, which enforces the application of the kernel trick to train RBF-SVMs. As a result, RBF-SVMs are inefficient and poorly scalable as compared to their linear counterparts [31].

As an alternative, Chang et al. [7] proposed mapping the data samples by the feature transformation associated to the polynomial kernel of degree two as a prior step to fast linear-SVM classification. Their experimental results evidence that, using this method on some datasets, one may achieve accuracy rates close to those of using highly non-linear kernels. Unfortunately, as we have mentioned before, the dimensionality of $\mathcal{H}$ grows rapidly as the original dimension of samples increases. Therefore, the method proposed in [7] is not convenient when training samples have a significant number of features and those features are not sparse.

On the other hand, the method proposed in this paper can be used to efficiently condense the structure of a dataset in $\mathcal{H}$ to a low-dimensional representation of the samples. This representation can be used to train efficient linear classifiers that obtain accuracy rates almost as good as those trained on samples explicitly mapped by means of the embedding $\phi : \mathbb{R}^d \to \mathcal{H}$. To support our claim, we reproduced the experimental protocol developed in [7]. The accuracies obtained by using P-RP as a feature extractor prior to linear classification[2] were compared to the results presented by Chang et al. Table 6 compiles the experimental results obtained on various datasets by various dimensionality reduction and classification methods.[3] For details on the characteristics of the different datasets refer to Table 5.

[2] We used the linear SVM implementation provided by LIBSVM [11].
[3] The results concerning L-SVM, RBF-SVM and $\phi(\cdot)$ L-SVM where directly taken from [7].

The results presented above evidence that the proposed method can be used as a previous step to linear classification, boosting the capabilities of linear classifiers and approximating, in some cases, the performance attained by highly non-linear classification methods. Interestingly, the effect of increasing the m hyperparameter of P-RP has a more subtle effect in the classification accuracy than it had in the results concerning distance preservation in the previous section. Conveniently, high accuracy rates can be achieved using very small m values. In fact, no significant accuracy improvements where registered using $m > 20$. Moreover, the best results with P-RP for webspam and covtype datasets were achieved by using $m = 1$, which corresponds to the method described in Section 3. In addition, we can see that KG-RP achieved similar or slightly lower accuracies than our method on all datasets.

7. Discussion and future work

In this paper, a novel non-linear dimensionality reduction method has been presented. The proposed algorithm makes it possible to implicitly perform a Random Projection from the feature space associated to the polynomial kernel of degree two to an euclidean space of the desired dimension. This projection is conveniently performed without any explicit computation of the embedding $\phi : \mathbb{R}^d \to \mathcal{H}$. As opposed to previous techniques to perform random projections from kernel feature spaces [2,32], our method preserves the data-independence and efficiency properties of the original Random Projection algorithm. In fact, the training stage only involves the population of the random projection matrices, thus preserving much of the simplicity of the original Random Projection method. Moreover, the proposed method is compatible with the database-friendly distribution proposed by Achlioptas [1], thus allowing its implementation in terms of aggregate evaluation. This is achieved by delaying the floating-point multiplication present in Eq. (17). In addition, our work gives a positive answer to the question brought up by Balcan et al. [3,5], proving that it is possible to perform a Random Projection from the feature space of an specific kernel function without access to the data distribution.

Our experimental results show that the proposed method approximates the distance preservation properties of Random Projection, so the generated representations approximately preserve the pairwise distances between samples in the kernel feature space. Because of the data-independent nature of the proposed method, it could be applied in online-learning scenarios [13] where no data samples are initially available. In addition, the method proposed in this paper can be used to train efficient linear SVMs that approximate the performance of non-linear SVMs with the homogeneous polynomial kernel. First, P-RP is used to project the training data from $\mathcal{H}$ to $\mathbb{R}^k$; by doing so, the non-linearity of features in $\mathcal{H}$ is retained in the reduced space. A linear classifier can then be trained on $\mathbb{R}^k$ and used to solve classification problems that were not linearly separable in the original feature space.

The major shortcoming of the proposed method is that it was designed to work solely with a specific kernel function. In this regard, the efficiency and data-independence of our method was achieved at the expense of generality, as this method is only compatible with the homogeneous polynomial kernel of degree two. Nevertheless, the wide applicability and popularity of this kernel function [7,8,20,29,30] justifies the interest of our method. We also believe that it would be possible to develop similar methods for other specific kernel functions, but this possibility will be examined in future work. In addition, the suitability of the proposed method for other machine learning tasks (e.g. regression, clustering and document retrieval) could be empirically evaluated. Finally, we believe a similar approach to the one proposed in this paper could be used to develop a data-independent form of kernelized Locality-Sensitive Hashing (LSH) [21] with the polynomial kernel, as this task is also based on the projection of samples onto random hyperplanes in the kernel feature space.

Acknowledgement

The research of Daniel López-Sánchez has been financed by the Ministry of Education, Culture and Sports of the Spanish Government (University Faculty Training (FPU) programme, reference number FPU15/02339).

References

[1] D. Achlioptas, Database-friendly random projections, in: Proceedings of the Twentieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, ACM, 2001, pp. 274–281.
[2] A. Alavi, A. Wiliem, K. Zhao, B.C. Lovell, C. Sanderson, Random projections on manifolds of symmetric positive definite matrices for image classification, in: IEEE Winter Conference on Applications of Computer Vision, IEEE, 2014, pp. 301–308.
[3] M.-F. Balcan, A. Blum, S. Vempala, Kernels as features: on kernels, margins, and low-dimensional mappings, Mach. Learn. 65 (1) (2006) 79–94.
[4] E. Bingham, H. Mannila, Random projection in dimensionality reduction: applications to image and text data, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 245–250.
[5] A. Blum, Random projection, margins, kernels, and feature-selection, in: Subspace, Latent Structure and Feature Selection, Springer, 2006, pp. 52–68.
[6] L. Bo, L. Jiao, L. Wang, Working set selection using functional gain for ls-svm, IEEE Trans. Neural Networks 18 (5) (2007) 1541–1544.
[7] Y.-W. Chang, C.-J. Hsieh, K.-W. Chang, M. Ringgaard, C.-J. Lin, Training and testing low-degree polynomial data mappings via linear svm, J. Mach. Learn. Res. 11 (April) (2010) 1471–1490.
[8] D. Chen, Z. Yuan, G. Hua, N. Zheng, J. Wang, Similarity learning on an explicit polynomial kernel feature map for person re-identification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1565–1573.
[9] A. Coates, A. Ng, H. Lee, An analysis of single-layer networks in unsupervised feature learning, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 215–223.
[10] S. Dasgupta, A. Gupta, An elementary proof of a theorem of Johnson and Lindenstrauss, Random Struct. Algo. 22 (1) (2003) 60–65.
[11] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, LIBLINEAR: a library for large linear classification, J. Mach. Learn. Res. 9 (2008) 1871–1874.
[12] M. Fanty, R. Cole, Spoken letter recognition, in: Advances in Neural Information Processing Systems, 1991, pp. 220–226.

[13] Ó. Fontenla-Romero, B. Guijarro-Berdiñas, D. Martinez-Rego, B. Pérez-Sánchez, D. Peteiro-Barral, Online machine learning, Eff. Scalabil. Methods Comput. Intell. 27 (2013).
[14] S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (10) (2009) 959–977.
[15] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Inf. Sci. (Ny) 180 (10) (2010) 2044–2064.
[16] R.V. Hogg, E. Tanis, D. Zimmerman, Probability and statistical inference, ninth ed., Pearson Higher Ed, 2014, pp. 200–202.
[17] L. Jiao, L. Bo, L. Wang, Fast sparse approximation for least squares support vector machine, IEEE Trans. Neural Networks 18 (3) (2007) 685–697.
[18] O. Kallenberg, Foundations of Modern Probability, Springer Science & Business Media, 2006.
[19] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images (2009).
[20] T. Kudo, Y. Matsumoto, Fast methods for kernel-based text analysis, in: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, vol. 1, 2003, pp. 24–31.
[21] B. Kulis, K. Grauman, Kernelized locality-sensitive hashing for scalable image search, in: 2009 IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 2130–2137.
[22] P. Li, T.J. Hastie, K.W. Church, Very sparse random projections, in: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2006, pp. 287–296.
[23] J.M. Phillips, S. Venkatasubramanian, A gentle introduction to the kernel distance, arXiv:1103.1625, (2011).
[24] J. Ratcliffe, Foundations of Hyperbolic Manifolds, vol. 149, Springer Science & Business Media, 2006.
[25] A. Shashua, Introduction to machine learning: class notes 67577, arXiv:0904.3664, (2009).
[26] C. Sun, L. Jiao, H. Liu, S. Yang, New classifier based on compressed dictionary and ls-svm, Neurocomputing 216 (2016) 617–626.
[27] L. Van Der Maaten, E. Postma, J. Van den Herik, Dimensionality reduction: a comparative, J. Mach. Learn. Res. 10 (2009) 66–71.
[28] R.E. Walpole, R.H. Myers, S.L. Myers, K. Ye, Probability and statistics for engineers and scientists, vol. 5, Macmillan New York, 1993, pp. 107–117.
[29] S. Yaman, J. Pelecanos, Using polynomial kernel support vector machines for speaker verification, IEEE Signal Process. Lett. 20 (9) (2013) 901–904.
[30] S. Yaman, J. Pelecanos, M.K. Omar, On the use of non-linear polynomial kernel svms in language recognition, in: Thirteenth Annual Conference of the International Speech Communication Association, 2012.
[31] G.-X. Yuan, C.-H. Ho, C.-J. Lin, Recent advances of large-scale linear classification, Proc. IEEE 100 (9) (2012) 2584–2603.
[32] K. Zhao, A. Alavi, A. Wiliem, B.C. Lovell, Efficient clustering on Riemannian manifolds: a kernelised random projection approach, Pattern Recognit. 51 (2016) 333–345.