Kernel Approximation via Empirical Orthogonal Decomposition for Unsupervised Feature Learning

Yusuke Mukuta
The University of Tokyo
7-3-1 Hongo Bunkyo-ku, Tokyo, Japan
mukuta@mi.t.u-tokyo.ac.jp

Tatsuya Harada
The University of Tokyo
7-3-1 Hongo Bunkyo-ku, Tokyo, Japan
harada@mi.t.u-tokyo.ac.jp

Abstract

Kernel approximation methods are important tools for various machine learning problems. There are two major methods used to approximate the kernel function: the Nyström method and the random features method. However, the Nyström method requires relatively high-complexity post-processing to calculate a solution and the random features method does not provide sufficient generalization performance. In this paper, we propose a method that has good generalization performance without high-complexity post-processing via empirical orthogonal decomposition using the probability distribution estimated from training data. We provide a bound for the approximation error of the proposed method. Our experiments show that the proposed method is better than the random features method and comparable with the Nyström method in terms of the approximation error and classification accuracy. We also show that hierarchical feature extraction using our kernel approximation demonstrates better performance than the existing methods.

1. Introduction

Analyzing data with nonlinearity is one of the main tasks in machine learning. The kernel method maps input data into a high-dimensional feature space and computes the similarity in the feature space without computing the coordinates of data in that space. The computational complexity of the kernel method is determined by the size of data, regardless of the dimension of the feature space. The kernel method is applied to the classifiers and dimensionality reduction techniques, such as the kernel support vector machine (SVM) [6], kernel principal component analysis (PCA) [23], and kernel canonical correlation analysis (CCA) [12]. However, the complexity of kernel methods grows quadratically or cubically with the amount of the training data, which makes it difficult to scale directly for large-scale datasets. A method that approximates the kernel function using the inner product of the nonlinear feature functions, which map data into a relatively low-dimensional feature space, is useful because it is compatible with fast linear classifiers.

There are two major methods for approximating the kernel function: the Nyström method [8, 25] and the random features method [20]. The Nyström method generates low-rank approximations of the Gram matrix calculated from training data. For the random features method, the kernel function is expressed as the expectation value of the inner product of feature functions, which are randomly sampled from a proper probability distribution. However, the Nyström method requires the calculation of a $D \times D$ inverse matrix at the learning phase, where D is the feature dimension, and requires $O(D^2)$ post-processing at the classifying phase. Hence, we cannot use a high-dimensional feature when using the Nyström method. To derive greater generalization ability, the random features method requires the feature to have the same number of dimensions as the number of training data, making it difficult to take advantage of the approximation.

In this paper, we propose a method to closely approximate the kernel function via empirical orthogonal decomposition without post-processing for the features. In the proposed method, the kernel function is decomposed using the probability distribution estimated from training data, which enables it to have a high approximation ability. As the proposed method directly approximates the kernel function, post-processing for the features becomes unnecessary. We show that the spectral norm of the approximation error of the Gram matrix is bounded using eigenvalues and the distance between the true and approximate distributions. We also present the calculation method of the proposed kernel approximation using the Gaussian kernel.

Kernel approximations are also used to construct a hierarchical image feature by iteratively building kernels between image patches [2, 3, 4, 5, 17]. We combine our approximation method with convolutional kernel networks

(CKN) architecture [17] and propose a novel method for unsupervised feature learning without a time-consuming optimization process.

The results of our experiments show that the proposed method is better than the random features method and comparable with the Nyström method in terms of the approximation error and classification accuracy. The proposed method for unsupervised feature learning demonstrates better or comparable accuracy with a shorter learning time than CKN. Our contributions are as follows:

- We propose a method to construct the feature functions that do not need post-processing by decomposing the kernel function using the probability distribution estimated from training data.
- We provide a bound for the approximation error of the proposed method and present a calculation method when we assume the Gaussian kernel and a Gaussian distribution.
- Experimental results on artificial and real datasets show that the proposed methods demonstrate performance that is better than the random features method and comparable with the Nyström method with lower complexity.
- Experimental evaluations of the unsupervised feature learning method show that the proposed method demonstrates better or comparable accuracy with CKN.

2. Related work

There are many kernel approximation methods; however, we briefly introduce the Nyström method and random features method because we shall focus on these methods to approximate the kernel function using the inner product of nonlinear feature functions.

2.1. Nyström method

The Nyström method approximates the true Gram matrix using kernel similarity to randomly sample data from training examples. Let $\{x_1, x_2, \cdots, x_D\}$ denote the subset of samples and $K_D = U\Lambda U^t$ denote the eigen-decomposition of the Gram matrix generated by the subset of samples, then the Nyström method maps input x in the following way:

$$\Lambda^{-1/2}U^t(k(x,x_1), k(x,x_2), \cdots, k(x,x_D))^t \tag{1}$$

To analyze Nyström methods, the bound of the spectral norm of the approximation error is usually calculated. Drineas *et al.* [8] showed that the approximation error is $O(D^{-1/2})$. Because Bartlett *et al.* [1] showed that the generalization error of the kernel method is $O(N^{-1/2})$, where N is the size of training data, the required number of samples D should be $O(N)$ to achieve a small approximation error. According to the analysis of Yang *et al.* [29], the number of samples D is reduced to $O(N^{-1/2})$ by assuming that there is a large gap between the eigenvalues. Kumar *et al.* [11] provided a detailed comparison of various fixed and adaptive sampling techniques. However, an $O(D^3)$ calculation of $K^{-1/2}$ of the sample Gram matrix is required, and $O(D^2)$ post-processing for each datum is required, which is time-consuming when D is large.

2.2. Random features method

The random features method approximates the kernel function using an inner product of randomly sampled feature functions.

Definition 2.1. *For kernel k on domain X, if there are functions f_ω parameterized by ω and parameter distribution $p(\omega)$ that fulfill the equality*

$$k(x,y) = E_\omega[f_\omega(x)^* f_\omega(y)] = \int d\omega p(\omega) f_\omega(x)^* f_\omega(y), \tag{2}$$

then a random feature is a method that samples D ω_ds i.i.d from $p(\omega)$ and maps $x \to \frac{1}{\sqrt{D}}(f_{\omega_1}(x), ..., f_{\omega_D}(x))$.

If f_ω is uniformly bounded, then we can show that we can approximate the original kernel with high probability using a sufficiently large dimension D by applying Hoeffding's inequality.

Rahimi and Recht [20] proposed a random feature using trigonometric functions for a shift-invariant kernel in Euclidean space $\mathbb{R}^d$. A shift-invariant kernel is a kernel that can be calculated using only the difference between two inputs, such as $k(x,y) = \phi(x-y)$. Rahimi and Recht [20] constructed a random Fourier feature using Bochner's theorem, which connects shift-invariant kernels with probability distributions in Fourier space.

Theorem 2.1 (Bochner [22])**.** *For ϕ corresponding to a shift-invariant kernel, there is a probability $p(\omega)$ on $\mathbb{R}^d$ that*

$$k(x,y) = \phi(x-y) = \int d\omega p(\omega) e^{i\omega(x-y)} \tag{3}$$

holds.

According to Bocher's theorem, the shift-invariant kernel is a Fourier transform of some distribution. By sampling ω_d from this distribution $p(\omega)$, the mapping

$$x \to \frac{1}{\sqrt{D}}\left(e^{i\omega_1 x}, ..., e^{i\omega_D x}\right) \tag{4}$$

approximates the original kernel. Additionally, the method that uniformly samples b_d from $[0, 2\pi]$ and maps

$$x \to \frac{1}{\sqrt{D}}\left(\sqrt{2}\cos(\omega_1 x + b_1), ..., \sqrt{2}\cos(\omega_D x + b_D)\right) \tag{5}$$

also becomes a random feature, which is used to make the feature value real. We use this form for the experiments. This feature function is uniformly bounded, so it fulfills the condition for Hoeffding's inequality.

The framework of Eq. (4) is simple and versatile, but because the feature is random, the feature tends to be verbose. To solve this problem, Hamid *et al.* [9] proposed a method that oversamples ω and projects it in a lower-dimensional space, where the projection matrix is also randomly sampled. Yang *et al.* [27] proposed a method to use quasi-Monte Carlo instead of i.i.d. random variables. To decrease the complexity, Le *et al.* [13] proposed a method to approximate a feature function with complexity $O(D \log d)$.

As an application for data mining, Lopez-Paz *et al.* [15] combined a random feature with PCA and CCA and showed that they approximate kernel PCA and kernel CCA. Lu *et al.* [16] reported performance comparable with deep learning by combining multiple kernel learning and the composition of kernels. Dai *et al.* [7] and Xie *et al.* [26] proposed a method that combined a random feature with stochastic gradient descent to construct an online learning method.

Yang *et al.* [28] proposed the random Laplace feature for the kernel $k(x, y) = \phi(x+y)$ on a semi-group $(\mathbb{R}^m > 0, +)$ and applied it to kernels on histogram data, such as bag of visual words.

However, Rahimi and Recht [21] reported that the generalization performance using a random feature is $O(N^{-1/2} + D^{-1/2})$. Thus, we need to sample $O(N)$ random features to gain sufficient generalization performance, so the complexity does not decrease.

3. Proposed method

3.1. Approximation method

The Nyström method uses information from input data as the feature function and provides good generalization performance, but requires post-processing of the feature. The random features method approximates the kernel function and does not require post-processing of the feature. The information required to obtain the feature function $p(\omega), e^{i\omega x}$ requires only the kernel function, and hence, it provides lower generalization performance. In this section, we propose a method that approximates the kernel function using information from input data to overcome the limitations of both methods.

First, from Mercer's theorem [18], we can represent the kernel k on domain X with finite measure μ as

$$k(x, y) = \sum_{i=0}^{\infty} \lambda_i \psi_i(x) \psi_i^*(y), \tag{6}$$

using eigenvalues λ_i and the normalized eigenfunctions ψ_i of the positive definite operator T_k on $L_2(X)$ such that

$$(T_k f)(\cdot) = \int_X k(\cdot, x) f(x) d\mu(x). \tag{7}$$

We can regard the Nyström method as approximating this distribution μ using the histogram of randomly sampled input data. Additionally, using a shift-invariant kernel and a Lebesgue measure, the feature corresponds to a random Fourier feature. Because the Lebesgue measure is not finite, the decomposition is an integral instead of a discrete sum; therefore we need to randomly sample the feature function.

In this paper, we propose an intermediate approach that approximates the input distribution μ using a distribution for which its eigenfunction decomposition can be solved, and use the eigenfunctions as feature functions. The algorithm is as follows:

1. Estimate the parameter of some distribution $p(x; \theta)$ using training data.
2. Solve the eigenfunction decomposition $(T_k f)(\cdot) = \int_X k(\cdot, x) f(x) p(x; \theta) dx$ using the estimated distribution $p(x; \theta)$.
3. Use $\lambda_i^{1/2} \psi_i$ corresponding to the D largest eigenvalues as feature functions.

3.2. Analysis of the approximation error

In this section, we evaluate the expectation and high-probability bound for the spectral norm of the approximation error corresponding to the Gram matrix, which is important for the efficiency of the kernel approximation method. We denote the Gram matrix using N data $\{x_1, x_2, ..., x_N\}$ by K_{true}, the Gram matrix using the proposed approximation method by K_{app}, and assume that the kernel function is upper bounded by some κ such that $k(x, x) \leq \kappa$ for $\forall x \in X$. The following holds when we use the D-dimensional feature:

Theorem 3.1. *Given the true probability density* $p_{\text{true}}(x)$ *and the approximated density as* p_{app}*, then*

$$\begin{aligned} & E_{x_i \sim p_{\text{true}}}[\|K_{\text{true}} - K_{\text{app}}\|_2] \\ & \leq N\left(\sum_{n=D}^{\infty} \lambda_n + \kappa \int_X |p_{\text{true}}(x) - p_{\text{app}}(x)| dx\right), \end{aligned} \tag{8}$$

holds. Additionally, for a probability larger than $1 - \delta$*,*

$$\begin{aligned} & \|K_{\text{true}} - K_{\text{app}}\|_2 \\ \leq \; & N\left(\sum_{n=D}^{\infty} \lambda_n + \kappa \int_X |p_{\text{true}}(x) - p_{\text{app}}(x)| dx\right) \\ + \; & \sqrt{\frac{N\kappa^2}{2} \log \frac{1}{\delta}}, \end{aligned} \tag{9}$$

holds.

Proof. It holds that $K_{\text{diff}} = K_{\text{true}} - K_{\text{app}}$ is also a Gram matrix using kernel $k(x,y) = \sum_{i=D}^{\infty} \lambda_i \psi_i(x)\psi_i^*(y)$, so K_{diff} is a symmetric positive semidefinite matrix. Note that this does not hold for a random Fourier feature, which uses an integral instead of discrete sum and does not use its eigenvalues directly. Hence,

$$\begin{aligned} \|K_{\text{true}} - K_{\text{app}}\|_2 &= \lambda_{\max}(\|K_{\text{diff}}\|) \qquad (10) \\ &\leq \text{trace}\|K_{\text{diff}}\| = \sum_{i=1}^{N} k_{\text{diff}}(x_i, x_i), \end{aligned}$$

holds. Hence, $E_{x_i \sim p_{\text{true}}}[\|K_{\text{true}} - K_{\text{app}}\|_2] \leq N E_{x \sim p_{\text{true}}}[k_{\text{diff}}(x,x)]$. Moreover,

$$\begin{aligned} & E_{x \sim p_{\text{true}}}[k_{\text{diff}}(x,x)] \\ = & \int_X (p_{\text{true}}(x) - p_{\text{app}}(x)) k_{\text{diff}}(x,x) dx \\ + & E_{x \sim p_{\text{app}}}[k_{\text{diff}}(x,x)], \qquad (11) \end{aligned}$$

The former is bounded by $\int_X |(p_{\text{true}}(x) - p_{\text{app}}(x))| |k_{\text{diff}}(x,x)| dx \leq \kappa \int_X |(p_{\text{true}}(x) - p_{\text{app}}(x))| dx$, and using the property of eigenfunction decomposition,

$$E_{x \sim p_{\text{app}}}[\psi_i(x)\psi_i^*(y)] = 1, \qquad (12)$$

the latter becomes $\sum_{n=D}^{\infty} \lambda_n$. Thus, the inequality for the expectation Eq. (8) holds.

Because $0 \leq k_{\text{diff}}(x,x) \leq \kappa$, applying Hoeffding's inequality to Eq. (11), we obtain

$$P(\sum_{i=1}^{N} k_{\text{diff}}(x_i, x_i) - N E_{x \sim p_{\text{true}}}[k_{\text{diffx}}] \geq Nt) \leq \exp\left(-\frac{2Nt^2}{\kappa^2}\right). \qquad (13)$$

Thus, a high probability bound Eq. (9) is obtained. □

From the discussion in the work by Yang *et al.* [29], we require that $\|K_{\text{true}} - K_{\text{app}}\|_2 = O(N^{1/2})$ holds for good generalization performance; that is, we require $\sum_{n=D}^{\infty} \lambda_n, \int_X |p_{\text{true}}(x) - p_{\text{app}}(x)| dx$ to be $O(N^{-1/2})$ for sufficient performance.

3.3. Gaussian case

As an example of an analytic solution for eigenfunction decomposition, we consider the Gaussian kernel and a Gaussian distribution as an approximate distribution. If the dimension $d = 1$, setting $k(x,y) = \exp(-b(x-y)^2), p(x) = N(0, \frac{1}{4a})$, and using $c = \sqrt{a^2 + 2ab}$, $A = a + b + c$, and $B = b/A$, the eigensystem is as presented by Zhu *et al.* [30]:

$$\lambda_n = \sqrt{\frac{2a}{A}} B^n \qquad (14)$$

$$\psi_n(x) = \exp(-(c-a)x^2) \mathrm{H}_n(\sqrt{2c}x), \qquad (15)$$

where H_n denotes a Hermite polynomial of integer order n and is defined as $\mathrm{H}_n(x) = (-1)^n \exp(x^2) \frac{d^n}{dx^n} \exp(-x^2)$. The feature function is localized and better reflects the properties of the Gaussian kernel, for which the similarity diminishes if the data are distant, than a random Fourier feature, which does not attenuate. Additionally, as n increases higher resolutional information can be obtained. There are studies that use this solution for kernel learning [24, 30], but to the best of our knowledge, the present research is the first to learn the distribution from data and apply it to unsupervised feature learning.

When the input dimension d is larger than 1 and the covariance matrix is diagonal, the eigensystem is a product of the above Hermite solution. Even if the covariance is non-diagonal, the solution reduces to the case in which covariance is diagonal by rotating the axis. To evaluate the approximation error, we denote the feature dimension by D and simplify the calculation by assuming a is the same for each dimension. This bounds the general case. Thus, $\sum_{n=D}^{\infty} \lambda_n = (\frac{2a}{A})^{d/2}((\frac{1}{1-B})^d - (\frac{1-B^{D/d}}{1-B})^d) \simeq (\frac{2a}{A})^{d/2}(\frac{1}{1-B})^d dB^{D/d} = dB^{D/d}$ and we can see that the error decreases exponentially with D.

3.4. Gaussian mixture case

To approximate a more complex distribution, we consider a Gaussian mixture. The analytic solution using this Gaussian mixture is not known, so we consider approximating it using the result for a Gaussian distribution. We denote the number of components by K and set $p(x) = \sum_{k=1}^{K} \gamma_k \mathcal{N}(\mu_k, \Sigma_k)$. Let (λ_n^k, ψ_n^k) be the eigensystem for $\mathcal{N}(\mu_k, \Sigma_k)$. Because the kernel can be decomposed as

$$k(x,y) = \sum_{k=1}^{K} \omega_k k(x,y) = \sum_{k=1}^{K} \sum_{n=0}^{\infty} \omega_k \lambda_n^k \psi_n^k(x) \psi_n^{k*}(y), \qquad (16)$$

we consider using $(\omega_k \lambda_n^k)^{1/2} \psi_n^k(x)$ for larger $\omega_k \lambda_n^k$ as feature functions.

Next we analyze the performance of this method. As the feature function is not the true eigenfunction, the above discussion does not hold in its current form. However, we can see that K_{diff} is a symmetric positive semidefinite matrix, and we have only to bound $E_{x \sim p}[k_{\text{diff}}(x,x)]$. To simplify this, we assume that $\omega_k = \frac{1}{K}$ and a, b are the same for each distribution and dimension, and each μ_k is well separated such that $E_{x \sim \mathcal{N}(\mu_k, \Sigma_k)}[\psi^{k'}(x)\psi^{k*}(y)] < R$ for some R. In this case, using the result from the previous section, $E_{x \sim p}[k_{\text{diff}}(x,x)] < (1 + (k-1)R) dB^{\frac{D}{dk}}$ is obtained. Thus, we infer that this method has an exponential gain in performance with D.

3.5. Relation to kernel PCA

When we apply kernel methods, we often use PCA in the projected high-dimensional space to obtain uncorrelated useful features. The proposed methods, which use eigenfunctions, are automatically projected in the feature spaces, so correlations between features are small. Thus, it is expected that our methods demonstrate a similar effect to PCA.

When we use the proposed method with Gaussian distribution, we rotate the input space so that each input element is uncorrelated. Then, the axis with a large variance has large B, so even high-order eigenfunctions with a high resolution are used. Conversely, the axis with a small variance makes a small contribution to the feature vector. In particular, the axis on which only the 0-th order eigenfunction is used only contributes to the norm of the feature, thus we can ignore it when, for example, the features are normalized. Thus, we can say that the proposed method also applies dimensionality reduction in the input space. When the input space is d dimensional, the number of substantial dimensions is d', and we extract the D dimensional feature, the complexity of the proposed method is $O(dd')$ for rotation plus an $O(d'D)$ calculation of Hermite polynomials. We replace d' with d when we use the full input vector. In all cases, it is much smaller than the Nyström method, which requires an $O(D)$ calculation of kernel values between d dimension vectors plus an $O(D^2)$ whitening step when D is large.

3.6. Application to unsupervised feature learning

We can combine the proposed method with CKN [17] architecture for unsupervised feature learning. The CKN hierarchically defines the kernel between image patches as the summation of kernels between 2-d positions of points in the patches multiplied by the kernels between feature vectors of points. The kernel value between patches Ω, Ω' can be represented as follows:

$$K(\Omega,\Omega') \\ =\sum_{z\in\Omega}\sum_{z'\in\Omega'}\|\phi(z)\|\|\phi'(z')\|e^{-\frac{1}{2\beta^2}\|\delta z\|^2}e^{-\frac{1}{2\sigma^2}\|\delta\phi\|^2}, \quad (17)$$

where z and z' are positions of the points, $\phi(z)$ and $\phi'(z')$ denote feature vectors of the points, and δz and $\delta\phi$ denote $z-z'$ and $\phi(z)-\phi'(z')$ respectively. When we approximate the kernel between positions as $\xi_{\text{pos}}(z)^T\xi_{\text{pos}}(z')$ and the kernel between feature vectors as $\xi_{\text{feat}}(\phi(z))^T\xi_{\text{feat}}(\phi'(z'))$, the convolutional kernel is approximated as the linear inner product of

$$\sum_{z\in\Omega}\|\phi(z)\|\xi_{\text{pos}}(z)\otimes\xi_{\text{feat}}(\phi(z)), \quad (18)$$

where $\otimes$ denotes the Kronecker product. CKN hierarchically applies this mapping and uses the feature vector of the final layer for recognition.

The original CKN uses the approximated feature for the kernel between positions as $\xi_{\text{pos}}(z) = e^{\frac{1}{\beta^2}\|z\|^2}$, and uses the approximated feature for the kernel between features in the form of $\eta_d^{1/2}e^{\frac{1}{\sigma^2}\|\phi(z)-w_d\|^2}$, and then learns η_d, w_d so that the reconstruction error of kernel values

$$\sum_{i=1}^{n}\left(e^{-\frac{1}{2\sigma^2}\|\delta\phi\|^2}-\sum_{d=1}^{D}\eta_d e^{\frac{1}{\sigma^2}\|\phi(z_i)-w_d\|^2}e^{\frac{1}{\sigma^2}\|\phi(z_i')-w_d\|^2}\right)^2, \quad (19)$$

is minimized, where $(z_i, z_i')_{i=1}^n$ are patch pairs sampled from training data and D is the dimension of the feature.

Instead of learning feature with a gradient descent, we can use other kernel approximation methods. We propose using eigenfunctions with distribution learned from z_i, z_i' as an approximation function. The proposed method does not experience a long optimization time and local minima.

4. Experiments

To test the efficiency of our methods, we compared the approximation error of the Gram matrices, the classification accuracy, and performance for unsupervised feature learning.

4.1. Approximation error of the Gram matrices

First, we evaluated the approximation performance using synthesized data. We set the dimensionality of the input data to $d = 10$, number of samples to $N = 5000$, and kernel parameter to $b = \frac{1}{2d}$, and compared the Nyström method (Nyström), random Fourier feature (Random), and proposed methods (Proposed) using data sampled from the Gaussian distribution with mean equal to 0 and a covariance identity matrix, from the Laplace distribution with location parameter equal to 0 and scale parameter equal to 1 as a super-Gaussian distribution, and from a uniform distribution from [-1,1] as a sub-Gaussian distribution. For each method, we evaluated the normalized spectral norm of the error matrix $\frac{\|K_{\text{true}}-K_{\text{app}}\|_2}{\|K_{\text{true}}\|_2}$. We set the feature dimension $D = 40, 160, 640, 2560$, and the number of mixture components to $1, 4, 16, 64$. Note that if the number of mixture components is 1, the situation is equivalent to the Gaussian case. To estimate the parameter of the data distribution, we used another set of N data sampled from the same distribution. For preprocessing, we rotated the data so that the estimated covariance was diagonal and used the diagonal Gaussian mixture. This rotation did not change the kernel value. We performed the experiment 10 times for each setting and calculated the mean value.

Figure 1 shows the results. The number in the "Proposed" label indicates the number of mixture components. The figure shows that for each distribution, the proposed method that assumed a Gaussian distribution yielded the best approximation performance if the dimension was low. Even if the dimension was high, the proposed method yielded a performance comparable with the

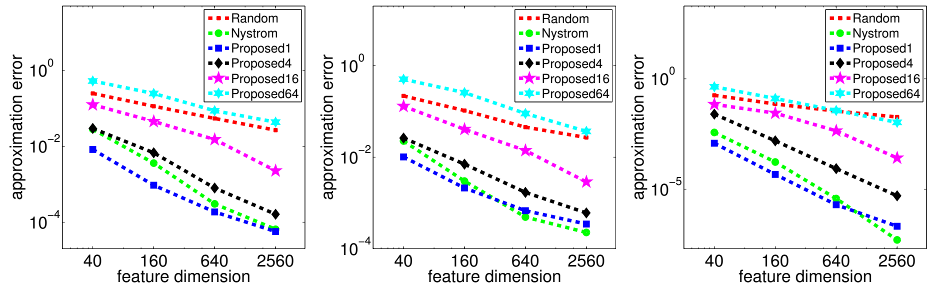

Figure 1. Comparison of the approximation error for the Gram matrix on synthesized data sampled from (**left**) a Gaussian distribution, (**center**) Laplace distribution, and (**right**) uniform distribution.

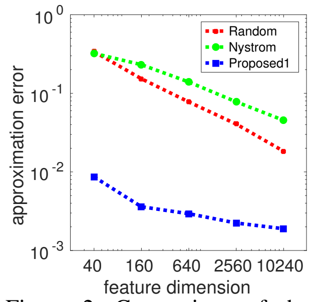

Figure 2. Comparison of the approximation error for the Gram matrix on GoogLeNet features extracted from the ILSVRC2015 dataset.

TASK	DATA	# TRAIN	# TEST	# Attr.	TASK	DATA	# TRAIN	# TEST	# Attr.
Reg.	CPUSMALL	7392	800	12	Class.	ADULT	32561	16281	123
Reg.	CADATA	18640	2000	8	Class.	IJCNN1	49990	91701	22
Reg.	YEARMSD	463715	51630	90	Class.	COVTYPE	522910	58102	54

Table 1. Statistics for the datasets.

Nyström method. Better performance for low feature dimensions occurred because the rough estimation of the proposed method approximated the true distribution better than the estimation using a small sample histogram from the Nyström method. The random Fourier feature demonstrated similar performance for each distribution, which agrees with the fact that random features method does not use distribution information. By contrast, when we assumed a Gaussian mixture, the performance was lower and the performance gain was also smaller than the case that assumed a Gaussian distribution in each case. The uncertainty associated with the parameter estimation and the decrease of decay speed of eigenvalues influenced the performance more than the approximation accuracy of the true distribution.

We then compared the performance of the proposed kernel approximation method assuming Gaussian distribution using the ILSVRC2015 classification dataset. The dataset contained approximately 1,200,000 images and we used the output of the global average pooling layer of GoogLeNet as input features, which are 1,024 dimensional per image. We used 100,000 samples for model inference and evaluated the normalized spectral norm of the error matrix $\frac{\|K_{\text{true}}-K_{\text{app}}\|_2}{\|K_{\text{true}}\|_2}$ using 5,000 randomly chosen samples. We plotted the mean of five trials. We set the feature dimension $D = 40, 160, 640, 2560, 10240$

Figure 2 shows that the proposed approximation method demonstrates better performance even for a real image dataset.

4.2. Classification Accuracy

We compared regression performance and classification accuracy using real data. We used the data from the LIBSVM site [1]. We scaled each element of the input data to [0,1] for the classification task and [-1,1] for the regression task. Table 1 shows the statistics for the datasets. We set the kernel parameter $b = \frac{1}{2d}$ and used LIBLINEAR[2] with $C = 100$ to compare the classification accuracy of the test data for classification tasks and ridge regression $\min_w \|\Psi^t w - t\|_2^2 + \lambda\|w\|_2^2$ with $\lambda = 0.01$ to compare the mean squared error of the test data $\frac{1}{n}\sum_{i=1}^{n} \|t_i - w^t\psi(x_i)\|_2^2$ for regression tasks. We set the feature dimension $D = 40, 160, 640, 2560$ for CPUSMALL, CADATA, ADULT and IJCNN1, and $40, 160, 640$ for the larger datasets YEARMSD and COVTYPE. We set the number of mixture components to $1, 4, 16$. For parameter estimation, we sampled 1000 data for CPUSMALL, CADATA, and 10000 data for ADULT, IJCNN1, YEARMSD and COVTYPE. For each setting, we conducted 10 experiments and calculated the mean.

Figure 3 and Figure 4 show the results. The result for Nyström is overlapped by that for Proposed1 in CPUSMALL. We omitted the results for Proposed4 and Proposed16 in YEARMSD because, in some cases, they had a mean squared error that was too large. The figures show that the proposed method assuming a Gaussian distribution demonstrated better performance than the random Fourier feature, especially when the dimension was small. Additionally, they demonstrated comparable performance with the Nyström method for each dimension. Because the Nyström method requires $O(D^2)$ post-processing for each feature, our method is more efficient considering the computation complexity. Generally, the proposed method as-

[1]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/
[2]https://www.csie.ntu.edu.tw/ cjlin/liblinear/

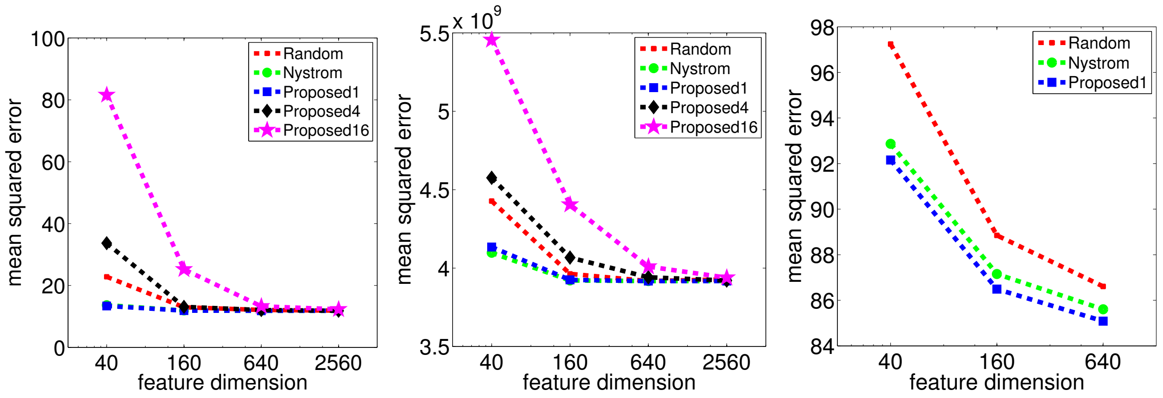

Figure 3. Comparison of the mean squared error for the datasets (**left**) CPUSMALL, (**middle**) CADATA, and (**right**) YEARMSD.

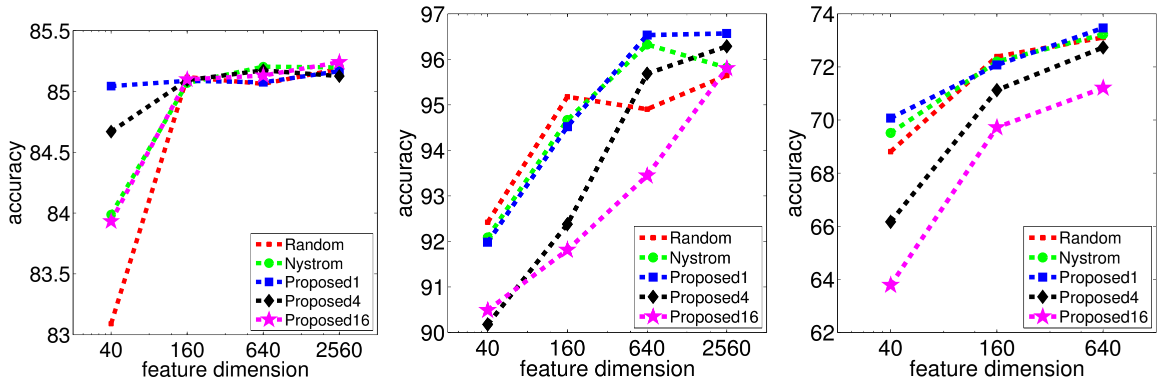

Figure 4. Comparison of the classification accuracy for the datasets (**left**) ADULT, (**middle**) IJCNN1, and (**right**) COVTYPE.

suming a Gaussian mixture demonstrated poorer performance than the other methods. However, the differences between each performance were small when the dimension was high, and Proposed16 demonstrated the best performance in ADULT. The proposed methods assuming the mixture model work well if the feature dimension is not small and the model fits the data distribution. Generally, when assuming a Gaussian distribution, the data distribution was sufficiently approximated and our method demonstrated comparable performance with the Nystöm method.

4.3. Unsupervised feature learning

Next, we used the random features method, Nyström method, and proposed method with Gaussian distribution, which demonstrated good performance in previous experiments as kernel approximation methods for CKN architecture, and compared the accuracy with the original CKN.

We used MNIST [14], CIFAR-10 [10], CIFAR-100 [10], and SVHN [19] as datasets and adopted similar network architectures to those of Mairal *et al*. [17]. We denote the detail of the architectures in the Appendix. We used 300,000 patch pairs for feature learning, LIBLINIEAR as a linear classifier and determined the regularization parameter using 5-fold crossvalidation from 2^i,$i = -15, ..., 15$.

We show the results in Table 2. In most cases, the proposed method demonstrated better performance than the original CKN. This is because the proposed method did not experience local minima of the optimization and could use the input information more efficiently. The proposed method did not require a time-consuming optimization phase, therefore it was a good choice to adopt the proposed kernel approximation method for CKN architecture. Additionally, the proposed method demonstrated better performance than the random features method and comparable or better performance than the Nyström method in most settings, which demonstrated a similar tendency to that of previous experiments. This indicates the effectiveness of hierarchically approximating the kernel, and the proposed method is reasonable and effective.

Additionally, we showed the covariance of the rescaled first 400-dimensional feature in the final layer learned from CIFAR-10 with setting 1 in Figure 5. The figure shows that while CKN had relatively large non-diagonal covariance, the proposed method demonstrated uniformly small non-diagonal covariance, which agrees with the argument that the proposed method demonstrated a similar effect to PCA.

Additionally, we varied the number of feature maps in the final layer to 200, 400, 600, 800 and evaluated the performance using CIFAR-10 with setting 1. We show the result in Figure 6. As the dimension decreased, the method using random features demonstrated poorer performance

Setting	CKN	Random	Nyström	Proposed
		MNIST		
1	99.34	**99.49**	99.38	99.36
2	99.28	99.46	**99.48**	99.43
3	99.42	99.45	99.47	**99.51**
		CIFAR-10		
1	74.59	75.93	75.72	**76.00**
2	79.19	80.73	**81.52**	81.27
3	77.41	77.68	**78.57**	78.29
		CIFAR-100		
1	43.25	43.64	43.36	**43.66**
2	53.37	**55.20**	54.44	55.01
3	50.41	51.02	50.53	**51.04**
		SVHN		
1	91.80	91.54	91.96	**91.98**
2	90.79	90.75	91.18	**91.36**
3	85.52	85.60	85.88	**86.05**

Table 2. Classification accuracy for MNIST, CIFAR-10, CIFAR-100, STL-10, and SVHN.

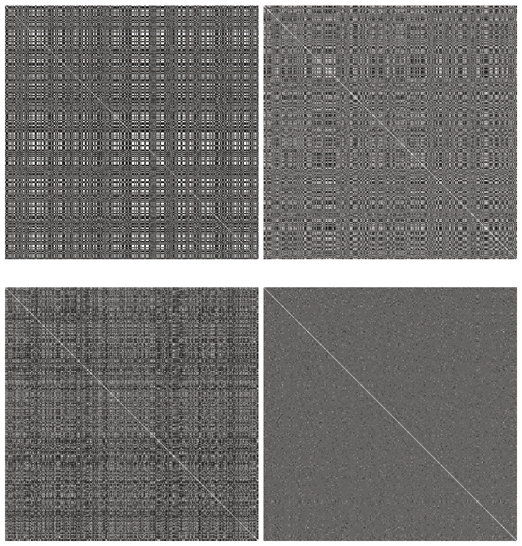

Figure 5. Covariance of learned feature for (**top left**) CKN, (**top right**) Random, (**bottom left**) Nyström, and (**bottom right**) Proposed. The proposed method shows less non-diagonal covariance.

than the proposed method and Nyström method, which illustrates the importance of using input information for approximation. Conversely, these three methods demonstrated similar performance when the dimension was 200. This suggests that when the dimension was very small, the input information was not sufficient and we needed to include discriminative information in learning. In any case, the proposed method demonstrated much better performance than the original CKN.

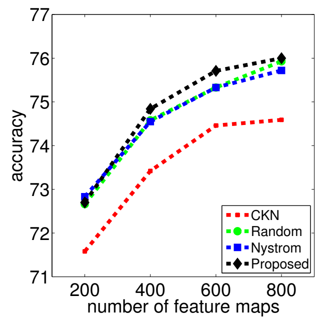

Figure 6. Accuracy of CKN with fewer feature maps.

5. Conclusion

We proposed an approximation method that first inferred the distribution of input data and then used the eigenfunctions of the operator using the kernel function and the approximated distribution as feature functions. The proposed method used information from the input data for the feature function and therefore exhibited better performance than the random features method, which only used information from the kernel. Additionally, because the proposed method directly approximated the kernel function instead of the Gram matrix, no post-processing of the feature was required. The approximation error for the proposed method can be bounded using the divergence of the true and approximated distribution and the eigenvalue of the integral operator using the kernel. We showed how to calculate the feature function for a Gaussian kernel and the approximation distribution was a Gaussian distribution or Gaussian mixture. Experiments using synthesized and real data demonstrated that our proposed method yielded a performance that was better than the random features method while comparable with the Nyström method.

There are two alternatives to expand this research. The first is to apply our method to another kernel. In the Gaussian case, we can obtain an analytic solution, but generally, the solution is not known. However, if the kernel and distribution can be decomposed into a product of functions for each dimension, we only need to consider a 1-dimensional case. Additionally, if the input dimension is high, the degree of the eigenfunction does not need to be high; hence, we can approximate the eigenfunction using, for example, series expansions. The second alternative is to approximate more accurately the eigenfunction using a mixture distribution. In this paper, we assumed that each distribution was sufficiently distant and used eigenfunctions for each distribution. However, we need to correct the original eigenfunction, for example, by adding another eigenfunction for a better approximation.

Acknowledgements

This work was supported by CREST, JST.

References

[1] P. L. Bartlett, O. Bousquet, and S. Mendelson. Local rademacher complexities. *Annals of Statistics*, 33(4):1497–1537, 2005. 2

[2] L. Bo, K. Lai, X. Ren, and D. Fox. Object recognition with hierarchical kernel descriptors. In *CVPR*, 2011. 1

[3] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *NIPS*, 2010. 1

[4] J. Bouvrie, L. Rosasco, and T. Poggio. On invariance in hierarchical models. In *NIPS*, 2009. 1

[5] Y. Cho and L. K. Saul. Large-margin classification in infinite neural networks. *Neural computation*, 22(10):2678–2697, 2010. 1

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. 1

[7] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *NIPS*, 2014. 3

[8] P. Drineas and M. W. Mahoney. On the nyström method for approximating a gram matrix for improved kernel-based learning. *JMLR*, 6(Dec):2153–2175, 2005. 1, 2

[9] R. Hamid, Y. Xiao, A. Gittens, and D. DeCoste. Compact random feature maps. In *ICML*, 2014. 3

[10] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009. 7

[11] S. Kumar, M. Mohri, and A. Talwalkar. Sampling methods for the nyström method. *JMLR*, 13(Apr):981–1006, 2012. 2

[12] P. L. Lai and C. Fyfe. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems*, 10(05):365–377, 2000. 1

[13] Q. Le, T. Sarlós, and A. Smola. Fastfood–approximating kernel expansions in loglinear time. In *ICML*, 2013. 3

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *IEEE*, 86(11):2278–2324, 1998. 7

[15] D. Lopez-Paz, S. Sra, A. Smola, Z. Ghahramani, and B. Schölkopf. Randomized nonlinear component analysis. In *ICML*, 2014. 3

[16] Z. Lu, A. May, K. Liu, A. B. Garakani, D. Guo, A. Bellet, L. Fan, M. Collins, B. Kingsbury, M. Picheny, et al. How to scale up kernel methods to be as good as deep neural nets. *arXiv preprint arXiv:1411.4000*, 2014. 3

[17] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid. Convolutional kernel networks. In *NIPS*, 2014. 1, 2, 5, 7

[18] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 209(441-458):415–446, 1909. 3

[19] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011. 7

[20] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007. 1, 2

[21] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *NIPS*, 2009. 3

[22] W. Rudin. *Fourier analysis on groups*. John Wiley & Sons, 2011. 2

[23] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput*, 10(5):1299–1319, 1998. 1

[24] C. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In *ICML*, 2000. 4

[25] C. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In *NIPS*, 2001. 1

[26] B. Xie, Y. Liang, and L. Song. Scale up nonlinear component analysis with doubly stochastic gradients. *arXiv preprint arXiv:1504.03655*, 2015. 3

[27] J. Yang, V. Sindhwani, H. Avron, and M. Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. In *ICML*, 2014. 3

[28] J. Yang, V. Sindhwani, Q. Fan, H. Avron, and M. W. Mahoney. Random laplace feature maps for semigroup kernels on histograms. In *CVPR*, 2014. 3

[29] T. Yang, Y.-F. Li, M. Mahdavi, R. Jin, and Z.-H. Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *NIPS*, 2012. 2, 4

[30] H. Zhu, C. K. Williams, R. Rohwer, and M. Morciniec. Gaussian regression and optimal finite dimensional linear models. 1997. 4