Kernel Matrix Completion for Learning Nearly Consensus Support Vector Machines

Sangkyun Lee(✉) and Christian Pölitz

Fakultät für Informatik, LS VIII, Technische Universität Dortmund,
44221 Dortmund, Germany
{sangkyun.lee,christian.poelitz}@tu-dortmund.de

Abstract. When feature measurements are stored in a distributed fashion, such as in sensor networks, learning a support vector machine (SVM) with a full kernel built with accessing all features can be pricey due to required communication. If we build an individual SVM for each subset of features stored locally, then the SVMs may behave quite differently, being unable to capture global trends. However, it is possible to make the individual SVMs behave nearly the same, using a simple yet effective idea we propose in this paper. Our approach makes use of two kernel matrices in each node of a network, a local kernel matrix built with only locally stored features, and an estimate of remote information (about "local" kernels stored in the other nodes). Using matrix completion, remote information is fully recovered with high probability from a small set of sampled entries. Due to symmetric construction, each node will be equipped with nearly identical kernel matrices, and therefore individually trained SVMs on these matrices are expected to have good consensus. Experiments showed that such SVMs trained with relatively small numbers of sampled remote kernel entries have competent prediction performance to full models.

Keywords: Support vector machines · Kernel methods · Matrix completion · Multiple kernel learning · Distributed features

1 Introduction

Training the support vector machines (SVMs) [2] in distributed environments has been an interesting topic in machine learning research. Considering distributed storage of data, the topic can be largely divided into two categories depending on whether examples or features are distributed. When examples are distributed (in this case, features are usually not assumed to be distributed), an SVM can be trained using a distributed optimization algorithm [3] but incurring potentially a good amount of communication, or individual SVMs can be trained locally with extra constraints to reduce their disparity to other SVM models in a network [7] with less amount of information exchange. Alternatively, local SVMs can be trained completely independently on data partitions and then combined to produce a more stable and accurate model than individual ones [6,10].

A. Fred et al. (Eds.): ICPRAM 2014, LNCS 9443, pp. 93–109, 2015.
DOI: 10.1007/978-3-319-25530-9_7

On the other hand, when features are distributed (examples are not distributed) and they are not agglomerated for analysis, learning machineries have to deal with a set of partitioned feature spaces, figuring out how they can accurately predict global trends overall feature spaces. Although this scenario has potential use for emerging applications such as sensor network [14], it has not been studied much in machine learning research for obvious difficulties. This paper focuses on this scenario and proposes a simple but effective method for training SVMs for distributed features, with a major difference to the existing methods [12,20] that no central coordination will be involved in learning. Part of this paper has been published in a conference [11], and this paper extends the previous one with updated results on projection error and matrix completion, and new discussions on classification error bounds of using approximate kernels.

Figure 1 shows a sketch of our proposed method. Each local feature storage node (represented as a circle) creates a local kernel matrix using only locally stored features. It also creates an empty matrix to store remote information, for which some entries are sampled from kernels stored in the other nodes (built with only features stored in them). The unseen entries of this partially observed remote kernel matrix are recovered via matrix completion, where the recovery is perfect with high probability if samples are noise-free. Then, each node makes use of a combination of the two matrices, a local and a recovered remote kernels, to train an SVM predictor as if the node has seen all features over a network.

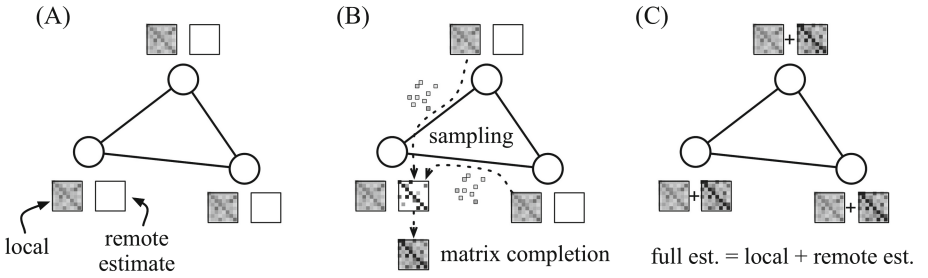

Fig. 1. A sketch of the proposed method. (A) each local feature storage node (a circle) create a kernel matrix for local features, and a null estimate of remote information, (B) each node obtains sampled entries of "local" kernels in the other nodes, and then recovers unseen entries by matrix completion, (C) each node makes use of its local kernel matrix together with recovered remote information to create its own SVM predictor.

In the following, we first introduce two decompositions of kernel matrices that enable us to approximate an original full-feature kernel matrix with separated kernels corresponding to local and remote features. Then we discuss the idea of matrix completion, a new observation of support vectors, and generalization error bounds of our method. Our description focuses on SVM classifiers, however it can be generalized to other kernel-based methods. We denote the Euclidean norm of vectors by $\|\cdot\|$ and the cardinality of a finite set A by $|A|$ throughout the paper.

2 Decomposition of Kernel Matrices

Consider local feature storages represented as nodes $n = 1, 2, \ldots, N$ in a network, where each node stores its features in a vector $\mathbf{x}_i[n]$ of length p_n. Here i is an index for examples, $i = 1, 2, \ldots, m$, and we assume that all nodes can observe the same examples (but through different set of features) and their label $y_1, \ldots, y_m$. For simplicity, we allow for communication between any pair of nodes. Then the collection of all features can be written as a single vector $\mathbf{x}_i = (\mathbf{x}_i[1]^T, \mathbf{x}_i[2]^T, \ldots, \mathbf{x}_i[N]^T)^T$ of length $p := \sum_{n=1}^{N} p_n$ (this vector is never created in our method).

2.1 Support Vector Machines

The dual formulation of the SVM, in a form where a bias term is augmented in a weight vector, is described as follows [13],

$$\min_{\mathbf{0} \le \boldsymbol{\alpha} \le C\mathbf{1}} \quad \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{Q}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha}, \tag{1}$$

where $\boldsymbol{\alpha}$ is a vector of length m, $\mathbf{1} := (1, 1, \ldots, 1)^T$, $\mathbf{y} := (y_1, y_2, \ldots, y_m)^T$, and $C > 0$ is a given constant. The $m \times m$ matrix $\mathbf{Q}$ is a scaled kernel matrix, that is, $\mathbf{Q} := \mathbf{YKY}$ for a positive semidefinite kernel matrix $\mathbf{K}$, where $\mathbf{Y} := \mathrm{diag}(\mathbf{y})$ is the diagonal matrix with labels from $\mathbf{y}$. A typical SVM is built with $\mathbf{K}_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are "full" feature vectors and $\phi : \Re^p \to \mathcal{H}$ is a map from the space of feature vectors to a Hilbert space [18].

2.2 Schur and MKL Kernels

Our goal is to build an SVM for each local feature storage node in a network, as if we have accessed all features but without explicitly doing so. This becomes possible by the observations we introduce here, about how to decompose the original kernel matrices into parts corresponding to local and remote features.

The construction is surprisingly simple. For instance, the expression of the Gaussian kernel [18] can be rewritten as follows,

$$\begin{aligned} \text{(Schur Kernel)} \quad [\mathbf{K}]_{ij} &= e^{-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|} = \prod_{n=1}^{N} e^{-\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2} \\ &= \underbrace{e^{-\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2}}_{[\mathbf{S}_n]_{ij}\ :\ \text{local for node } n} \quad \underbrace{\prod_{n' \neq n} e^{-\gamma\|\mathbf{x}_i[n'] - \mathbf{x}_j[n']\|^2}}_{[\mathbf{S}_{-n}]_{ij}\ :\ \text{remote for node } n}. \end{aligned} \tag{2}$$

Here $\gamma > 0$ is a given parameter. As we can see, the Gaussian kernel matrix can be decomposed into a matrix $\mathbf{S}_n$ created using only the features stored in a node n, and another matrix $\mathbf{S}_{-n}$ that captures information about features not in the node n. Since this kernel matrix is written as an elementwise product of

matrices, known as the Schur (or Hadamard) product, we rename this kernel as the Schur kernel to distinguish itself from local Gaussian kernels such as $\mathbf{S}_n$.

We also consider another type of kernels constructed by a simple linear combination of multiple local Gaussian kernels. This kernel typically appears in the multiple kernel learning [9] scenarios, so we name it as the MKL kernel. This kernel is obviously decomposable:

$$\begin{aligned}(\text{MKL Kernel})\ [\mathbf{K}]_{ij} &= \frac{1}{N}\sum_{n=1}^{N} e^{-\gamma_n \|\mathbf{x}_i[n]-\mathbf{x}_j[n]\|^2} \\ &= \frac{1}{N}\underbrace{e^{-\gamma_n \|\mathbf{x}_i[n]-\mathbf{x}_j[n]\|^2}}_{[\mathbf{M}_n]_{ij}\text{: local for node } n} + \frac{1}{N}\underbrace{\sum_{n'\neq n} e^{-\gamma_{n'} \|\mathbf{x}_i[n']-\mathbf{x}_j[n']\|^2}}_{[\mathbf{M}_{-n}]_{ij}\text{: remote for node } n}. \end{aligned} \tag{3}$$

Similarly to the previous case, this kernel is decomposed into a local ($\mathbf{M}_n$) and a remote $\mathbf{M}_{-n}$ parts. Compared to the Schur kernel, the MKL kernel requires to specify N positive kernel parameters $\gamma_1, \gamma_2, \ldots, \gamma_N$, rather than a single parameter $\gamma > 0$. Despite of this potential inconvenience, MKL kernels have an advantage that we can reduce the size of kernel matrices and therefore speed up the process of matrix completion as discussed later in Sect. 3.3.

3 Recovery of Unseen Kernel Elements

In our model, each node n creates a matrix $\mathbf{S}_n$ or $\mathbf{M}_n$ depending on the type of kernel it use (Schur or MKL), from only locally stored features. Also, each node n creates an empty matrix for $\mathbf{S}_{-n}$ or $\mathbf{M}_{-n}$, in which each element is just a product (Schur) or a sum (MKL) of "local" kernel matrices stored in the other nodes, where only a few of these entries are observed by the node n by uniform random sampling.

3.1 Sampling

The key elements in our sampling strategy are that (i) the size of sample should be minimized to reduce communication cost, and at the same time (ii) the sample size should be large enough to guarantee the recovery of unseen entries with accuracy. As we see later, the theory of matrix completion make it possible to achieve both goals, surprisingly enough, with simple uniform random sampling.

For a node n, we denote the index pair set of observed entries of remote kernel matrices ("local" in the other nodes) by

$$\Omega_n \subset \{(i,j) : 1 \le i,j \le m\},$$

where the pairs (i, j) are chosen uniformly at random[1]. Then, all the other nodes n' transfer the entries of their local kernel matrix corresponding to Ω_n, i.e.,

$$\text{Node } n \longleftarrow \left\{ \begin{array}{ll} [\mathbf{S}_{n'}]_{\Omega_n} & \text{(Schur)} \\ [\mathbf{M}_{n'}]_{\Omega_n} & \text{(MKL)} \end{array} \right\} \longleftarrow \text{Node } n'.$$

This requires that the sample index pair set Ω_n should be known to all nodes[2], which can be done once by exchanging such information when a pier recognizes other piers in a network. Or, we can fix all sets Ω_n for $n = 1, 2, \ldots, N$ the same, so that no exchange will be necessary if the set is pre-determined.

Given Ω_n, the communication cost of this type of transfer will be $\mathcal{O}((N-1)|\Omega|)$, if a node n is connected to all the other nodes. As discussed later, matrix completion requires the size Ω_n relatively small, $\mathcal{O}(m \log^6 m)$, to guarantee a perfect recovery in high probability.

When a node n receives the information, it stores the entries to the corresponding storage as follows (left: Schur, right: MKL):

$$[\mathbf{S}_{-n}]_{\Omega_n} = \prod_{n' \neq n} [\mathbf{S}_{n'}]_{\Omega_n}, \qquad [\mathbf{M}_{-n}]_{\Omega_n} = \sum_{n' \neq n} [\mathbf{M}_{n'}]_{\Omega_n}.$$

The next step is to recover the unobserved entries in the matrix $\mathbf{S}_{-n}$ or $\mathbf{M}_{-n}$.

3.2 Low-Rank Matrix Completion with Kernel Constraints

To recover the full matrix of $\mathbf{S}_{-n}$ or $\mathbf{M}_{-n}$ from few observed entries indexed by Ω_n, we use the low-rank matrix completion [17]. Some modifications are required, however, to deal with the constraints that $\mathbf{S}_{-n}$ or $\mathbf{M}_{-n}$ should be a valid kernel matrix. In particular, $\mathbf{S}_{-n}$ and $\frac{1}{N-1}\mathbf{M}_{-n}$ must be symmetric and positive semi-definite matrices to satisfy the Mercer's theorem [18].

To describe a formal framework for matrix completion, suppose that $\mathbf{X} \in \Re^{m \times m}$ is a matrix we wish to recover, and that the observed entries of $\mathbf{X}$ (corresponding to a sample index pair set Ω_n) are stored in a data matrix $\mathbf{D}$ so that $\mathbf{D}_{\Omega_n} = [\mathbf{S}_{-n}]_{\Omega_n}$ for Schur kernels or $\mathbf{D} = [\mathbf{M}_{-n}]_{\Omega_n}$ for MKL kernels. Matrix completion recovers the full matrix of $\mathbf{X}$ as a solution of the following convex optimization problem [17],

$$\min_{\mathbf{X} \in \Re^{m \times m}} \sum_{(i,j) \in \Omega_n} (\mathbf{X}_{ij} - \mathbf{D}_{ij})^2 + \lambda \|\mathbf{X}\|_*, \quad \|\mathbf{X}\|_* := \sum_{k=1}^{m} \sigma_k(\mathbf{X}),$$

where $\|\mathbf{X}\|_*$ is called the *nuclear norm* of $\mathbf{X}$, which is the summation of singular values $\sigma_k(\mathbf{X})$ of $\mathbf{X}$ and penalizes the rank of $\mathbf{X}$ in effect. When the rank of $\mathbf{X}$ is r, then the nuclear norm simplifies to the expression [16,17],

$$\|\mathbf{X}\|_* = \min_{\mathbf{X} = \mathbf{L}\mathbf{R}^T} \frac{1}{2}\|\mathbf{L}\|_F^2 + \frac{1}{2}\|\mathbf{R}\|_F^2,$$

[1] We require Ω_n to be symmetric, that is, if $(i, j) \in \Omega_n$ then $(j, i) \in \Omega_n$, and also the values corresponding to the entries are the same.

[2] Not actual values, but the positions to be sampled from.

where $\mathbf{L}$ and $\mathbf{R}$ are $m \times r$ matrices, and $\|A\|_F := \left(\sum_{ij} A_{ij}^2\right)^{1/2}$ is the Frobenius norm of a matrix A. Using this property, the above optimization can be rewritten for rank-r matrix completion,

$$\min_{\mathbf{L},\mathbf{R}\in\Re^{m\times r}} \sum_{(i,j)\in\Omega_n} (\mathbf{L}_{i\cdot}\mathbf{R}_{j\cdot}^T - \mathbf{D}_{ij})^2 + \frac{\lambda}{2}\|\mathbf{L}\|_F^2 + \frac{\lambda}{2}\|\mathbf{R}\|_F^2. \tag{4}$$

A solution of this optimization can be obtained using the JELLYFISH algorithm [17] for example, which is a highly parallel incremental gradient descent algorithm using the fact that the gradient of the objective function depends on row vectors $\mathbf{L}_{i\cdot}$ and $\mathbf{R}_{j\cdot}$ and therefore updates of iterates can be distributed for the pairs (i, j) in Ω_n.

Projection to the Mercer Kernel Space. The outcome $\mathbf{X}^* = \mathbf{L}^*(\mathbf{R}^*)^T$ of low-rank matrix completion (4) may not be a valid Mercer kernel matrix. To make it a valid one, we project $\mathbf{X}^*$ to the space of Mercer kernels [18], which is a convex set $\mathcal{K}$ of symmetric positive semidefinite rank-r matrices in our case,

$$\mathcal{K} := \{\mathbf{X} : \mathbf{X} \succeq 0,\ \mathbf{X}^T = \mathbf{X},\ \mathrm{rank}(\mathbf{X}) = r\} = \{\mathbf{Z}\mathbf{Z}^T : \mathbf{Z} \in \Re^{m\times r}\}.$$

Here $\mathbf{X} \succeq 0$ means that $\mathbf{X}$ is a positive semi-definite matrix which satisfies $\mathbf{w}^T\mathbf{X}\mathbf{w} \geq 0$ for any $\mathbf{w} \in \Re^m$. The last equality implies that an element $\mathbf{X} \in \mathcal{K}$ must have the form $\mathbf{Z}\mathbf{Z}^T$ for an $m \times r$ matrix $\mathbf{Z}$, which can be easily verified using the eigen-decomposition [21] of symmetric $\mathbf{X}$.

To project $\mathbf{X}^* = \mathbf{L}^*(\mathbf{R}^*)^T$ onto $\mathcal{K}$, we use a simple projection for which a closed-form solution exists (derivable from the optimality conditions),

$$\mathbf{Z}^* = \operatorname*{arg\,min}_{\mathbf{Z}\in\Re^{m\times r}} \frac{1}{2}\|\mathbf{Z} - \mathbf{L}^*\|_F^2 + \frac{1}{2}\|\mathbf{Z} - \mathbf{R}^*\|_F^2, \quad \mathbf{Z}^* = \frac{\mathbf{L}^* + \mathbf{R}^*}{2}. \tag{5}$$

The next result shows that the gap between a recovered matrix from matrix completion $\mathbf{L}^*(\mathbf{R}^*)^T$ and a valid kernel matrix obtained after projection $\mathbf{Z}^*(\mathbf{Z}^*)^T$ is bounded and becomes small whenever $\mathbf{L}^* \approx \mathbf{R}^*$, which is indeed likely to happen since we require that the sample index pair set Ω is symmetric.

Lemma 1. *The distance between* $\mathbf{Z}^*(\mathbf{Z}^*)^T$ *and* $\mathbf{L}^*(\mathbf{R}^*)^T$ *is bounded as follows,*

$$\|\mathbf{Z}^*(\mathbf{Z}^*)^T - \mathbf{L}^*(\mathbf{R}^*)^T\|_F \leq \frac{\|\mathbf{L}^* - \mathbf{R}^*\|_F}{4} + \frac{\|\mathbf{L}^*(\mathbf{R}^*)^T - \mathbf{R}^*(\mathbf{L}^*)^T\|_F}{2}.$$

Proof. The result follows from the definition of $\mathbf{Z}^*$ and the triangle inequality.

After training SVMs, we apply the same technique for new test examples to build a test kernel matrix. This usually involves a smaller matrix completion problem corresponding to the support vectors and test examples.

3.3 Reduction with MKL Kernels Using Support Vectors

The matrix completion optimization (4) for recovering full kernel matrices involves m^2 variables, and therefore it requires more computational resource for larger m. It turns out that we can work on a smaller number of variables than m^2, using a property we discovered for *support vectors* (SVs) in case of the MKL kernel. To remind, a support vector is an example indexed by $i \in \{1, 2, \ldots, m\}$ for which the optimal solution α_i^* of the SVM problem (1) is strictly positive.

Let us consider a "full-information" SVM problem with the MKL kernel built with accessing all features, and its set of SVs S^*,

$$\boldsymbol{\alpha}^* := \mathop{\arg\min}_{0 \le \boldsymbol{\alpha} \le C\mathbf{1}} \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{Y} \left[\frac{1}{N}\sum_{n=1}^{N} \mathbf{M}_n\right] \mathbf{Y}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha}, \quad S^* := \{i : 1 \le i \le m, [\boldsymbol{\alpha}^*]_i > 0\}.$$

And, we also consider the corresponding "local" SVM problems and their SV sets for each node n, $n = 1, 2, \ldots, N$,

$$\boldsymbol{\alpha}_n^* := \mathop{\arg\min}_{0 \le \boldsymbol{\alpha} \le C\mathbf{1}} \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{Y}[\mathbf{M}_n]\mathbf{Y}\boldsymbol{\alpha} - \mathbf{1}^T\boldsymbol{\alpha}, \quad S_n^* := \{i : 1 \le i \le m, [\boldsymbol{\alpha}_n^*]_i > 0\}.$$

The number of SVs, $|S^*|$, is often much smaller than the total number of examples m, and it is well known that an SVM predictor is fully determined by the SVs [18]. Therefore, if we estimate S^* without too much cost, then we can focus on variables corresponding to S^* for recovery in matrix completion rather than considering all m^2 variables. Our next theorem shows that an estimation of S^* is possible without solving the full-information problem, simply by the union of local SV sets (the proof is in Appendix).

Theorem 1. *The support vector sets of the full-information and local SVM problems above with the MKL kernel satisfy*

$$S^* \subseteq \bigcup_{n=1}^{N} S_n^*.$$

Using the theorem, matrix completion (4) can be solved more efficiently with $|\cup_n S_n^*|^2$ variables. In our experiments, the number was much smaller than m^2: the ratio $|\cup_n S_n^*|^2/m^2$ was in the range of $0.31 \sim 0.98$, where in the half of the cases the ratio was below 0.5.

4 Matrix Recovery and Classification Error Bounds

To decide the size of the sample index set Ω, it is important to understand when a (perfect) recovery of unseen matrix elements is possible from only a few observed entries. It is also closely related how well an SVM trained with a recovered kernel matrix will perform in classification, compared to the case of using full-information kernel matrices.

4.1 Conditions for Matrix Recovery

The technique of matrix completion guarantees the perfect recovery of a partially observed matrix with high probability, under a certain condition called the *strong incoherence property* [4].

To describe this property, we define a parameter $\mu > 0$ as follows. Suppose that we try to recover all unseen entries of a matrix $\mathbf{D} \in \Re^{m \times m}$ from a few observed entries $\mathbf{D}_{ij}$, $(i,j) \in \Omega$. The rank of $\mathbf{D}$ is assumed to be r, so that the reduced singular value decomposition of $\mathbf{D}$ can be written as,

$$\mathbf{D} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T, \;\; \mathbf{U}^T\mathbf{U} = \mathbf{I}, \;\; \mathbf{V}^T\mathbf{V} = \mathbf{I},$$

where $\mathbf{U} \in \Re^{m \times r}$ and $\mathbf{V} \in \Re^{m \times r}$ contain the left and right singular vectors of $\mathbf{D}$ that constitute orthonormal bases of the range space of $\mathbf{D}$ and $\mathbf{D}^T$, respectively (if $\mathbf{D}$ is symmetric then $\mathbf{U} = \mathbf{V}$), and $\boldsymbol{\Sigma} \in \Re^{r \times r}$ is a diagonal matrix with singular values. Then the orthogonal projection onto the range space of $\mathbf{D}$ and $\mathbf{D}^T$ can be described as $\mathcal{P}_{\mathbf{U}} = \mathbf{U}\mathbf{U}^T$ and $\mathcal{P}_{\mathbf{V}} = \mathbf{V}\mathbf{V}^T$, resp. Using these, we define two parameters $\mu_1 > 0$ and $\mu_2 > 0$ such that

$$\max_{i,j} \max \left\{ \left| [\mathcal{P}_{\mathbf{U}}]_{ij} - \frac{r}{m}\chi_{i=j} \right|, \;\; \left| [\mathcal{P}_{\mathbf{V}}]_{ij} - \frac{r}{m}\chi_{i=j} \right| \right\} \leq \mu_1 \frac{\sqrt{r}}{m} \tag{6}$$

where $\chi_{i=j}$ is an indicator function returning 1 if $i = j$ and 0 otherwise, and,

$$\max_{i,j} |[\mathbf{U}\mathbf{V}^T]_{ij}| \leq \mu_2 \frac{\sqrt{r}}{m}. \tag{7}$$

Finally, the parameter $\mu > 0$ is defined simply as

$$\mu := \max\{\mu_1, \mu_2\},$$

and we say $\mathbf{D}$ satisfies the strong incoherence property when μ is small (i.e., $\mu = \mathcal{O}(\sqrt{\log m})$). Conceptually speaking, small values of μ indicate that observing an element in $\mathbf{D}$ provides good information about the other elements in the same row and column, since its information is "dissolved" into many components of singular vectors and so are the other elements.

The next theorem states that when the rank r or the incoherence parameter μ of a matrix $\mathbf{D}$ we want to recover is small, then exact recovery via matrix completion (4) is possible with high probability with only a small number of observed entries.

Theorem 2 (Candès and Plan [5]). *Let $\mathbf{D} \in \Re^{m \times m}$ a matrix with rank $r \in (0, m]$ and a strong incoherence parameter $\mu > 0$. If the number of observed entries from $\mathbf{D}$ satisfies*

$$|\Omega| \geq C\mu^2 mr \log^6 m$$

for some constants $C > 0$, then the minimizer of the matrix completion problem (4) is unique and equal to the original full matrix $\mathbf{D}$ with probability at least $1 - m^{-3}$.

4.2 Classification Error Bounds for Using Estimated Kernels

When $|\Omega|$ is sufficiently large, then two factors can contribute to differences between the original and the estimated kernel matrices: (i) the noise in observed entries corresponding to Ω (note that Theorem 2 is for noise-free cases) and (ii) the gap induced by a projection onto the cone of symmetric positive semidefinite matrices (Lemma 1). Let us define the recovery error ϵ due to noise for elements corresponding to Ω as follows,

$$\epsilon := \sum_{(i,j)\in\Omega_n} (\mathbf{L}^*_{i\cdot}(\mathbf{R}^*_{j\cdot})^T - \mathbf{D}_{ij})^2.$$

Using this, the gap between an original kernel matrix $\mathbf{K}$ and an estimated one $\widetilde{\mathbf{K}}$ can be stated as follows (for the case of the MKL kernel),

$$\Delta := \|\mathbf{K} - \widetilde{\mathbf{K}}\|_F \leq \underbrace{\|\mathbf{K} - \mathbf{L}^*(\mathbf{R}^*)^T\|_F}_{=\delta_1} + \underbrace{\|\mathbf{L}^*(\mathbf{R}^*)^T - \mathbf{Z}^*(\mathbf{Z}^*)^T\|_F}_{=\delta_2}, \tag{8}$$

where the two terms on the right are bounded respectively as follows,

$$\delta_1 \leq 2\epsilon \left[2\sqrt{(1 + 2m^2/|\Omega|)\, m} + 1\right], \qquad \text{(due to [5, Theorem 7])}$$

$$\delta_2 \leq \|\mathbf{L}^* - \mathbf{R}^*\|_F/4 + \|\mathbf{L}^*(\mathbf{R}^*)^T - \mathbf{R}^*(\mathbf{L}^*)^T\|_F/2 \qquad \text{(due to Lemma 1).}$$

For simplicity we do not discuss the case of Schur kernels here since the first term in the right-hand side of (8) becomes quite complicated.

To show a classification error bound using an estimate kernel matrix $\widetilde{\mathbf{K}}$, we use a result from Nguyen et al. [15] that if a kernel $\mathbf{K}$ is universal on the input space [19] as the Gaussian kernels, the *error coefficient* $\mathcal{E}_{\mathbf{K}}$ approaches the Bayes error in probability, that is,

$$\mathcal{E}_{\mathbf{K}} = \frac{1}{m}\mathbf{1}^T\boldsymbol{\alpha}^*_{\mathbf{K}} \;\rightarrow\; \inf_{f\in\mathcal{F}} \mathbb{P}(y \neq f(\mathbf{x})),$$

where $\boldsymbol{\alpha}^*_{\mathbf{K}}$ is the optimal solution of an SVM (1) using $\mathbf{K}$, and $\mathcal{F}$ is an arbitrary family of measurable functions that contains the optimal Bayes classifier. This implies that the classification error of using an estimate kernel matrix $\widetilde{\mathbf{K}}$ can be characterized by the distance between $\mathcal{E}_{\widetilde{\mathbf{K}}}$ and $\mathcal{E}_{\mathbf{K}}$, as we state in the next theorem (the proof is in Appendix):

Theorem 3. *Suppose that $\widetilde{\mathbf{K}}$ is an MKL (3) kernel matrix constructed with remote kernel information recovered by matrix completion (4) with a sufficiently large sample index set Ω satisfying the condition of Theorem 2 and projection (5), where errors identified by δ_1 and δ_2 as above. Then, with very high probability, the classification error of using $\widetilde{\mathbf{K}}$ instead of the full-information MKL kernel $\mathbf{K}$ is bounded by*

$$|\mathcal{E}_{\widetilde{\mathbf{K}}} - \mathcal{E}_{\mathbf{K}}| \leq \frac{C}{\lambda_1(\mathbf{Q})}(\delta_1 + \delta_2),$$

where C is a parameter for the SVM, and $\lambda_1(\mathbf{Q})$ is the smallest eigenvalue of $\mathbf{Q} = \mathbf{YKY}$, $\mathbf{Y} = diag(\mathbf{y})$.

Note that when observations are made without noise, then $\delta_1 = 0$ with probability at least $1 - m^{-3}$ by Theorem 2. Also, δ_2 is expected to be small due to our symmetry construction of Ω, in which case $\mathbf{L}^* \approx \mathbf{R}^*$. Moreover, we often specify $C = C'/m$ for some $C' > 0$, and therefore if $1/\lambda_1(Q) = o(m)$, that is, $\lim_{m\to\infty} \frac{o(m)}{m} = 0$, then the term $C/\lambda_1(\mathbf{Q})$ becomes small as m increases.

5 Experiments

For experiments, we used five benchmark data sets from the UCI machine learning repository [1], summarized in Table 1, and also their subset composed of 5000 training and 5000 test examples (denoted by 5k/5k) to study characteristics of algorithms under various circumstances.

We have implemented our algorithm as open-source in C++, based on the JELLYFISH code[3] [17] for matrix completion, and SVMLIGHT[4] [8] for solving SVMs. Our implementation makes use of the union SVs set theorem (Theorem 1) for the MKL approach to reduce kernel completion time, but not for Schur since the theorem does not apply for this case.

Table 1. Data sets and their training parameters. Different values of C were used for the full data sets (column C) and smaller $5k/5k$ sets (column C ($5k/5k$)).

Name	m (train)	Test	p	C	C ($5k/5k$)	γ
ADULT	40701	8141	124	10	10	0.001
MNIST	58100	11900	784	0.1	1162	0.01
CCAT	89702	11574	47237	100	156	1.0
IJCNN	113352	28339	22	1	2200	1.0
COVTYPE	464809	116203	54	10	10	1.0

For all experiments, we split the original input feature vectors into subvectors of almost equal lengths, one for each node of $N = 3$ nodes (for 5k/5k sets) and $N = 10$ nodes (for full data sets). The tuning parameters C and γ were determined by cross validation for the full sets, and the C values for the 5k/5k subsets were determined by independent validation subsets, both with SVMLIGHT. The results of SVMLIGHT were included for a comparison to a non-distributed SVM training. Following [12], the local Gaussian kernel parameters for MKL were adjusted to $\gamma_n = \frac{p}{p_n} \approx N\gamma$ for a given γ, so that $\gamma_n \|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|$ will have the same order of magnitude $\mathcal{O}(\gamma p)$ as $\gamma\|\mathbf{x}_i - \mathbf{x}_j\|$.

[3] Available for download at http://hazy.cs.wisc.edu/hazy/victor/jellyfish/.
[4] Available at http://svmlight.joachims.org/.

5.1 Characteristics of Kernel Matrices

The first set of experiments is to verify that how well kernel matrices fit for matrix completion. For this, we computed the two types of full kernel matrices, Schur (2) and MKL (3), accessing all features of the small 5k/5k subsets of the five UCI data sets (in this case the Schur kernel is simply the Gaussian kernel).

The important characteristics of the kernel matrices with respect to matrix completion are its rank (r) and coherence parameters μ_1 and μ_2 defined in (6) and (7). When these values are small, then Theorem 2 tells that we only need small number of observations for perfect matrix completion with high probability.

Table 2 summarizes these characteristics. Clearly, the rank (numerically effective rank, with eigenvalues larger than a threshold of 0.01) and coherence values were small in cases of `ADULT`, `IJCNN`, and `COVTYPE` compared to the other data sets, and the conditions were improved when the MKL kernel was used compared to the Schur kernel. So it was expected that matrix completion would perform better for these three data sets compared to the rest, `MNIST` and `CCAT`, for a fixed size of the sample index set $|\Omega|$.

Table 2. The density, rank r, and coherence parameters μ_1 and μ_2 defined in (6) and (7) of kernel matrices. Effective numbers of ranks are shown, which correspond to eigenvalues larger than a threshold (0.01). Smaller rank and coherence parameters are better for matrix completion.

	Schur				MKL			
	Density	r	μ_1	μ_2	Density	r	μ_2	μ_2
`ADULT`	1.0	789	25.7	24.4	1.0	222	12.0	5.5
`MNIST`	1.0	4782	68.1	68.5	1.0	4568	66.6	66.2
`CCAT`	1.0	4984	69.6	70.6	1.0	4982	69.6	70.6
`IJCNN`	1.0	1516	37.2	37.8	1.0	698	25.1	6.6
`COVTYPE`	1.0	1423	35.9	35.8	1.0	424	19.3	2.7

5.2 The Effect of Sampling Size

Next, we have used the 5k/5k data sets to investigate how the prediction performance of SVMs changed over several difference sizes of the sample index set Ω. We define the sampling ratio as

$$\text{Sampling Ratio} := |\Omega|/(m^2),$$

where the value of m is 5000 in this experiment. We compared the prediction performance of using Schur and MKL to that of SVMLIGHT.

Figure 2 illustrates the test accuracy values for five sampling ratios in up to 10 %. The statistics are over $N = 3$ nodes and over random selections of Ω. The performance on `ADULT`, `IJCNN`, and `COVTYPE` was close to that of SVMLIGHT,

Table 3. Test prediction performance on full data sets (mean and standard deviation). Two sampling ratios (2 % and 10 %) are tried for our method. The SVMLIGHT results are from using the classical Gaussian kernels with matching parameters. $|\cup_n S_n^*|^2/m^2$ is the fraction of the reduced number of variables compared to m^2.

	MKL			ASSET	SVMLIGHT
	$\lvert\cup_n S_n^*\rvert^2/m^2$	2 %	10 %		
ADULT	0.37	81.4±1.00	84.2±0.18	80.0±0.02	84.9
MNIST	0.98	78.9±1.69	87.0±0.20	88.9±0.39	98.9
CCAT	0.71	87.2±1.00	92.0±0.35	73.7±1.00	95.8
IJCNN	0.31	96.0±0.35	96.5±0.23	90.9±0.88	99.3

and it kept increasing with the growth of $|\Omega|$. This behavior was expected in the previous section as their kernel matrices had good conditions for matrix completion. On the other hand, the performance on MNIST and CCAT was far inferior to that of SVMLIGHT, as also expected.

The bottom-right corner of Fig. 2 shows the concentration of eigenvalue spectrum in the five kernel matrices. The height of each box represents the magnitude of the corresponding normalized eigenvalue, so that the height a stack of boxes represents the proportion of entire spectrum concentrated in the top 10 eigenvalues. The plot shows that 90 % of the spectrum in ADULT is concentrated in the top 10 eigenvalues, indicating that its kernel matrix has a very small numerically effective rank. This gives one explanation why our method performs as good as SVMLIGHT for the case of ADULT.

Comparing Schur to MKL, both showed similar prediction performance. However, higher concentration of the eigen spectrum of MKL indicated that it would make a good alternative to Schur, also considering the extra saving with MKL discussed in Sect. 3.3.

5.3 Performance on Full Data Sets

In the last experiment, we used the full data sets for comparing our method to one of the closely related approaches, ASSET [12]. Since ASSET admits only MKL-type kernels, we have omitted the Schur kernel in comparison. Among the several versions of ASSET in [12], we used the "Separate" version with central optimization. COVTYPE was excluded due to runtime issues of SVMLIGHT.

The results are in Table 3. The second column shows the ratio between a union SV set and an entire training set. The square of these numbers indicates the saving we have achieved by the union SVs trick, for example the size of matrix is reduced to 37 % of the original size for ADULT. The saving was substantial for ADULT and IJCNN. In terms of prediction performance, we have achieved test accuracy approaching to that of SVMLIGHT (within 1 % point (ADULT), 3.8 % points (CCAT), and 2.8 % points (IJCNN) on average) with 10 % sampling ratio, except for the case of MNIST where the gap was significantly larger (11.9 %): this

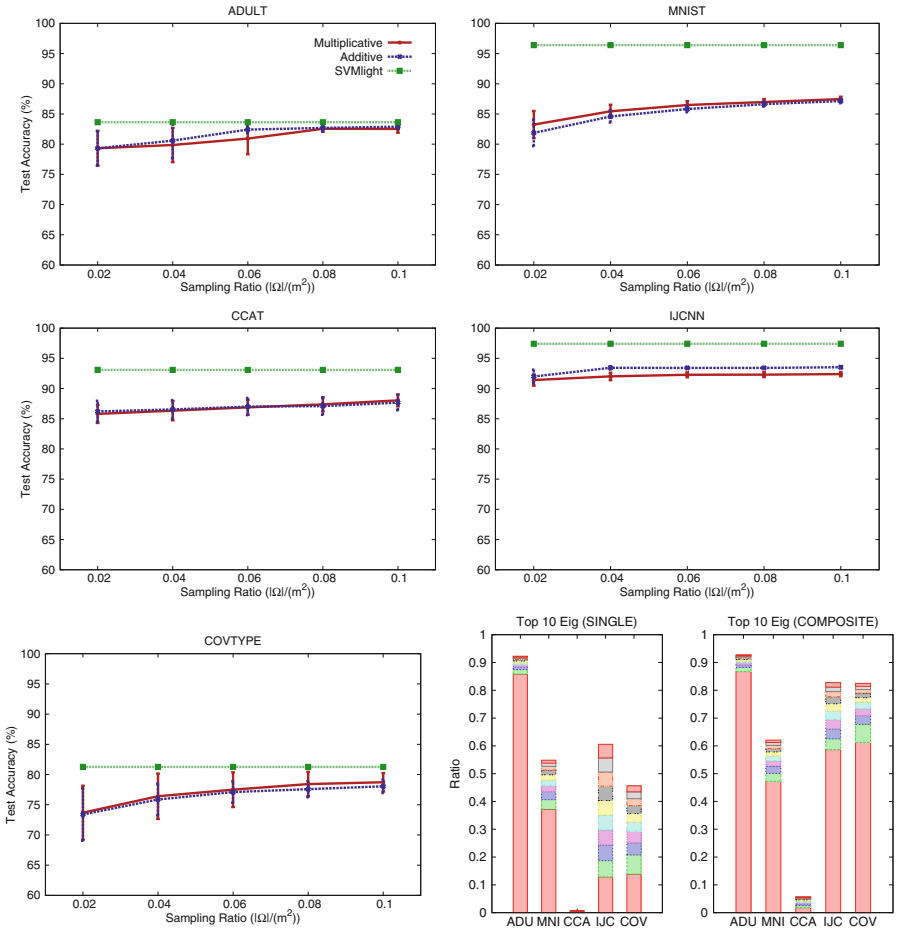

Fig. 2. Prediction accuracy on test sets for 5k/5k subsets of the five UCI data sets, over different sampling ratios in kernel completion. The average and standard deviation over multiple trials with random Ω and $N = 3$ nodes are shown. The bottom-right plot illustrates the proportion of the entire eigen-spectrum concentrated in the top ten eigenvalues.

result was consistent to the discussion in Sects. 5.1 and 5.2. Our method (with 10 % sampling) also outperformed ASSET (by 4.2 %, 18.3 %, and 5.6 % on average for `ADULT`, `CCAT`, and `IJCNN` respectively) except for the case of `MNIST` with a small but not negligible margin (1.9 %). We conjecture that the approximation of kernel mapping in ASSET have fitted particularly well for `MNIST`, but it remains to be investigated.

6 Conclusions

We have proposed a simple framework for learning nearly consensus SVMs for scenarios where features are stored in a distributed manner. Our method makes use of decompositions of kernels, together with kernel matrix completion to recover unobserved entries of remote kernel matrices. The resulting SVMs performed well with relatively small numbers of sampled entries, but under certain conditions. A newly discovered property of support vectors also helped us further reduce computation cost in matrix completion.

Several aspects of our method remain to be investigated further. First, different types of kernels may involve different types of decomposition, with new characteristics in terms of matrix completion. Second, although parameters of SVMs and kernels can be tuned using small aggregated data, it would be desirable to tune parameters locally, or to consider entirely parameter-free alternatives if possible. Also, despite the benefits of the MKL kernel, it requires more kernel parameters to be specified compared to the Schur kernel. Therefore when the budget for parameter tuning is limited, Schur would be preferred to MKL. Finally, it would be worthwhile to analyze the characteristics of the suggested algorithm in real communication systems to make it more practical, considering non-uniform communication cost on non-symmetric networks.

Acknowledgements. The authors acknowledge the support of Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", projects A1 and C1.

Appendix

Proof of Theorem 1

Let us consider an index $i \in S^*$ of an SV of the global SVM problem, such that $[\boldsymbol{\alpha}^*]_i > 0$. Suppose that the ith component of the gradient of all local SVM problems at $\boldsymbol{\alpha}^*$ is strictly positive, that is,

$$[\mathbf{Y}\mathbf{M}_n\mathbf{Y}\boldsymbol{\alpha}^* - \mathbf{1}]_i > 0, \quad \forall n \in \{1, 2, \ldots, N\}. \tag{9}$$

Let us look into the optimality condition of the global SVM, regarding the ith component of the optimizer $\boldsymbol{\alpha}^*$. From the KKT conditions, we have

$$\frac{1}{N}\sum_{n=1}^{N}[\mathbf{Y}\mathbf{M}_n\mathbf{Y}\boldsymbol{\alpha}^* - \mathbf{1}]_i - [\mathbf{p}^*]_i + [\mathbf{q}^*]_i = 0, \quad [\mathbf{p}^*]_i[\boldsymbol{\alpha}^*]_i = 0, \quad [\mathbf{q}^*]_i[C\mathbf{1} - \boldsymbol{\alpha}^*]_i = 0,$$

where $\mathbf{p}^* \in \Re_+^m$ and $\mathbf{q}^* \in \Re_+^m$ are the Lagrange multipliers for the constraints $\boldsymbol{\alpha} \geq 0$ and $\boldsymbol{\alpha} \leq C\mathbf{1}$, respectively. Then $[\boldsymbol{\alpha}^*]_i > 0$ implies $[\mathbf{p}^*]_i = 0$, and therefore

$$\frac{1}{N}\sum_{n=1}^{N}[\mathbf{Y}\mathbf{M}_n\mathbf{Y}\boldsymbol{\alpha}^* - \mathbf{1}]_i + [\mathbf{q}^*]_i = 0.$$

If (9) is true, then we have a contradiction here since the first term above becomes strictly positive, where the second term satisfies $[\mathbf{q}^*]_i \geq 0$, and therefore the equality cannot hold. This implies that there exists at least one node n for which the condition in (9) is not satisfied, that is, $[\mathbf{Y}\mathbf{M}_n\mathbf{Y}\boldsymbol{\alpha}^* - \mathbf{1}]_i \leq 0$. This means that if we search for the local SVM solution at the node n starting from $\boldsymbol{\alpha}^*$, we must increase the value of the ith component from $[\boldsymbol{\alpha}^*]_i$ to reach the minimizer $[\boldsymbol{\alpha}_n^*]_i$ of this local SVM problem, since otherwise we will increase the objective function value. That is,

$$[\boldsymbol{\alpha}_n^*]_i \geq [\boldsymbol{\alpha}^*]_i > 0.$$

This implies that the index i also becomes an SV of at least one local SVM problem. Therefore, $i \in \cup_{n=1}^N S_n^*$, which implies the claim.

Proof of Theorem 3

From the definition of the error coefficient and the Cauchy-Schwarz inequality, we have

$$|\mathcal{E}_{\widetilde{\mathbf{K}}} - \mathcal{E}_{\mathbf{K}}| = \frac{1}{m}\mathbf{1}^T(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*) \leq \frac{1}{\sqrt{m}}\|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*\|_2. \tag{10}$$

On the other hand, from the optimality conditions of (1) with $\mathbf{K}$ and $\widetilde{\mathbf{K}}$,

$$(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T(\mathbf{Q}\boldsymbol{\alpha}_{\mathbf{K}}^* - \mathbf{1}) \geq \mathbf{0}, \quad (\boldsymbol{\alpha}_{\mathbf{K}}^* - \boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^*)^T(\widetilde{\mathbf{Q}}\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \mathbf{1}) \geq \mathbf{0},$$

and adding them up gives

$$(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T(\mathbf{Q}\boldsymbol{\alpha}_{\mathbf{K}}^* - \widetilde{\mathbf{Q}}\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^*) \geq 0.$$

This implies that (the idea is from [15]),

$$(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T(\mathbf{Q} - \widetilde{\mathbf{Q}})\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* \geq (\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T\mathbf{Q}(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*).$$

From $(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T(\mathbf{Q} - \widetilde{\mathbf{Q}})\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* \leq \|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*\|_2\|\mathbf{Q} - \widetilde{\mathbf{Q}}\|_2\|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^*\|_2$ (Cauchy-Schwarz and the definition of operator norms) and $(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*)^T\mathbf{Q}(\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*) \geq \lambda_1(\mathbf{Q})\|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*\|_2^2$ (properties of the Rayleigh quotient; $\lambda_1(\mathbf{Q})$ is the smallest eigenvalue of $\mathbf{Q}$), the above inequality leads to (with $\|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^*\|_2 \leq C\sqrt{m}$),

$$\|\boldsymbol{\alpha}_{\widetilde{\mathbf{K}}}^* - \boldsymbol{\alpha}_{\mathbf{K}}^*\|_2 \leq \frac{C\sqrt{m}}{\lambda_1(\mathbf{Q})}\|\mathbf{Q} - \widetilde{\mathbf{Q}}\|_2.$$

From (10), the facts that $\mathbf{Q} = \mathbf{Y}\mathbf{K}\mathbf{Y}$ for $\mathbf{Y} = \mathrm{diag}(\mathbf{y})$, $\|\mathbf{A}\mathbf{B}\|_2 \leq \|\mathbf{A}\|_2\|\mathbf{B}\|_2$, and $\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_F$, and finally by (8), we have the claim,

$$|\mathcal{E}_{\widetilde{\mathbf{K}}} - \mathcal{E}_{\mathbf{K}}| \leq \frac{C}{\lambda_1(\mathbf{Q})}\|\mathbf{Q} - \widetilde{\mathbf{Q}}\|_2 = \frac{C}{\lambda_1(\mathbf{Q})}\|\mathbf{K} - \widetilde{\mathbf{K}}\|_F \leq \frac{C}{\lambda_1(\mathbf{Q})}(\delta_1 + \delta_2).$$

References

1. Bache, K., Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
2. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152 (1992)
3. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. **3**(1), 1–122 (2011)
4. Candes, E.J., Tao, T.: The power of convex relaxation: near-optimal matrix completion. IEEE Trans. Inf. Theor. **56**(5), 2053–2080 (2010)
5. Candes, E., Plan, Y.: Matrix completion with noise. Proc. IEEE **98**(6), 925–936 (2010)
6. Crammer, K., Dredze, M., Pereira, F.: Confidence-weighted linear classification for natural language processing. J. Mach. Learn. Res. **13**, 1891–1926 (2012)
7. Forero, P.A., Cano, A., Giannakis, G.B.: Consensus-based distributed support vector machines. J. Mach. Learn. Res. **11**, 1663–1707 (2010)
8. Joachims, T.: Making large-scale support vector machine learning practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, chap. 11, pp. 169–184. MIT Press, Cambridge (1999)
9. Lanckriet, G., Cristianini, N., Bartlett, P., E.G., L., Jordan, M.: Learning the kernel matrix with semidefinite programming. In: Proceedings of the 19th International Conference on Machine Learning (2002)
10. Lee, S., Bockermann, C.: Scalable stochastic gradient descent with improved confidence. In: Big Learning - Algorithms, Systems, and Tools for Learning at Scale, NIPS Workshop (2011)
11. Lee, S., Pölitz, C.: Kernel completion for learning consensus support vector machines in bandwidth-limited sensor networks. In: International Conference on Pattern Recognition Applications and Methods (2014)
12. Lee, S., Stolpe, M., Morik, K.: Separable approximate optimization of support vector machines for distributed sensing. In: De Bie, T., Cristianini, N., Flach, P.A. (eds.) ECML PKDD 2012, Part II. LNCS, vol. 7524, pp. 387–402. Springer, Heidelberg (2012)
13. Mangasarian, O.L., Musicant, D.R.: Lagrangian support vector machines. J. Mach. Learn. Res. **1**, 161–177 (2001)
14. Morik, K., Bhaduri, K., Kargupta, H.: Introduction to data mining for sustainability. Data Min. Knowl. Disc. **24**(2), 311–324 (2012)
15. Huang, L., Huang, L., Joseph, A.D., Joseph, A.D., Nguyen, X.L., Nguyen, X.L.: Support vector machines, data reduction, and approximate kernel matrices. In: Goethals, B., Goethals, B., Daelemans, W., Daelemans, W., Morik, K., Morik, K. (eds.) ECML PKDD 2008, Part II. LNCS (LNAI), vol. 5212, pp. 137–153. Springer, Heidelberg (2008)
16. Recht, B., Fazel, M., Parrilo, P.A.: Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev. **52**(3), 471–501 (2010)
17. Recht, B., Ré, C.: Parallel stochastic gradient algorithms for large-scale matrix completion. Technical report, University of Wisconsin-Madison, April 2011
18. Scholkopf, B., Smola, A.J.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)

19. Steinwart, I.: On the influence of the kernel on the consistency of support vector machines. J. Mach. Learn. Res. **2**, 67–93 (2002)
20. Stolpe, M., Bhaduri, K., Das, K., Morik, K.: Anomaly detection in vertically partitioned data by distributed core vector machines. In: Nijssen, S., Železný, F., Blockeel, H., Kersting, K. (eds.) ECML PKDD 2013, Part III. LNCS, vol. 8190, pp. 321–336. Springer, Heidelberg (2013)
21. Trefethen, L.N., Bau, D.: Numerical Linear Algebra. SIAM, Philadelphia (1997)