A review of approximate methods for kernel-based Big Media Data Analysis

Alexandros Iosifidis[†], Anastasios Tefas[‡], Ioannis Pitas[‡§] and Moncef Gabbouj[†]
[†]Department of Signal Processing, Tampere University of Technology, Finland
[‡]Department of Informatics, Aristotle University of Thessaloniki, Greece
[§]Department Electrical & Electronic Engineering, University of Bristol, UK
Email: {alexandros.iosifidis,moncef.gabbouj}@tut.fi, tefas@aiia.csd.auth.gr

Abstract—**With the increasing size of today's image and video data sets, standard pattern recognition approaches, like kernel based learning, need to face new challenges. Kernel-based methods require the storage and manipulation of the kernel matrix, having dimensions equal to the number of training samples. When the data set cardinality becomes large, the application of kernel methods becomes intractable. Approximate kernel-based learning approaches have been proposed in order to reduce the time and space complexities of kernel methods, while achieving satisfactory performance. In this paper, we provide a overview of such approximate kernel-based learning approaches finding application in media data analysis.**

I. Introduction

Media data analysis plays a key role in many applications, including security, human-computer interaction and human behaviour analysis for assisted living, to name a few. Specific media data analysis problems that have been heavily researched during the last two decades are facial image and action video analysis and classification [1], [2], [3], [4], [5]. For these problems, it has been shown that the use of linear modeling approaches provides inferior performance, when compared to nonlinear ones. This is why they have been primarily approached by exploiting nonlinear models, like kernel-based learning approaches [6], [7]. Such models have been widely adopted in many small- and medium-scale classification problems due to their excellent performance, theoretical foundation and easy implementation. However, in the case of big data analysis, standard kernel-based learning becomes impractical.

Challenges that need to be appropriately addressed in order to achieve practical implementations are related to the models' space and time complexities. In kernel-based learning, the so-called kernel matrix containing the dot product values between all training data pairs in the kernel space needs to be calculated and an optimization problem involving the kernel matrix needs to be solved. This process typically has space and time complexities which are quadratic and cubic with respect to the cardinality of the training set. During the test phase, one needs to store the training data[1] for dot product evaluation with a test sample, which has a linear time complexity with respect to the training data size. Thus, for problems involving media data sets available today, where the number of samples is of the order of millions, the use of standard kernel models is impractical.

[1]In kernel methods involving sparse models (like SVMs), one needs to store a subset of the training data.

In order to address these challenges approximate approaches for kernel-based learning have been proposed. In this paper, we overview such approximation approaches and discuss their application in big media data analysis. We start by describing the problem of Big Media Data analysis using kernel-based learning. Subsequently, we group the various approaches in categories depending on the type of approximation they use and we describe the basic constructive principles of each category and their connections with specific methods proposed in the literature.

II. Problem Statement

Let us denote by $\mathcal{U}$ an annotated media database formed by N samples. Depending on the application, samples forming $\mathcal{U}$ may be text, images or videos. Let us also assume that these samples have been pre-processed in order to obtain vectors $\mathbf{x}_i \in \mathbb{R}^D$, $i = 1, \ldots, N$, each representing a sample in $\mathcal{U}$ in a D-dimensional feature space preserving properties of the data which are useful for the subsequent analysis.

Standard kernel methods exploit the so-called *kernel function* $\kappa(\cdot,\cdot)$ defined on D-dimensional data pairs $\{\mathbf{x}_i, \mathbf{x}_j\}$ and expressing dot products of the data representation in the *kernel space* $\mathcal{F}$, i.e.:

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) \triangleq \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \qquad (1)$$

where $\phi(\cdot)$ is a nonlinear function mapping a D-dimensional vector to the kernel space $\mathcal{F}$:

$$\mathbf{x}_i \in \mathbb{R}^D \xrightarrow{\phi(\cdot)} \phi(\mathbf{x}_i) \in \mathcal{F}. \qquad (2)$$

The dimensionality of $\mathcal{F}$ is arbitrary and depends on the kernel function choice. For example, the kernel space dimensionality for the linear kernel function is equal to D, while for the Radial Basis Function kernel its dimensionality is infinite. After applying the nonlinear mapping in (2), one solves the original (linear) problem in $\mathcal{F}$. Linear modeling in $\mathcal{F}$ corresponds to a nonlinear model in the input space $\mathbb{R}^D$. Let us denote by $\mathbf{W}_\phi \in \mathbb{R}^{|\mathcal{F}| \times Q}$ a matrix containing parameters of the linear model learned in the kernel space $\mathcal{F}$. When $\mathbf{W}_\phi$ contains the parameters of a (multi-class or One-Versus-Rest) classification model, Q is equal to the number of classes, while when the learnt model corresponds to linear projection Q denotes the dimensionality of the sub-space.

As a concrete example, let us consider the (kernel) regression problem optimizing the following criterion:

$$\mathcal{J} = \sum_{i=1}^{N} \|\mathbf{W}_\phi^T \phi(\mathbf{x}_i) - \mathbf{t}_i\|_2^2, \tag{3}$$

where $\mathbf{t}_i \in \mathbb{R}^Q$ is the target vector for $\mathbf{x}_i$. The regression model (3) has been used for both classification and dimensionality reduction [10], [8], [9]. For example, as shown in [10], Fisher Discriminant Analysis is equivalent to a regression problem using target values $\{-1, 1\}$ and Kernel Discriminant Analysis is equivalent to a kernel regression problem using class-dependent target vectors [9]. In order to handle issues related to the arbitrary dimensionality of $\mathcal{F}$, the *kernel trick* is exploited, which states that the model parameters can be expressed as a linear combination of the training data representations in $\mathcal{F}$, i.e.:

$$\mathbf{W}_\phi = \mathbf{\Phi}\mathbf{A}, \tag{4}$$

where $\mathbf{\Phi} = [\phi(\mathbf{x}_i, \dots, \phi(\mathbf{x}_N)] \in \mathbb{R}^{|\mathcal{F}| \times N}$ and $\mathbf{A} \in \mathbb{R}^{N \times Q}$. Substituting (4) in the optimization problem to be solved ((3) in our case), we obtain:

$$\mathcal{J} = \sum_{i=1}^{N} \|\mathbf{A}^T \mathbf{\Phi}^T \phi(\mathbf{x}_i) - \mathbf{t}_i\|_2^2 = \|\mathbf{A}^T \mathbf{K} - \mathbf{T}\|_F^2, \tag{5}$$

where $\mathbf{T} = [\mathbf{t}_1, \dots, \mathbf{t}_N] \in \mathbb{R}^{Q \times N}$. $\mathbf{K} \in \mathbb{R}^{N \times N}$ is the so-called *kernel matrix* having elements equal to $[\mathbf{K}]_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$, $i = 1, \dots, N$, $j = 1, \dots, N$. In standard kernel methods $\mathbf{K}$ is restricted to be positive semi-definite, while approaches exploiting indefinite and generic similarity matrices have also been proposed [11], [12]. The solution of (5) is given by:

$$\mathbf{A} = (\mathbf{K} + \delta \mathbf{I})^{-1} \mathbf{T}^T, \tag{6}$$

where $\delta > 0$ is a regularization parameter used to scale the diagonal elements of $\mathbf{K}$ in the case of singularity. As can be seen, the solution involves the inversion of a $N \times N$ matrix which typically has a time complexity in the order of $O(N^3)$ and a space complexity of $O(N^2)$.

III. Approximation Approaches

In order to alleviate the time and space complexity of standard kernel methods, approximate approaches have been proposed. We categorize these approaches in the following categories:

- Methods exploiting low-rank a approximation of $\mathbf{K}$,
- Methods exploiting a reduced kernel space definition,
- Methods exploiting a randomized kernel space definition.

In the following, we provide a description of all three approaches and show how they can be applied for nonlinear regression.

A. *Methods exploiting low-rank a approximation of* $\mathbf{K}$

Methods belonging to this category try to determine a low-rank approximation of the kernel matrix $\mathbf{K} \in \mathbb{R}^{N \times N}$ using a rectangular matrix $\mathbf{C} \in \mathbb{R}^{N \times n}$, $n \ll N$, i.e.:

$$\mathbf{K} \approx \tilde{\mathbf{K}} = \mathbf{C}\mathbf{C}^T. \tag{7}$$

Using $\mathbf{C}$ instead of $\mathbf{K}$ both the space and time complexities of the model can be highly reduced. For example, by substituting (7) in (5) and exploiting the WoodBury formula, the approximate kernel regression solution can be obtained by:

$$\mathbf{A} \simeq \left(\tilde{\mathbf{K}} + \delta \mathbf{I}\right)^{-1} \mathbf{T}^T = \frac{1}{\delta} \left[\mathbf{I} - \mathbf{C} \left(\delta \mathbf{I} + \mathbf{C}^T \mathbf{C}\right)^{-1} \mathbf{C}^T \right] \mathbf{T}^T. \tag{8}$$

By comparing (6) and (8), we can observe that the approximate solution requires the inversion of an $n \times n$ matrix, reducing the overall time complexity of the kernel regression model from $O(N^3)$ to $O(nN^2 + n^3)$. Regarding the space requirements of the approximate solution, one needs to store an $n \times N$ matrix, thus highly reducing it in the case where $n \ll N$.

Clearly, the performance of methods exploiting a low-rank approximation of $\mathbf{K}$ highly depends on the choice of $\mathbf{C}$. One of the possible choices involves the spectral decomposition of $\mathbf{K}$. Let us denote by $\mathbf{\Lambda} \in \mathbb{R}^{N \times N}$ a (diagonal) matrix containing the eigenvalues and by $\mathbf{U} \in \mathbb{R}^{N \times N}$ matrix formed by eigenvectors of $\mathbf{K}$. Then, $\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$. In order to obtain an n-rank approximation of $\mathbf{K}$, one can keep the largest n eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and the corresponding eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_n$ in order to form the matrices $\mathbf{\Lambda}_n = diag(\lambda_1, \dots, \lambda_n)$ and $\mathbf{U}_n = [\mathbf{u}_1, \dots, \mathbf{u}_n]$. Then, $\mathbf{C} = \mathbf{U}_n \mathbf{\Lambda}_n^{\frac{1}{2}}$. While the above-described process will lead to the analysis of the approximate kernel matrix $\tilde{\mathbf{K}} = \mathbf{U}_n \mathbf{\Lambda}_n \mathbf{U}_n^T$ (which is the best approximation of $\mathbf{K}$ in terms of Frobenius norm) it requires the calculation of $\mathbf{K}$ and its eigen-decomposition.

In order to speedup the eigenvector and eigenvalue calculation process and reduce the memory requirements of this approach, an approximate Singular Value Decomposition (SVD) method has been proposed in [13]. It is based on sampling n columns of $\mathbf{K}$ according to probability values p_i in order to form a matrix $\mathbf{G} \in \mathbb{R}^{N \times n}$. The columns of $\mathbf{G}$ are scaled using a factor $\frac{1}{\sqrt{np_k}}$, where p_k corresponds to the k-th column of $\mathbf{G}$. Then, the eigenvectors and eigenvalues of $\mathbf{K}$ are approximated by the eigenvectors of the matrix $\tilde{\mathbf{G}} = \mathbf{G}^T \mathbf{G}$. Specifically, the eigenanalysis of $\tilde{\mathbf{G}}$ is applied to get the left singular vectors and the singular values of $\mathbf{G}$, i.e. $\tilde{\mathbf{G}} = \mathbf{G}^T \mathbf{G} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T$. Then, the left singular vectors of $\mathbf{G}$ (and, thus, the eigenvectors of $\tilde{\mathbf{K}}$) are obtained by $\mathbf{U} = \mathbf{G}\mathbf{V}\mathbf{\Sigma}^\dagger$, where the superscript $\cdot^\dagger$ denotes the Moore-Penrose pseudo-inverse. A special case of this process is uniform sampling, i.e. when probability values $p_i = \frac{1}{N}$ are used [19].

Another low-rank approximation matrix approach exploits the projection of the kernel matrix $\mathbf{K}$ using a random (Gaussian) matrix $\mathbf{\Omega} \in \mathbb{R}^{N \times n}$, i.e. $\mathbf{Y} = \mathbf{K}\mathbf{\Omega}$ [15]. Subsequently, orthogonal matrix decomposition is applied, i.e. $\mathbf{Y} = \mathbf{Q}\mathbf{R}$, where $\mathbf{Q} \in \mathbb{R}^{N \times n}$ is an orthogonal matrix, and $\mathbf{K}$ is mapped to the matrix $\mathbf{Q}$ by applying $\mathbf{B} = \mathbf{Q}^T\mathbf{K}\mathbf{Q} \in \mathbb{R}^{n\times n}$. Then, the matrix $\mathbf{C}$ is obtained by $\mathbf{C} = \mathbf{Q}\hat{\mathbf{U}}\mathbf{D}^{\frac{1}{2}}$, where $\hat{\mathbf{U}}$ and $\mathbf{D}$ are the eigenvectors and eigenvalues of $\mathbf{B}$, i.e. $\mathbf{B} = \hat{\mathbf{U}}\mathbf{D}\hat{\mathbf{U}}^T$. While this approach leads to reduced computational cost for the subsequent kernel-based learning process, its memory complexity is not reduced, since it requires the calculation of the entire kernel matrix $\mathbf{K}$.

Perhaps the most widely used low-rank approximation approach for kernel-based learning exploits the Nyström method, which has been originally proposed for approximating eigenfunctions. Subsequently, it has been successfully employed for approximating the eigenvalues and eigenvectors of positive semi-definite matrices for kernel-based learning [16], [17], [18]. The standard Nyström method exploits a sub-matrix of the original kernel matrix $\mathbf{K}$ in order to determine its low-rank approximation.

Let us denote by $\mathbf{K}_{Nn} \in \mathbb{R}^{N\times n}$ the sub-matrix of $\mathbf{K}$ that is formed by the columns corresponding to n samples. In addition, let us denote by $\mathbf{K}_{nn}$ the sub-matrix of $\mathbf{K}$ that is formed by the intersection of the columns and rows corresponding to the n selected samples. By applying eigen-decomposition on $\mathbf{K}_{nn}$ we obtain:

$$\mathbf{K}_{nn} = \mathbf{U}_{(n)}\mathbf{\Lambda}_{(n)}\mathbf{U}_{(n)}^T. \tag{9}$$

The matrix containing the n leading eigenvectors of $\mathbf{K}$ can be approximated by the Nyström extension by:

$$\mathbf{U}_n \approx \mathbf{K}_{Nn}\mathbf{U}_{(n)}\mathbf{\Lambda}_{(n)}^{-1}. \tag{10}$$

An n-rank approximation of $\mathbf{K}$ is given by exploiting (9) and (10):

$$\mathbf{K} \approx \tilde{\mathbf{K}} = \mathbf{K}_{Nn}\mathbf{K}_{nn}^{-1}\mathbf{K}_{Nn}^T = \mathbf{G}\mathbf{G}^T, \tag{11}$$

where $\mathbf{G} = \mathbf{K}_{Nn}\mathbf{K}_{nn}^{-\frac{1}{2}}$.

In the case where the true rank of $\mathbf{K}$ is smaller or equal to n, the approximation in (11) is exact. When this is not the case, errors occur and (10), (11) are approximations of $\mathbf{U}_n$ and $\mathbf{K}$, respectively. Regarding the selection of the matrices $\mathbf{K}_{Nn}$ and $\mathbf{K}_{nn}$, the following processes have been proposed: uniform sampling [16], [19], probabilistic sampling [20], column-norm sampling [13], adaptive sampling [19], [21], clustering-based sampling [22] and deterministic sampling [23]. Extensions of the standard Nyström method include the Density Weight Nyström method [24], the Nyström method by spectral shifting [25], the ensemble Nyström method [26] and the CUR-based Nyström method [27].

Another kernel matrix approximation approach exploits explicit feature maps, so that the dot product of the derived data representations will be approximately equal to the dot product of the data representations in the kernel space for shift-invariant kernel function [28], [29].

Low-rank approximation methods have been employed in regression-based classification, spectral clustering and manifold learning problems involving images in [16], [33], [28], [34]. As it was demonstrated in these works, the derived models can achieve satisfactory performance, while achieving considerable computational gains.

B. Methods exploiting a reduced kernel definition

Methods following a reduced kernel approach can be categorized in those solving a reduced learning problem [30], [32] and the ones exploiting a reduced kernel trick definition [35], [36], [37], [40], [38], [39].

Methods solving a reduced learning problem employ a sampling process in order to reduce the cardinality of the training set used for model creation. Let us denote by $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N] \in \mathbb{R}^{D\times N}$ a matrix formed by the vectors forming the original training set and by $\tilde{\mathbf{X}} \in \mathbb{R}^{D\times K}$ a matrix formed by the training samples selected for solving the reduced learning problem. Training data selection can be performed either by using random (or uniform) sampling or by evaluating the importance of each sample before selection, e.g. the extreme points for SVM [31]. In the case of random sampling, $\tilde{\mathbf{X}}$ can be obtained by:

$$\tilde{\mathbf{X}} = \mathbf{X}\mathbf{P}\mathbf{J}, \tag{12}$$

where $\mathbf{P} \in \mathbb{R}^{N\times N}$ is a (column) permutation matrix and $\mathbf{J} \in \mathbb{R}^{N\times K}$ is a diagonal matrix with ones. Attention should be given in sampling adequate number of samples for each class of the problem. After the determination of the reduced training set, a model is obtained by optimizing the original criterion on the smaller data set. While this approach sets the strong assumption that properties of interest between the classes forming the problem at hand are preserved, it has been shown that for several sample density functions and optimization criteria, it provides the best approximation performance [32].

Methods employing a reduced kernel trick [35], [37], [40], [38], [39] exploit an approximation of (4), i.e.:

$$\mathbf{W}_\phi = \tilde{\mathbf{\Phi}}\tilde{\mathbf{A}}, \tag{13}$$

where $\tilde{\mathbf{\Phi}} = [\phi(\mathbf{z}_1), \dots, \phi(\mathbf{z}_K)] \in \mathbb{R}^{|\mathcal{F}|\times K}$ and $\tilde{\mathbf{A}} \in \mathbb{R}^{K\times Q}$. That is, it is assumed that the model parameters can be expressed as a linear combination of reference vectors $\mathbf{z}_k$, $k = 1, \dots, K$ in $\mathcal{F}$.

Substituting (13) in (3), we obtain:

$$\mathcal{J} = \sum_{i=1}^{N} \|\tilde{\mathbf{A}}^T\tilde{\mathbf{\Phi}}^T\phi(\mathbf{x}_i) - \mathbf{t}_i\|_2^2 = \|\tilde{\mathbf{A}}^T\tilde{\mathbf{K}} - \mathbf{T}\|_F^2, \tag{14}$$

where $\tilde{\mathbf{K}} \in \mathbb{R}^{K\times N}$ is a reduced kernel matrix having elements equal to $[\tilde{\mathbf{K}}]_{ij} = \kappa(\mathbf{z}_i, \mathbf{x}_j)$, $i = 1, \dots, K$, $j = 1, \dots, N$. $\tilde{\mathbf{A}}$ is then given by:

$$\tilde{\mathbf{A}} = \left(\tilde{\mathbf{K}}\tilde{\mathbf{K}}^T\right)^{-1}\tilde{\mathbf{K}}\mathbf{T}^T, \tag{15}$$

As can be seen, the solution involves the inversion of a $K \times K$ matrix, and has a time complexity in the order of $O(K^3 + N^2)$ and a space complexity of $O(KN)$. Reference vectors can be can either be obtained by random training vector sampling [35], [37] or can be determined as the centroids obtained by applying a clustering technique, e.g. K-means, on the training data [40], [38], [39], [41]. The Nyström method has also been used in order to determine a nonlinear data mapping from the input space to a subspace of the kernel space in [42].

Methods exploiting a reduced kernel definition have been applied in facial image classification and verification problems [37], [38], [39], [41], [42], as well as on semi-supervised learning [36], [40].

C. Methods exploiting a randomized kernel definition

Methods belonging to this category exploit the connection between kernel techniques and infinite single-hidden layer (SLFN) networks [43], [44]. For SLFN networks employing a linear activation function for the network output layer, by letting the number of hidden layer neurons L go to infinity and setting a Gaussian prior to the hidden layer weights $\mathbf{v}_k \in \mathbb{R}^D$, $k = 1, ..., L$, $L \rightarrow \infty$, the evaluation of $E_{\mathbf{v}}[\mathbf{h}_i, \mathbf{h}_j]$ for all pairs of the training data representations in the feature space determined by the network's hidden layer outputs $\boldsymbol{\phi}_i$ leads to the determination of the covariance function needed to describe the SLFN network as a Gaussian process. These expectations are obtained by integrating over the relevant probability distributions of the biases and the input weights $\mathbf{v}$.

For a Gaussian prior over the distribution of $\mathbf{v}_k$ so that $\mathbf{v}_k \sim N(0, \sigma_v^2 \mathbf{I})$, the adoption of an RBF hidden layer activation function $\phi(\mathbf{x}_i, \mathbf{v}_k) = exp\left(-\frac{\|\mathbf{x}_i - \mathbf{v}_k\|_2^2}{2\sigma_g^2}\right)$ leads to a covariance function of the form:

$$\mathbf{C}(\mathbf{x}_i, \mathbf{x}_j) = s^D e^{\left(-\frac{\|\mathbf{x}_i\|_2^2}{2\sigma_m^2}\right)} e^{\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma_s^2}\right)} e^{\left(-\frac{\|\mathbf{x}_j\|_2^2}{2\sigma_m^2}\right)}, \quad (16)$$

where $s = \frac{\sigma_e}{\sigma_v}$, $\sigma_e^2 = (\sigma_v^2 + \sigma_g^2)/(\sigma_v^2 \sigma_g^2)$, $\sigma_s^2 = 2\sigma_v^2 + \sigma_g^4/\sigma_v^2$ and $\sigma_m^2 = 2\sigma_v^2 + \sigma_g^2$. If $\sigma_v^2 \rightarrow \infty$, we find that $\mathbf{C}(\mathbf{x}_i, \mathbf{x}_j) \propto exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma_s^2}\right)$, i.e. the RBF kernel function defined on the training (and test) data $\mathbf{x}_i$.

For the case of sigmoid hidden layer activation function, by making the assumption that $\mathbf{v}_k$, $k = 1, \ldots, L$ are drawn from a zero-mean Gaussian distribution with covariance matrix $\mathbf{\Sigma}$, i.e., $\mathbf{v}_k \sim N(0, \sigma_v^2 \mathbf{\Sigma})$, the corresponding covariance function is given by [44]:

$$\mathbf{C}(\mathbf{x}_i, \mathbf{x}_j) = \frac{2}{\pi} sin^{-1} \frac{\tilde{\mathbf{x}}_i^T \mathbf{\Sigma} \tilde{\mathbf{x}}_j}{\sqrt{\left(1 + \tilde{\mathbf{x}}_i^T \mathbf{\Sigma} \tilde{\mathbf{x}}_i\right)\left(1 + \tilde{\mathbf{x}}_j^T \mathbf{\Sigma} \tilde{\mathbf{x}}_j\right)}}, \quad (17)$$

where $\tilde{\mathbf{x}}_i$ is the augmented input vector $\tilde{\mathbf{x}}_i = [1, \mathbf{x}_i^T]^T$.

Exploiting this connection between the $\mathbf{K}$ and the covariance matrix of infinite SLFN networks, an approximation of the $\mathbf{K}$ can be obtained by randomly sampling $\mathbf{v}_k, k = 1, \ldots, L$ vectors in $\mathbb{R}^D$ according to a multi-dimensional distribution (e.g Gaussian). That is, let us denote by $h(\cdot, \cdot)$ an activation function, like the RBF and sigmoid functions, defined over the input data $\mathbf{x}_i$ and the network's hidden layer weights $\mathbf{v}_k$. Let us also denote by $\mathbf{H} \in \mathbb{R}^{L \times N}$ a matrix having elements $[\mathbf{H}]_{ki} = h(\mathbf{v}_k, \mathbf{x}_i)$, $k = 1, \ldots, L$, $i = 1, \ldots, N$. Then, $\mathbf{K} \approx \mathbf{H}^T \mathbf{H}$ [45]. Thus, one can exploit the training vectors $\mathbf{h}_i \in \mathbb{R}^L$, $i = 1, \ldots, N$ in order to train a linear model, which corresponds to a nonlinear model in the input space $\mathbb{R}^D$.

For example, the nonlinear regression problem can be defined over the data representations $\mathbf{h}_i$ as follows:

$$\mathcal{J} = \sum_{i=1}^{N} \|\mathbf{W}_h^T \mathbf{h}_i - \mathbf{t}_i\|_2^2, \quad (18)$$

where $\mathbf{W}_h \in \mathbb{R}^{L \times Q}$ is a matrix containing the parameters of the linear model in $\mathbb{R}^L$. $\mathbf{W}_h$ is obtained by:

$$\mathbf{W}_h = \left(\mathbf{H}\mathbf{H}^T\right)^{-1} \mathbf{H}\mathbf{T}^T. \quad (19)$$

As can be seen, the solution involves the inversion of a $L \times L$ matrix, and has a time complexity in the order of $O(L^3 + N^2)$ and a space complexity of $O(LN)$.

The above-described approach has been exploited in Extreme Learning Machine-based learning models for (semi-)supervised learning [46], [47], [50], [51], [48], [49] and dimensionality reduction [52], as well as similar approaches, like the Random Kitchen Sinks classifier [53]. Such methods have found application in generic and facial image classification, human action recognition [51], [54], [55].

IV. Conclusions

In this paper, we provided an overview of approximate methods for kernel-based learning using large-scale data sets. We categorized the existing approaches based on the type of approximation they exploit in three categories, i.e., methods adopting a low-rank kernel matrix assumption, methods exploiting a reduced kernel space definition and methods exploiting a randomized kernel space definition. We described the basic principles of each category and provided connections to specific methods applied in media data analysis problems.

References

[1] P. Barr, J. Noble and R. Biddle, *Video game values: Human-computer interaction and games*, Interacting with Computers, vol. 19, no. 2, pp. 180-195, 2007.

[2] Z. Li, U. Park and A. Jain, *A discriminative model for age invatiant face recognition*, IEEE Transactions on Information Forensics and Security, vol. 6, no. 3, pp. 1028-1037, 2011.

[3] A. Tefas and I. Pitas, *Human centered interfaces for assisted living*, International Conference on Man-Machine Interraction, 2011.

[4] A. Iosifidis, E. Marami, A. Tefas, I. Pitas and K. Lyroudia, *The MOBISERV-AIIA Eating and Drinking multi-view database for vision-based assisted living*, Journal of Information Hiding and Multimedia Signal Processing, vol. 6, no. 2, pp. 254-273, 2015.

[5] A. Iosifidis, A. Tefas and I. Pitas, *Class-specific reference discriminant analysis with application in human behavior analysis*, IEEE Transactions on Human-Machine Systems, vol. 45, no. 3, pp. 315326, 2015.

[6] V. Vapnik, *Statistical Learning Theory*, Wiley-Interscience, 1998.

[7] B. Scholkopf and A. Smola, *Learning with Kernels*, MIT Press, 2001.

[8] J.A.K. Suykens, T. Van Gestel, J. De Brabantter, B. De Moor and J. Vandewalle, *Least Squares Support Vector Machines*, World Scientific, 2002.

[9] D. Cai, X. He and J. Han, *SRDA: An efficient algorithm for large-scale Discriminant Analysis*, IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 1, pp. 112, 2008.

[10] R. Duda, P. Hart and D. Stork, *Pattern Classification*, 2nd ed. Wiley-Interscience, 2000.

[11] M. Balcan, A. Blum andN. Srebro, *A theory of learning with similarity functions*, Machine Learning, vol. 72, pp. 89-112, 2008.

[12] F. Schleif and P. Tino, *Indefinite proximity learning: A review*, Neural Computation, vol. 27, pp. 2039-2096, 2015.

[13] P. Drineas, R. Kannan and M.W. Mahoney, *Fast Mont Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix*, SIAM Journal on Computing, vol. 36, pp. 158-183, 2011.

[14] S. Kumar, M. Mohri and A. Talwalkar, *On sampling-based approximate spectral decomposition*, International Conference on Machine Learning, 2009.

[15] N. Halko, P.G. Martinsson and J.A. Tropp, *Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Reviews, vol. 53, pp. 217-288, 2011.

[16] C. Williams and M. Seeger, *Using the Nyström method to speed up kernel machines*, Advances on Neural Information Processing Systems, vol. 13, pp. 682-688, 2001.

[17] S. Sun, J. Zhao and J. Zhu, *A review of Nyström methods for large-scale machine learning*, Information Fusion, vol. 26, pp. 36-48, 2015.

[18] A. Iosifidis and M. Gabbouj, *Nyström-based Approximate Kernel Subspace Learning*, Pattern Recognition, vol. 57, pp. 190-197, 2016.

[19] S. Kumar, M. Mohri and A. Talwalkar, *On sampling-based approximate spectral decomposition*, Internationa Conference

[20] P. Drineas and M.W. Mahoney, *On the Nyström method for approximating a Gram matrix for improved kernel-based learning*, Journal on Machine Learning Research, vol. 6, pp. 2153-2175, 2005.

[21] A. Deshpante, L. Rademacher, S. Vempala and G. Wang, *Matrix approximation and projective clustering via volume sampling*, ACM-SIAM Symposium on Discete Algorithm, 2006.

[22] K. Zhang, I.W. Tsang and J.T. Kwok, *Improved Nyström low-rank approximation and error analysis*, International Conference on Machine Learnin, 2008.

[23] M.A. Belabbas and P.J. Wolf, *Spectral methods in machine learning and new strategies for very large datasets*, Proceedings of the National Academy of Sciences, pp. 369-374, 2006.

[24] K. Zhang and J.T. Kwok, *Density-weighted Nyström methods for computing large kernel eigensystems*, Neural Computation, vol. 21, pp. 121-146, 2009.

[25] Z. Zhang, *The matrix ridge approximation: algorithms and applications*, Machine Learning, vol. 97, pp. 227-258, 2014.

[26] S. Kumar, M. Mohri and A. Talwalkar, *Ensemble Nyström method*, Advances on Neural Information Processing Systems, 2009.

[27] S. Wang and Z. Zhang, *Improving CUR matrix decomposition and Nyström approximation via adaptive sampling*, Journal of Machine Learning Research, vol. 14, pp. 2729-2769, 2013.

[28] A. Rahimi and B. Recht, *Random features for large-scale kernel machines*, Advances on Neural Information Processing Systems, 2007.

[29] A. Vedaldi and A. Zisserman, *Efficient Additive Kernels via Explicit Feature Maps*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 3, pp. 480-492, 2012

[30] A. El Alaoui and M. Mahoney, *Fast randomized kernel methods with statistical guarantees*, CoRR,abs/1411.0306, 2014

[31] M. Nandan, P.P. Khargonekar and S. S. Talathi, *Fast svm training using approximate extreme points*, Journal of Machine Learning Research, vol. 15, no. 1, pp. 5998, 2014.

[32] N. Cesa-Bianchi, Y. Mansour and O. Shamir, *On the Complexity of Learning with Kernels*, Journal of Machine Learning Research, vol. 40, no. 1, pp. 1-29, 2015.

[33] C. Fowlkes, S. Belongie, F. Chung and J. Malik, *Spectral grouping using the Nyström method*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 26, vol. 2, pp. 214-225, 2004.

[34] A. Talwalkar, S. Kumar and H. Rowley, *Large-scale manifold learning*, IEEE Conference on Computer Vision and Pattern Recognition, 2008.

[35] Y. Lee and S. Huang, *Reduced Support Vector Machines: A statistical theory*, IEEE Transactions on Neural Networks, vol. 18, pp. 1-13, 2007.

[36] K. Zhang, J.T. Kwok and B. Parvin, *Prototype Vector Machine for Large Scale Semi-Supervised Learning*, International Conference on Machine Learning, 2009.

[37] A. Iosifidis, A. Tefas and I. Pitas, *Large-scale nonlinear facial image classification based on approximate kernel extreme learning machine*, IEEE International Conference on Image Processing, 2015.

[38] A. Iosifidis, M. Gabbouj and P. Pekki, *Class-specific nonlinear projections using class-specific kernel spaces*, IEEE International Conference on Big Data Science and Engineering, 2015.

[39] V. Mygdalis, A. Iosifidis, A. Tefas and I. Pitas, *Large-scale classification by an Approximate Least Squares One-Class Support Vector Machine ensemble*, IEEE International Conference on Big Data Science and Engineering, 2015.

[40] K. Zhang, L. Lan, J. Kwok, S. Vucetic, and B. Parvin, *Scaling up graph-based semisupervised learning via Prototype Vector Machines*, IEEE Transactions on Neural Networks and Learning Systems, vol. 26, no. 3, pp. 444-457, 2015.

[41] A. Iosifidis, A. Tefas and I. Pitas, *Regularized Extreme Learning Machine for large-scale media content analysis*, INNS Conference on Big Data, 2015.

[42] A. Iosifidis and M. Gabbouj, *On the kernel Extreme Learning Machine speedup*, Pattern Recognition Letters, vol. 68, pp. 205-210, 2015.

[43] R.M. Neal, *Bayesian Learning for Neural Networks*, Lecture Notes in Statistics, 1996.

[44] C.K.I. Williams, *Computation with Infinite Neural Networks*, Neural Computation, vol. 10, no. 5, pp. 1203-1216, 1998.

[45] A. Iosifidis, A. Tefas and I. Pitas, *On the Kernel Extreme Learning Machine Classifier*, Pattern Recognition Letters, vol. 54, pp. 11-17, 2015.

[46] G.B. Huang, Q.Y. Zhu and C.K. Siew, *Extreme learning machine: a new learning scheme of feedforward neural networks*, IEEE International Joint Conference on Neural Networks, 2004.

[47] G.B. Huang, H, Zhou, X. Ding and R. Zhang, *Extreme learning machine for regression and multiclass classification*, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 42, no. 2, pp. 513-529, 2012.

[48] X. Liu, S. Lin, J. Fang and Z. Xu, *Is extreme learning machine feasible? A theoretical assessment (Part I)*, IEEE Transactions on Neural Networks and Learning Systems vol. 26, no. 1, pp. 720, 2015.

[49] S. Lin, X. Liu, J. Fang and Z. Xu, *Is extreme learning machine feasible? A theoretical assessment (Part II)*, IEEE Transactions on Neural Networks and Learning Systems vol. 26, no. 1, pp. 2134, 2015.

[50] G. Huang, S. Song, J.N.D. Gupta and C. Wu, *Semi-supervised and Unsupervised Extreme Learning Machines*, IEEE Transactions on Cybernetics, vol. 44, no. 12, pp. 2405-2417, 2014.

[51] A. Iosifidis, A. Tefas and I. Pitas, *Regularized Extreme Learning Machine for Multi-view Semi-supervised Action Recognition*, Neurocomputing, vol. 145, pp. 250-262, 2014.

[52] A. Iosifidis, *Extreme Learning Machine based Supervised Subspace Learning*, Neurocomputing, vol. 167, pp. 158-164, 2015.

[53] A. Rahimi and B. Recht, *Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning*, Advances on Neural Information Processing Systems, 2008.

[54] A. Iosifidis, A. Tefas and I. Pitas, *Minimum Class Variance Extreme Learning Machine for Human Action Recognition*, IEEE Transactions on Circuits and Systems for Video Technology, vol. 23, no. 11, pp. 1968-1979, 2013.

[55] A. Iosifidis, A. Tefas and I. Pitas, *Graph Embedded Extreme Learning Machine*, IEEE Transactions on Cybernetics, vol. 46, no. 1, pp. 311-324, 2016.