Kernel Completion for Learning Consensus Support Vector Machines in Bandwidth-limited Sensor Networks

Sangkyun Lee and Christian Pölitz
Fakultät für Informatik, LS VIII, Technische Universität Dortmund, 44221 Dortmund, Germany

Keywords: Sensor Networks, Support Vector Machines, Gaussian Kernels, Sampling, Matrix Completion.

Abstract: Recent developments in sensor technology allows for capturing dynamic patterns in vehicle movements, temperature changes, and sea-level fluctuations, just to name a few. A usual way for decision making on sensor networks, such as detecting exceptional surface level changes across the Pacific ocean, involves collecting measurement data from all sensors to build a predictor in a central processing station. However, data collection becomes challenging when communication bandwidth is limited, due to communication distance or low-energy requirements. Also, such settings will introduce unfavorable latency for making predictions on unseen events. In this paper, we propose an alternative strategy for such scenarios, aiming to build a consensus support vector machine (SVM) in each sensor station by exchanging a small amount of sampled information from local kernel matrices amongst peers. Our method is based on decomposing a "global" kernel defined with all features into "local" kernels defined only with attributes stored in each sensor station, sampling few entries of the decomposed kernel matrices that belong to other stations, and filling in unsampled entries in kernel matrices by matrix completion. Experiments on benchmark data sets illustrate that a consensus SVM can be built in each station using limited communication, which is competent in prediction performance to an SVM built with accessing all features.

1 INTRODUCTION

Sensors can monitor many different kinds of dynamics in nature, generating numerous data, and thereby embodying research challenges in machine learning and data mining (Whittaker et al., 1997; Lippi et al., 2010; Morik et al., 2012). There is a wide spectrum of sensing devices available today, but they share a common property: communication is costly and should be avoided whenever possible, due to restrictions in bandwidth or in energy consumption. This is a clear barrier for global decision making, for which it is typically required to agglomerate all local sensor measurements into a central location for processing.

On the other hand, many sensors are stationed within devices equipped with surprisingly powerful and energy-efficient computation units. This has motivated us to use computation to save communication. Specifically, we aim to build a support vector machine (SVM) (Boser et al., 1992) in each of such devices, called sensor stations, using local measurements and a small amount of sampled information transmitted from other stations. The goal is to obtain a *consensus SVM* in each station that behaves similarly to a global SVM that could be constructed if we collect information from all stations for central processing.

Our work is closely related to learning SVMs in distributed environments, which can be split into two categories. Case I: examples are distributed (features are not distributed). In such cases a global SVM can be trained using a distributed optimization algorithm (Boyd et al., 2011), or separate SVMs can be trained locally for data partitions with extra constraints to produce similar models (Forero et al., 2010). Alternatively, local SVMs can be trained in their primal form independently on data partitions and then combined to produce a model with a reduced variance (Lee and Bockermann, 2011; Crammer et al., 2012). Case II: features are distributed (examples are not distributed). In such cases a central coordination of local SVM training has been considered to improve global prediction performance (Lee et al., 2012; Stolpe et al., 2013). Our work focuses on the second case where features are distributed, considering communication-efficient approximations to a global kernel matrix (which could be built by accessing all features) in each station, without any central coordination.

Lee S. and Pölitz C..
Kernel Completion for Learning Consensus Support Vector Machines in Bandwidth-limited Sensor Networks.
DOI: 10.5220/0004829401130124
In *Proceedings of the 3rd International Conference on Pattern Recognition Applications and Methods* (ICPRAM-2014), pages 113-124
ISBN: 978-989-758-018-5

Our suggested method is based on decompositions of a (global) kernel into separate parts, where each of them is another kernel defined with attributes stored locally in each sensor station. Each decomposed kernel matrix is stored in a sensor station where corresponding attributes are stored. Each station receives few sampled entries of the decomposed kernel matrices stored in remote stations, and then applies matrix completion to approximate the values of unobserved entries. Using these altogether, a consensus SVM is created in each station, which can be applied for predicting future events using local and remote information in a similar fashion.

We denote the Euclidean norm by $\|\cdot\|$ and the cardinality of a finite set A by $|A|$ throughout the paper.

2 SUPPORT VECTOR MACHINES WITH DECOMPOSED KERNELS

Let us consider sensor stations represented as nodes $n = 1,2,\ldots,N$ in a network, where each node stores measurements from its own sensors, in a feature vector $\mathbf{x}_i[n] \in \Re^{p_n}$, of sensing targets $i = 1,2,\ldots,m$. For simplicity we assume that communication between any pair of nodes is allowed. A collection of all these vectors $\mathbf{x}_i = (\mathbf{x}_i[1]^T, \mathbf{x}_i[2]^T, \ldots, \mathbf{x}_i[N]^T)^T$ can be seen as an input vector of length $p = \sum_{n=1}^{N} p_n$.

2.1 Support Vector Machines

The dual formulation of SVMs is described as follows (Shawe-Taylor and Sun, 2011),

$$\begin{aligned} \min_{\alpha\in\Re^m} \ & \frac{1}{2}\alpha^T\mathbf{Q}\alpha - \mathbf{1}^T\alpha \ , \\ \text{subject to } & \mathbf{y}^T\alpha = 0, \\ & \mathbf{0} \le \alpha \le C\mathbf{1} \ . \end{aligned} \tag{1}$$

Here $\mathbf{1} := (1,1,\ldots,1)^T$ and $\mathbf{y} := (y_1, y_2, \ldots, y_m)^T$ are column vectors of length m, and C is a given constant. (Without loss of generality, we focus on the case of classification – our method can be generalized for other types.) The matrix $\mathbf{Q} \in \Re^{m\times m}$ 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 whose elements are given by the vector $\mathbf{y}$. SVMs have been successful in many applications, including multitask multiclass learning problems (Ji and Sun, 2013) for example.

2.2 Decomposition of Kernels

We consider two different decompositions of the kernel matrix $\mathbf{K}$, especially those obtainable from the popular Gaussian kernel. We refer to them as "MULTIPLICATIVE" and "ADDITIVE", defined as follows for $i, j = 1,2,\ldots,m$,

(MULTIPLICATIVE)

$$[\mathbf{K}]_{ij} = \prod_{n=1}^{N} \exp\left(-\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2\right) \ , \text{ and}$$

(ADDITIVE)

$$[\mathbf{K}]_{ij} = \frac{1}{N}\sum_{n=1}^{N} \exp\left(-\gamma_n\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2\right) \ . \tag{2}$$

The MULTIPLICATIVE kernel is indeed the same as the standard Gaussian kernel (Scholkopf and Smola, 2001), but our description above reveals that it can be constructed by multiplying "local" Gaussian kernels defined with attributes stored locally in sensor stations. The construction of ADDITIVE is similar, except that local Gaussian kernels are averaged, not multiplied. ADDITIVE resembles how kernels are used in the multiple kernel learning (Lanckriet et al., 2002): the connection is further discussed in Section 4.3. Note that MULTIPLICATIVE has a single parameter $\gamma > 0$, but ADDITIVE has a separate parameter $\gamma_n > 0$ for each local kernel.

2.3 Local and Remote Parts in Decomposition

From the definitions in (2), we identify the parts that can be computed with attributes stored locally in each node (local parts), and that need to be transferred from other nodes in a sensor network (remote parts).

First, the expression of MULTIPLICATIVE can be rewritten in the following way,

$$\begin{aligned} [\mathbf{K}]_{ij} &= \exp\left(\sum_{n=1}^{N} -\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2\right) \\ &= \exp\left(-\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2\right) \\ &\qquad \prod_{n'\neq n} \exp\left(-\gamma\|\mathbf{x}_i[n'] - \mathbf{x}_j[n']\|^2\right) \\ &= [\mathbf{G}_n]_{ij}[\mathbf{G}_{-n}]_{ij} \ , \end{aligned} \tag{3}$$

where the "local" Gaussian kernel $\mathbf{G}_n$ for a node n and the product $\mathbf{G}_{-n}$ of all "remote" kernels are defined entrywise respectively by

$$\begin{aligned} [\mathbf{G}_n]_{ij} &:= \exp\left(-\gamma\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2\right) && \text{(local)} \\ [\mathbf{G}_{-n}]_{ij} &:= \prod_{n'\neq n} [\mathbf{G}_{n'}]_{ij} && \text{(remote)} \ . \end{aligned}$$

Similarly, ADDITIVE can be written as

$$[\mathbf{K}]_{ij} = \frac{1}{N}\left([\mathbf{H}_n]_{ij} + \sum_{n'\neq n}[\mathbf{H}_{n'}]_{ij}\right) = \frac{1}{N}[\mathbf{H}_n + \mathbf{H}_{-n}]_{ij} \ , \quad (4)$$

where $\mathbf{H}_n$ is the local part and $\mathbf{H}_{-n}$ is the remote part for node n, defined respectively by

$$[\mathbf{H}_n]_{ij} := \exp\left(-\gamma_n\|\mathbf{x}_i[n]-\mathbf{x}_j[n]\|^2\right) \quad \text{(local)}$$
$$[\mathbf{H}_{-n}]_{ij} := \sum_{n'\neq n}[\mathbf{H}_{n'}]_{ij} \quad \text{(remote)} \ .$$

For a node n, the computation of the local part $\mathbf{G}_n$ (or $\mathbf{H}_n$) is done exactly using local attributes, where the remote part $\mathbf{G}_{-n}$ (or $\mathbf{H}_{-n}$) is to be approximated.

3 KERNEL COMPLETION

Let us denote the kernel matrix to be estimated in the nth node by $\widetilde{\mathbf{K}}_n$, which is computed by

$$[\widetilde{\mathbf{K}}_n]_{ij} := \begin{cases} [\mathbf{G}_n]_{ij}[\widetilde{\mathbf{G}}_{-n}]_{ij} & \text{(MULTIPLICATIVE)} \\ \frac{1}{N}[\mathbf{H}_n + \widetilde{\mathbf{H}}_{-n}]_{ij} & \text{(ADDITIVE)} \ . \end{cases} \quad (5)$$

Here $\widetilde{\mathbf{G}}_{-n}$ (or $\widetilde{\mathbf{H}}_{-n}$) is an estimate of the remote part $\mathbf{G}_{-n}$ (or $\mathbf{H}_{-n}$). Once we have $\widetilde{\mathbf{K}}_n$, it can be plugged in (1) replacing $\mathbf{Q}$ in the form of $\widetilde{\mathbf{Q}}_n := \mathbf{Y}\widetilde{\mathbf{K}}_n\mathbf{Y}$.

In order to obtain the estimate $\widetilde{\mathbf{G}}_{-n}$ (or $\widetilde{\mathbf{H}}_{-n}$), we make use of *matrix completion* (Candès and Recht, 2009), which is a method to reconstruct a matrix from only a few sampled entries from it. The purpose of using matrix completion is (i) to reduce the number of entries required to be sampled from remote kernel parts in bandwidth-limited situations. Matrix completion will not be required if all nodes provide complete information. And it is (ii) to avoid the complexity of defining an optimal sampling strategy. That is, a simple uniform random sampling strategy is enough for matrix completion to guarantee the perfect recovery of the original kernel matrix with high probability.

We first discuss extra constraints we need to add to matrix completion, so that the resulting matrix is to be a valid kernel matrix.

3.1 Constraints on $\widetilde{\mathbf{G}}_{-n}$ and $\widetilde{\mathbf{H}}_{-n}$

First, $\widetilde{\mathbf{G}}_{-n}$ has to be a symmetric matrix where diagonal entries are all ones. It becomes a valid kernel matrix if and only if it is positive semidefinite (Scholkopf and Smola, 2001), that is, $\mathbf{z}^T\widetilde{\mathbf{G}}_{-n}\mathbf{z} \geq 0$ for all $\mathbf{z} \in \Re^m$. Since each entry of a local Gaussian kernel $\mathbf{G}_{n'}$ is in the range $(0,1]$ by definition, the product of such entries in $\widetilde{\mathbf{G}}_{-n}$ should be in the same range as well. Next, $\widetilde{\mathbf{H}}_{-n}$ shares the same properties as $\widetilde{\mathbf{G}}_{-n}$, except for that each diagonal element of $\widetilde{\mathbf{H}}_{-n}$ is $(N-1)$, not one, by construction.

There is another possible way to decompose the MULTIPLICATIVE kernel,

$$[\mathbf{K}]_{ij} = \exp\Big(-\gamma\|\mathbf{x}_i[n]-\mathbf{x}_j[n]\|^2 - \gamma\sum_{n'\neq n}\|\mathbf{x}_i[n']-\mathbf{x}_j[n']\|^2\Big) = \exp\left([\mathbf{D}_n + \mathbf{D}_{-n}]_{ij}\right) \ ,$$

with

$$[\mathbf{D}_n]_{ij} := -\gamma\|\mathbf{x}_i[n]-\mathbf{x}_j[n]\|^2, \quad [\mathbf{D}_{-n}]_{ij} := \sum_{n'\neq n}[\mathbf{D}_{n'}]_{ij} \ .$$

Then our task becomes making an estimate $\widetilde{\mathbf{D}}_{-n}$ of a distance matrix $\mathbf{D}_{-n}$, which has zero diagonal entries. The estimate defines a valid distance matrix if and only if it is conditionally positive semidefinite, that is, $\mathbf{z}^T\widetilde{\mathbf{D}}_{-n}\mathbf{z} \geq 0$ for all $\mathbf{z} \in \Re^m$ with $\mathbf{z}^T\mathbf{1} = 0$ (Schoenberg, 1938). This implies that $\widetilde{\mathbf{D}}_{-n}$ is positive semidefinite, or it has a single negative eigenvalue. It turned out that our kernel completion in forms of (3) performed better in our experiments, so we did not pursue this direction further.

3.2 Low-rank Matrix Completion

For the description of matrix completion, we follow the line of discussion in (Recht and Ré, 2011). Matrix completion reconstructs a full matrix from only a few entries sampled from the original matrix. In general, matrix completion works with matrices in any shape, but we focus on square matrices here.

Suppose that $\mathbf{X} \in \Re^{m\times m}$ is a matrix we wish to recover, and that the entries at $(i,j) \in \Omega$ of $\mathbf{X}$ are revealed and stored in another matrix $\mathbf{M}$. Matrix completion solves the following convex optimization problem to recover $\mathbf{X}$,

$$\min_{\mathbf{X}} \sum_{(i,j)\in\Omega}(\mathbf{X}_{ij}-\mathbf{M}_{ij})^2 + \lambda\|\mathbf{X}\|_*, \ \|\mathbf{X}\|_* := \sum_{k=1}^{m}\sigma_k(\mathbf{X}) \ .$$

Here $\|\mathbf{X}\|_*$ is 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 $\mathbf{X}$.

The nuclear norm simplifies when we assume that the matrix $\mathbf{X}$ has the rank r, and consider a factorization of $\mathbf{X}$ into $\mathbf{L}\mathbf{R}^T$ for some $\mathbf{L} \in \Re^{m\times r}$ and $\mathbf{R} \in \Re^{m\times r}$. This leads to $\mathbf{X}_{ij} = [\mathbf{L}\mathbf{R}^T]_{ij} = \mathbf{L}_{i\cdot}\mathbf{R}_{j\cdot}^T$, and

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

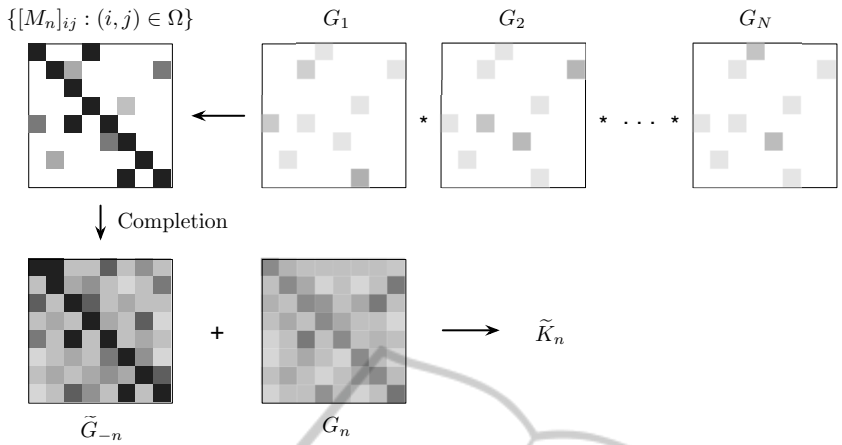

Figure 1: A schematic of kernel completion, with MULTIPLICATIVE kernel. Each node *n* (i) collects and summarizes the samples corresponding to Ω from remote kernel matrices $\mathbf{G}_{n'}$ as the known entries of the matrix $\mathbf{M}_n$, and then (ii) fills up the unknown entries of $\mathbf{M}_n$ via matrix completion, producing $\widetilde{\mathbf{G}}_{-n}$, (iii) forming an estimate of kernel matrices together with the exact local kernel $\mathbf{G}_n$.

where $\|\cdot\|_F$ is the Frobenius norm. The equivalence can be understood by taking a singular value decomposition $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^T$ and setting $\mathbf{L} = \mathbf{U}\Sigma^{1/2}$ and $\mathbf{R} = \mathbf{V}\Sigma^{1/2}$. Then $\|\mathbf{X}\|_* = \mathrm{tr}(\Sigma)$, $\|\mathbf{L}\|_F^2 = \mathrm{tr}(\mathbf{L}^T\mathbf{L}) = \mathrm{tr}(\Sigma)$, and $\|\mathbf{R}\|_F^2 = \mathrm{tr}(\Sigma)$, so the equality holds. For details, we refer to (Recht et al., 2010; Recht and Ré, 2011).

Using the property of the nuclear norm on rank-*r* matrices, we can reformulate the matrix completion optimization as

$$\min_{\mathbf{L},\mathbf{R}} \sum_{(i,j)\in\Omega} (\mathbf{L}_{i\cdot}\mathbf{R}_{j\cdot}^T - \mathbf{M}_{ij})^2 + \frac{\lambda}{2}\|\mathbf{L}\|_F^2 + \frac{\lambda}{2}\|\mathbf{R}\|_F^2 \quad . \quad (6)$$

To obtain solutions, we use the JELLYFISH algorithm (Recht and Ré, 2011), which is a highly parallel incremental gradient descent procedure to find the minimizers, making use of the fact that the gradient of the above objective depends on only $\mathbf{L}_{i\cdot}$ and $\mathbf{R}_{j\cdot}$, and therefore the computation of each iteration can be easily distributed for the pairs $(i,j) \in \Omega$.

3.2.1 Constrained Matrix Completion

To incorporate the constraints discussed in Section 3.1, we need to find a matrix $\widetilde{\mathbf{X}}^*$ that is close to $\mathbf{L}^*(\mathbf{R}^*)^T$ where $\mathbf{L}^*$ and $\mathbf{R}^*$ are the solutions of (6) (both are *m* by *r*, $r \in (0,m]$), belonging to a convex set $\mathcal{K}$ of symmetric positive semidefinite rank-*r* matrices,

$$\mathcal{K} := \{\widetilde{\mathbf{X}} : \widetilde{\mathbf{X}} \succeq 0,\ (\widetilde{\mathbf{X}})^T = \widetilde{\mathbf{X}},\ \mathrm{rank}(\widetilde{\mathbf{X}}) = r\} \quad .$$

The following lemma shows that the description of this set can be simplified.

Lemma 3.1. *The elements $\widetilde{\mathbf{X}}$ in $\mathcal{K}$ must have the form*

$$\widetilde{\mathbf{X}} = \mathbf{Z}\mathbf{Z}^T,\ \textit{where}\ \mathbf{Z} \in \Re^{m\times r} \quad .$$

Proof. Suppose that $\widetilde{\mathbf{X}}$ is in $\mathcal{K}$. Since $\widetilde{\mathbf{X}}$ is symmetric and positive semidefinite, from the eigen decomposition of $\widetilde{\mathbf{X}}$ there exists a factor $\mathbf{U} \in \Re^{m\times m}$ such that $\widetilde{\mathbf{X}} = \mathbf{U}\Sigma\mathbf{U}^T$ where $\Sigma \geq \mathbf{0}$ is the diagonal matrix of eigenvalues. Removing the columns of $\mathbf{U}$ and the part of Σ corresponding to the zero eigenvalues, we obtain $\widetilde{\mathbf{U}} \in \Re^{m\times r}$ and $\widetilde{\Sigma} \in \Re^{r\times r}$. Then $\mathbf{Z} = \widetilde{\mathbf{U}}\widetilde{\Sigma}^{1/2}$ can be constructed so that $\widetilde{\mathbf{X}} = \mathbf{Z}\mathbf{Z}^T$.

Conversely, any $\widetilde{\mathbf{X}}$ in the form of $\widetilde{\mathbf{X}} = \mathbf{Z}\mathbf{Z}^T$ satisfies $\widetilde{\mathbf{X}}^T = \widetilde{\mathbf{X}}$ (symmetric) and $\mathbf{z}^T\widetilde{\mathbf{X}}\mathbf{z} = \|\mathbf{z}^T\mathbf{Z}\|_2^2 \geq 0$ for all $\mathbf{z} \in \Re^m$ (positive semidefinite). Therefore $\widetilde{\mathbf{X}} = \mathbf{Z}\mathbf{Z}^T$ is an element of $\mathcal{K}$. □

This lemma indicates that the set $\mathcal{K}$ can be rewritten as simple as,

$$\mathcal{K} = \{\mathbf{Z}\mathbf{Z}^T : \mathbf{Z} \in \Re^{m\times r}\} \quad .$$

The next step is to find a matrix $\mathbf{Z}$ such that $\mathbf{Z}\mathbf{Z}^T$ is close to $\mathbf{L}^*(\mathbf{R}^*)^T$. An ℓ_2 projection of $\mathbf{L}^*(\mathbf{R}^*)^T$ onto $\mathcal{K}$ requires an iterative procedure which is as costly as finding $\mathbf{L}^*$ and $\mathbf{R}^*$. Therefore we consider an alternative projection for which we have a closed-form solution,

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

From the KKT conditions, the solution is obtained by

$$\mathbf{Z}^* = \frac{\mathbf{L}^* + \mathbf{R}^*}{2} \quad .$$

Then a projection $\widetilde{\mathbf{X}}^*$ is obtained by $\widetilde{\mathbf{X}}^* = \mathbf{Z}^*(\mathbf{Z}^*)^T$, which has a guarantee on its quality as stated in the next lemma:

Lemma 3.2. *The trace-norm distance between* $\widetilde{\mathbf{X}}^* = \mathbf{Z}^*(\mathbf{Z}^*)^T$*, where* $\mathbf{Z}^* = (\mathbf{L}^* + \mathbf{R}^*)/2$*, and* $\mathbf{L}^*(\mathbf{R}^*)^T$ *is bounded, that is,*

$$tr(\widetilde{\mathbf{X}}^* - \mathbf{L}^*(\mathbf{R}^*)^T) \leq \frac{1}{4}\|\mathbf{L}^* - \mathbf{R}^*\|_F^2 \ .$$

Proof. Using $\widetilde{\mathbf{X}}^* = \mathbf{Z}^*(\mathbf{Z}^*)^T$, the result can be derived as follows,

$$\begin{aligned}
&\mathrm{tr}(\widetilde{\mathbf{X}}^* - \mathbf{L}^*(\mathbf{R}^*)^T) \\
&= \frac{1}{4}\mathrm{tr}\{(\mathbf{L}^* + \mathbf{R}^*)(\mathbf{L}^* + \mathbf{R}^*)^T - 4\mathbf{L}^*(\mathbf{R}^*)^T\} \\
&= \frac{1}{4}\mathrm{tr}\{(\mathbf{L}^* - \mathbf{R}^*)(\mathbf{L}^* - \mathbf{R}^*)^T\} \\
&= \frac{1}{4}\|\mathbf{L}^* - \mathbf{R}^*\|_F^2 \ .
\end{aligned}$$

Here we have used the properties of the trace that $\mathrm{tr}(\mathbf{X}+\mathbf{Y}) = \mathrm{tr}(\mathbf{X}) + \mathrm{tr}(\mathbf{Y}) = \mathrm{tr}(\mathbf{X}) + \mathrm{tr}(\mathbf{Y}^T) = \mathrm{tr}(\mathbf{X} + \mathbf{Y}^T)$ and $\mathrm{tr}(\mathbf{X}\mathbf{X}^T) = \|\mathbf{X}\|_F^2$. □

The above lemma tells that the distance between $\widetilde{\mathbf{X}}^*$ and $\mathbf{L}^*(\mathbf{R}^*)$ becomes small whenever $\mathbf{L}^* \approx \mathbf{R}^*$, which is likely to happen in our case since we define $\mathbf{M}$ and Ω in such a way that if $(i,j) \in \Omega$ then $(j,i) \in \Omega$, and $\mathbf{M}_{ij} = \mathbf{M}_{ji}$.

3.2.2 Sample Index Pair Subset Ω

In our method, we assume that a single sample index pair set $\Omega \subset \{(i,j) : 1 \leq i,j \leq m\}$ is fixed across all nodes. It is more efficient than using multiple sample sets, since otherwise we have to store and complete each remote matrix $\mathbf{G}_{n'}$, $n' \in \{1,\ldots,N\} \setminus \{n\}$, separately. Using a pre-defined Ω across nodes can be implemented as using a fixed random seed for a pseudo random number generator, so that Ω does not have to be transferred at all.

Given Ω, each node n receives information from other nodes n' and stores it in $\mathbf{M}_n$ as follows for all $(i,j) \in \Omega$,

$$[\mathbf{M}_n]_{ij} = \begin{cases} \prod\limits_{n' \in \{1,\ldots,N\}\setminus\{n\}} [\mathbf{G}_{n'}]_{ij} & (\textsc{Multiplicative}) \\ \sum\limits_{n' \in \{1,\ldots,N\}\setminus\{n\}} \frac{[\mathbf{H}_{n'}]_{ij}}{N-1} & (\textsc{Additive}) \ . \end{cases}$$

That is, the communication cost for each node n is $O((N-1)|\Omega|)$. The use of matrix completion makes it possible to choose an Ω of relatively small size ($O(m^{1.2} r \log m)$ when $\mathbf{M}_n$ is a rank-r matrix, see Theorem 4.1 for details) in a simple way, that is, via random uniform sampling.

Once the matrix $\mathbf{M}_n$ is obtained, the node n solves the matrix completion (6) with $[\mathbf{M}_n]_\Omega$ to obtain $\mathbf{Z}_n^* = (\mathbf{L}_n^* + \mathbf{R}_n^*)/2$, and then compute $\widetilde{\mathbf{G}}_{-n} = \mathbf{Z}_n^*(\mathbf{Z}_n^*)^T$ or $\widetilde{\mathbf{H}}_{-n} = (N-1)\mathbf{Z}_n^*(\mathbf{Z}_n^*)^T$, based on Lemmas 3.1 and 3.2. An estimate of the kernel $\mathbf{K}$, obtained by (5), is then used for training an SVM.

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

3.3 Extra Saving with ADDITIVE

The description of the matrix completion optimization (6) involves all training examples. However, if a (super-)set of the *support vectors* (SVs), which fully determines a prediction function, is known a priori, then we can solve the completion problem only for the set, reducing the cost of matrix completion.

Let us consider the SVs of the "global" SVM problem (1) equipped with the exact ADDITIVE kernel (2), which is constructed by accessing all features in a central location. We denote this set of SVs as S^*. Note that S^* is never obtained, since we do not solve such a global problem.

We try to estimate S^* from the sets of "local" SVs. These local SVs are obtained from solving an individual SVM (1) in each node n, using only the local features, that is, setting the scaled kernel matrix as $\mathbf{Q} = \mathbf{Y}\mathbf{H}_n\mathbf{Y}$ for the local kernel matrix $\mathbf{H}_n = \exp(-\gamma_n\|\mathbf{x}_i[n] - \mathbf{x}_j[n]\|^2)$. We denote the set of SVs in the node n by S_n obtained in this way.

In the next theorem, we show that the union of the local SV sets S_n encompasses the global SV set S^*. To shorten the length of our proof, here we show the case for the SVMs without any intercept, that is, the constraint $\mathbf{y}^T\alpha = 0$ is removed (the same result holds for the case with intercepts).

Theorem 3.3. *Consider the global SVM problem with the* ADDITIVE *kernel and its set of SVs* S^**,*

$$\begin{aligned}
\alpha^* &:= \operatorname*{arg\,min}_{0 \leq \alpha \leq C\mathbf{1}} \ \frac{1}{2}\alpha^T\mathbf{Y}\left(\frac{1}{N}\sum_{n=1}^N \mathbf{H}_n\right)\mathbf{Y}\alpha - \mathbf{1}^T\alpha \ , \\
S^* &:= \{i : [\alpha^*]_i > 0\} \ ,
\end{aligned}$$

and the corresponding local SVM problem and its SVs for each node n*,* $n = 1,2,\ldots,N$*,*

$$\begin{aligned}
\alpha_n^* &:= \operatorname*{arg\,min}_{0 \leq \alpha \leq C\mathbf{1}} \ \frac{1}{2}\alpha^T\mathbf{Y}(\mathbf{H}_n)\mathbf{Y}\alpha - \mathbf{1}^T\alpha \ , \\
S_n^* &:= \{i : [\alpha_n^*]_i > 0\} \ .
\end{aligned}$$

Then we have

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

Proof. Let us consider an index $i \in S^*$ of an SV of the global SVM problem, such that $[\alpha^*]_i > 0$. Suppose that the ith component of the gradient of all local

SVM problems at α^* is strictly positive, that is,

$$[\mathbf{Y}\mathbf{H}_n\mathbf{Y}\alpha^* - \mathbf{1}]_i > 0, \ \forall n \in \{1,2,\ldots,N\} \ . \quad (7)$$

Let us look into the optimality condition of the global SVM, regarding the ith component of the optimizer α^*. From the KKT conditions, we have

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

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

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

If (7) 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 (7) is not satisfied, that is, $[\mathbf{Y}\mathbf{H}_n\mathbf{Y}\alpha^* - \mathbf{1}]_i \leq 0$. This means that if we search for the local SVM solution at the node n starting from α^*, we must increase the value of the ith component from $[\alpha^*]_i$ to reach the minimizer $[\alpha_n^*]_i$ of this local SVM problem, since otherwise we will increase the objective function value. That is,

$$[\alpha_n^*]_i \geq [\alpha^*]_i > 0.$$

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. □

Theorem 3.3 enables us to restrict our attention to the union SV set without losing any information for the case of ADDITIVE, where the size of the union SV set is typically much smaller than that of the entire training examples index set. In effect, this leads to more efficiency in solving the matrix completion problem (6), by reducing the number of variables from $O(m^2)$ to $O(|\cup_n S_n^*|^2)$.

3.4 Algorithm

Our kernel completion method for training SVMs is summarized in Algorithm 1. There, we have used the symbol $\circ$ to represent elementwise multiplications between matrices.

We have implemented our algorithm as open-source in C++, based on the JELLYFISH code[1] (Recht and Ré, 2011) for matrix completion, and SVM-LIGHT[2] (Joachims, 1999) for solving SVMs. Our implementation makes use of the union SVs set theorem (Theorem 3.3) for the ADDITIVE approach to reduce kernel completion time, but not for MULTIPLICATIVE since the theorem does not apply for this case.

[1] Available for download at http://hazy.cs.wisc.edu/hazy/victor/jellyfish/

4 RELATED WORK

Here we present existing methods that are closely related to our development.

4.1 Separable Approximate Optimization of SVMs

Lee, Stolpe, and Morik (Lee et al., 2012) have investigated the primal formulation of SVMs in a setting close to ours. In their work, the distributed nature of input features is considered via making an individual approximate feature mapping φ_n for each node n, such that for a given local kernel function k_n, it approximates kernel evaluations,

$$\langle \varphi_n(\mathbf{x}_i), \varphi_n(\mathbf{x}_j)\rangle \approx k_n(\mathbf{x}_i, \mathbf{x}_j), \ \forall i, j \ .$$

Using this mapping, each node solves its own local SVM in the primal, producing a decision vector $\mathbf{w}^*[n]$. Based upon the local solutions, a "global" SVM is explicitly constructed in a central node, which is defined with the collection of local decision vectors and local feature mappings (weighted by $\mu_n \geq 0$), that is,

$$\mathbf{w} := \begin{bmatrix} \mathbf{w}[1] \\ \vdots \\ \mathbf{w}[N] \end{bmatrix}, \ \varphi(\mathbf{x}) := \begin{bmatrix} \mu_1\varphi_1(\mathbf{x}[1]) \\ \vdots \\ \mu_N\varphi_N(\mathbf{x}[N]) \end{bmatrix} \ .$$

An interesting characteristic of this central SVM is that if we have optimized the local SVMs using the specific forms of loss functions ℓ_n whose weighted summation forms an upper bound of the original loss function ℓ, that is,

$$\ell(\mathbf{w}^T\varphi(\mathbf{x}), y) \leq \sum_{i=1}^{N}\mu_n\ell_n\left(\mathbf{w}[n]^T\varphi_n(\mathbf{x}[n]), y\right) \ ,$$

then it can be shown that this central SVM minimizes an upper bound of the standard SVM objective with the original loss function. The nonnegative weights $\mu_1, \mu_2, \ldots, \mu_N$ are optimized in the central node, which requires transferring $O(m)$ numbers from each local node $n = 1, 2, \ldots, N$.

[2] Available at http://svmlight.joachims.org/

Algorithm 1: Kernel Completion for SVMs.

input : A data set $\{(\mathbf{x}_i, y_i)\}_{i=1}^m$, a sample set Ω, and parameters γ, $\{\gamma_n\}_{n=1}^N$.
(parallel: in each node $n = 1, 2, \ldots, N$**)**
  input : local measurements/labels $\{(\mathbf{x}_i[n], y_i)\}_{i=1}^m$.
  Compute local kernel matrix $\mathbf{G}_n$ for MULTIPLICATIVE (or $\mathbf{H}_n$ for ADDITIVE);

  if ADDITIVE **then**
    `// Make the union of SV index sets (ADDITIVE only)`
    Solve the SVM (1) with $\mathbf{Q} \leftarrow \mathbf{Y}\mathbf{G}_n\mathbf{Y}$, to obtain the SV index set S_n^*;
    Receive $S_{n'}^*$ for all other nodes n';
    Trim Ω to fit $\cup_{n=1}^N S_n^*$;
  end

  `// Collect samples from remote kernel matrices`
  Initialize:

$$[\mathbf{M}_n]_\Omega \leftarrow \begin{cases} \mathbf{1}\mathbf{1}^T & \text{(MULTIPLICATIVE)} \\ \mathbf{0} & \text{(ADDITIVE)} \end{cases}$$

  for $n' \in \{1, 2, \ldots, N\} \setminus \{n\}$ **do**
    Receive $[\mathbf{G}_{n'}]_\Omega$ (MULTIPLICATIVE), or $[\mathbf{H}_{n'}]_\Omega$ (ADDITIVE).

$$[\mathbf{M}_n]_\Omega \leftarrow \begin{cases} [\mathbf{M}_n]_\Omega \circ [\mathbf{G}_{n'}]_\Omega & \text{(MULTIPLICATIVE)} \\ [\mathbf{M}_n]_\Omega + [\mathbf{H}_{n'}]_\Omega & \text{(ADDITIVE)} \end{cases}$$

  end
  For ADDITIVE, scale $[\mathbf{M}_n]_\Omega \leftarrow [\mathbf{M}_n]_\Omega / (N-1)$;

  `// Kernel completion for` $\widetilde{\mathbf{K}}_n$
  Solve matrix completion (6) with observed entries in $[\mathbf{M}_n]_\Omega$, to obtain $\mathbf{L}_n^*$ and $\mathbf{R}_n^*$;
  Compute projections, to obtain $\mathbf{Z}_n^* \leftarrow (\mathbf{L}_n^* + \mathbf{R}_n^*)/2$;
  Compute the estimated kernel matrix $\widetilde{\mathbf{K}}_n$ by (5):

$$\begin{cases} \widetilde{\mathbf{G}}_{-n} \leftarrow \mathbf{Z}_n^* (\mathbf{Z}_n^*)^T & \text{(MULTIPLICATIVE)} \\ \widetilde{\mathbf{H}}_{-n} \leftarrow (N-1)\mathbf{Z}_n^* (\mathbf{Z}_n^*)^T & \text{(ADDITIVE)} \end{cases}, \quad \widetilde{\mathbf{K}}_n \leftarrow \begin{cases} \mathbf{G}_n \circ \widetilde{\mathbf{G}}_{-n} & \text{(MULTIPLICATIVE)} \\ \frac{1}{N}(\mathbf{H}_n + \widetilde{\mathbf{H}}_{-n}) & \text{(ADDITIVE)} \end{cases}$$

  `// Obtain an estimated consensus SVM`
  Solve the SVM problem (1) replacing $\mathbf{Q}$ with $\widetilde{\mathbf{Q}}_n \leftarrow \mathbf{Y}\widetilde{\mathbf{K}}_n\mathbf{Y}$;
(end)

The kernel function of this central SVM is indeed a weighted approximation of our ADDITIVE kernel (4), when each local feature mapping approximates a Gaussian kernel (parametrized by γ_n) with local features, and the weights are fixed to $\mu_n = 1/\sqrt{N}$.

However, our work is quite different from this approach in several ways. First, we do not require a special node to build a central SVM, therefore avoiding a communication complexity of $O(mN)$. Moreover, to classify a test point in the central SVM approach, $O(N)$ elements have to be transferred to a central node for each test point. However in our case testing can be done in any node, although it also requires some communication. Second, in our method estimation happens only in kernel completion, whereas both kernels and loss functions are approximated in the central SVM approach. Lastly, we can use both ADDITIVE and MULTIPLICATIVE kernels, but only ADDITIVE kernels are allowed in the central SVM approach.

4.2 Consensus-based Distributed SVMs

Another closely related study is done by Forero, Cano, and Giannakis (Forero et al., 2010). The motivation of this work is very similar to ours, in the sense that it tries to construct a consensus SVM in a

distributed fashion, without having a central processing location. They have developed a fully distributed SVM training algorithm based on the alternating direction method of multipliers (Bertsekas and Tsitsiklis, 1997).

However, the consensus-based distributed SVM considers situations where *examples* are distributed over connected nodes, not *features* are distributed as in our work. Moreover, the consensus requirements are expressed as extra constraints in a distributed SVM optimization problem therein: in our case, consensus SVMs are obtained by making approximations in each node to a "global" kernel matrix that would have constructed if we have collected all features to a central location.

4.3 Multiple Kernel Learning

Our ADDITIVE kernel is closely related to the multiple kernel learning (MKL) approach. In MKL, we consider a convex combination of N kernel matrices:

$$k(\mathbf{x}_i,\mathbf{x}_j)=\sum_{n=1}^{N}\beta_n k_n(\mathbf{x}_i,\mathbf{x}_j),\ \beta_n\geq 0,\ \sum_{n=1}^{N}\beta_n=1\ .$$

MKL searches for the optimal mixing coefficients $\beta_1,\beta_2,\ldots,\beta_N$, as well as the optimal values of the SVM dual variables. This requires to solve a semidefinite program (Lanckriet et al., 2002), a quadratically constrained quadratic program (Lanckriet et al., 2004) when we normalize kernels so that $k_n(\mathbf{x}_i,\mathbf{x}_i)=1$, or a quadratic program (Rakotomamonjy et al., 2007) with further modifications.

In our ADDITIVE approach (4), we use fixed mixing coefficients to $\beta_n=1/N$, in order to avoid storing and completing individual local kernel matrices. We could replace our SVM training with an MKL problem, and it might have a benefit to identify unimportant nodes that could be excluded from future communication, but MKL will impose overhead in computation and communication which may not be affordable.

4.4 Theory of Matrix Completion

Matrix completion provides guarantees under certain conditions to recover the original full matrix using only a few entries from it. Here we introduce the idea following Candès and Recht (Candès and Recht, 2009; Candès and Recht, 2012).

Going back to the matrix completion problem (6), we have defined a matrix $\mathbf{M}\in\Re^{m\times m}$ with rank r, and a sample set Ω such that for $(i,j)\in\Omega$, the components $\mathbf{M}_{ij}$ are known to us. The goal is to recover the rest of the matrix $\mathbf{M}$. Let us consider the reduced singular value decomposition of $\mathbf{M}$,

$$\mathbf{M}=\mathbf{U}\Sigma\mathbf{V}^T,\ \mathbf{U}^T\mathbf{U}=\mathbf{I},\ \mathbf{V}^T\mathbf{V}=\mathbf{I},$$

where $\Sigma\in\Re^{r\times r}$ is a diagonal matrix with singular values. The columns of $\mathbf{U}\in\Re^{m\times r}$ and $\mathbf{V}\in\Re^{m\times r}$ compose orthonormal bases of $\mathcal{R}(\mathbf{M})$ and $\mathcal{R}(\mathbf{M}^T)$, respectively, where $\mathcal{R}(\mathbf{X})$ denotes the range (column space) of a matrix $\mathbf{X}$. Based on these, we define a measure called the *coherence* of $\mathcal{R}(\mathbf{M})$ (Candès and Recht, 2009):

Definition For $\mathbf{M}=\mathbf{U}\Sigma\mathbf{V}^T$, the coherence of $\mathcal{R}(\mathbf{M})$ is defined by

$$\mathrm{co}(\mathcal{R}(\mathbf{M})):=\frac{m}{r}\max_{i=1,2,\ldots,m}\|\mathbf{U}\mathbf{U}^T\mathbf{e}_i\|^2\ \in[1,m/r]\ ,$$

where $\mathbf{e}_i$ is the ith standard unit vector.

Here $\mathbf{U}\mathbf{U}^T$ defines the projection matrix onto $\mathcal{R}(\mathbf{M})$. Coherence $\mathrm{co}(\mathcal{R}(\mathbf{M}))$ measures the alignment between the range space of $\mathbf{M}$ and any of the standard unit vectors. That is, the maximal coherence m/r is achieved whenever $\mathcal{R}(\mathbf{M})$ contains any of the standard basis vector $\mathbf{e}_i$, $i=1,2,\ldots,m$. On the other hand, coherence decreases as the basis vectors of $\mathcal{R}(\mathbf{M})$ becomes more like random vectors. For example, suppose that $\mathbf{U}$ contains uniform random column vectors, i.e. the value of each entry is $O(1/m)$ in magnitude satisfying $\mathbf{U}\mathbf{U}^T=\mathbf{I}$. Then we have $\|\mathbf{U}\mathbf{U}^T\mathbf{e}_i\|^2=\|\mathbf{U}^T\mathbf{e}_i\|^2=O(r/m)$ for any i which gives the minimum coherence value, using the fact that $\mathbf{U}\mathbf{U}^T=\mathbf{I}$ and $\mathbf{U}^T\mathbf{e}_i\in\Re^r$. Repeating the same argument for $\mathbf{V}$, we see that $\mathbf{M}=\mathbf{U}\Sigma\mathbf{V}^T$ is likely to be a dense matrix if both $\mathrm{co}(\mathcal{R}(\mathbf{M}))$ and $\mathrm{co}(\mathcal{R}(\mathbf{M}^T))$ are small. That is, it becomes harder that many entries of $\mathbf{M}$ becomes zero, which is a necessary property for matrix completion so that recovery would be possible from only a few sampled entries (otherwise they will contain many zero entries which are non-informative).

The next theorem states the required conditions of $\mathbf{M}$ and the estimated size of the sample set Ω, so that matrix completion will succeed with high probability.

Theorem 4.1 (Candès and Recht, 2009)**.** *For a matrix $\mathbf{M}=\mathbf{U}\Sigma\mathbf{V}^T\in\Re^{m\times m}$ of rank r, suppose that there exists constants $\delta_0>0$ and $\delta_1>0$ such that*

$$(i)\ \ \max\{\mathrm{co}(\mathcal{R}(\mathbf{M})),\mathrm{co}(\mathcal{R}(\mathbf{M}^T))\}\leq\delta_0\ ,$$
$$(ii)\ \ \max_{i,j}|[\mathbf{U}\mathbf{V}^T]_{ij}|\leq\delta_1\sqrt{r}/m\ .$$

If we sample $|\Omega|$ elements of $\mathbf{M}$ uniformly at random, as many as

$$|\Omega|\geq\psi\max(\delta_1^2,\delta_0^{0.5}\delta_1,\delta_0 m^{0.25})mr(\beta\log m)$$

for some constants ψ and $\beta>2$, then the minimizer of the matrix completion problem (6) *is unique and equal to the original $\mathbf{M}$ with probability at least $1-zm^{-\beta}$ for some constant z. If rank is small, that $r\leq m^{0.2}/\delta_0$, then the requirement reduces to*

$$|\Omega|\geq\psi\delta_0 m^{1.2}r(\beta\log m)\ .$$

A natural conjecture from this theorem is that Gaussian kernels would fit well for matrix completion, as they typically produces dense and numerically low-rank matrices (note that they are always full-rank in theory), whose entries are bounded above by 1. We use this theorem in the following section to check how well kernel matrices constructed from various data sets satisfy the required conditions for matrix completion, and how they affect the prediction performance of the resulting SVMs.

5 EXPERIMENTS

For experiments, we used five benchmark data sets from the UCI machine learning repository (Bache and Lichman, 2013), 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.

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 $N = 10$ (for full data sets) nodes. 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 (Lee et al., 2012), the local Gaussian kernel parameters for ADDITIVE 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 $O(\gamma p)$ as $\gamma\|\mathbf{x}_i - \mathbf{x}_j\|$.

Throughput the experiments, we imposed that if $(i, j) \in \Omega$ then $(j, i) \in \Omega$ as well, and $\mathbf{M}_{ij} = \mathbf{M}_{ji}$.

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 exact kernel matrices defined in (2), MULTIPLICATIVE and ADDITIVE, accessing all features of the small 5k/5k subsets of the five UCI data sets (the MULTIPLICATIVE kernels were equivalent to the usual Gaussian kernels).

The important characteristics of the kernel matrices with respect to matrix completion are its rank (r), coherence ($\delta_0 \in [1, m/r]$), and the maximal value of $|[\mathbf{U}\mathbf{V}^T]_{ij}|$ (where $\mathbf{U}$ and $\mathbf{V}$ are the left and right factors from singular value decomposition), as discussed in Theorem 4.1. When δ_0 is closer to its smallest value of one, and $|[\mathbf{U}\mathbf{V}^T]_{ij}|$ is bounded above by a small value, then matrix completion becomes well-posed. Further, if the rank is small as well, then the theorem indicates that we can recover the original matrix from even smaller samples.

Table 2 summarizes these characteristics. Clearly, the rank (numerically effective rank, with eigenvalues larger than a threshold of 0.01) and coherence values were much smaller in case of ADDITIVE, indicating potential benefits of using this approach compared to MULTIPLICATIVE. All numbers of $|[\mathbf{U}\mathbf{V}^T]_{ij}|$ appeared to be small, especially for the ADDITIVE kernels of ADULT, IJCNN, and COVTYPE. Kernel matrices of these three sets also had much lower ranks than the rest. For MNIST and CCAT, the numbers hinted that matrix completion would suffer from difficulties, unless the sample set $|\Omega|$ was large.

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 set Ω. We define the sampling ratio as

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

where the value of m is 5000 in this section. We compared the prediction performance of using MULTIPLICATIVE and ADDITIVE 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, 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 Figure 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 spec-

Table 2: The density, rank, coherence (δ_0), and maximal values of $|[\mathbf{U}\mathbf{V}^T]_{ij}|$ of kernel matrices. Effective numbers of ranks are shown, which correspond to eigenvalues larger than a threshold (0.01). Coherence values are bounded by $1 \leq \delta_0 \leq m/r$. Smaller values of δ_0, r, and $\max|[\mathbf{U}\mathbf{V}^T]_{ij}|$ are indicative of better conditions for matrix completion.

	MULTIPLICATIVE					ADDITIVE				
	density	r	δ_0	m/r	$\max\|[\mathbf{U}\mathbf{V}^T]_{ij}\|$	density	r	δ_0	m/r	$\max\|[\mathbf{U}\mathbf{V}^T]_{ij}\|$
ADULT	1.0	789	5.54	6.34	0.87	1.0	222	8.32	22.52	0.37
MNIST	1.0	4782	1.03	1.05	0.99	1.0	4568	1.07	1.10	0.98
CCAT	1.0	4984	1.00	1.00	1.00	1.0	4982	1.00	1.00	1.00
IJCNN	1.0	1516	3.19	3.30	0.97	1.0	698	1.75	7.16	0.25
COVTYPE	1.0	1423	3.32	3.51	0.95	1.0	424	1.56	11.79	0.13

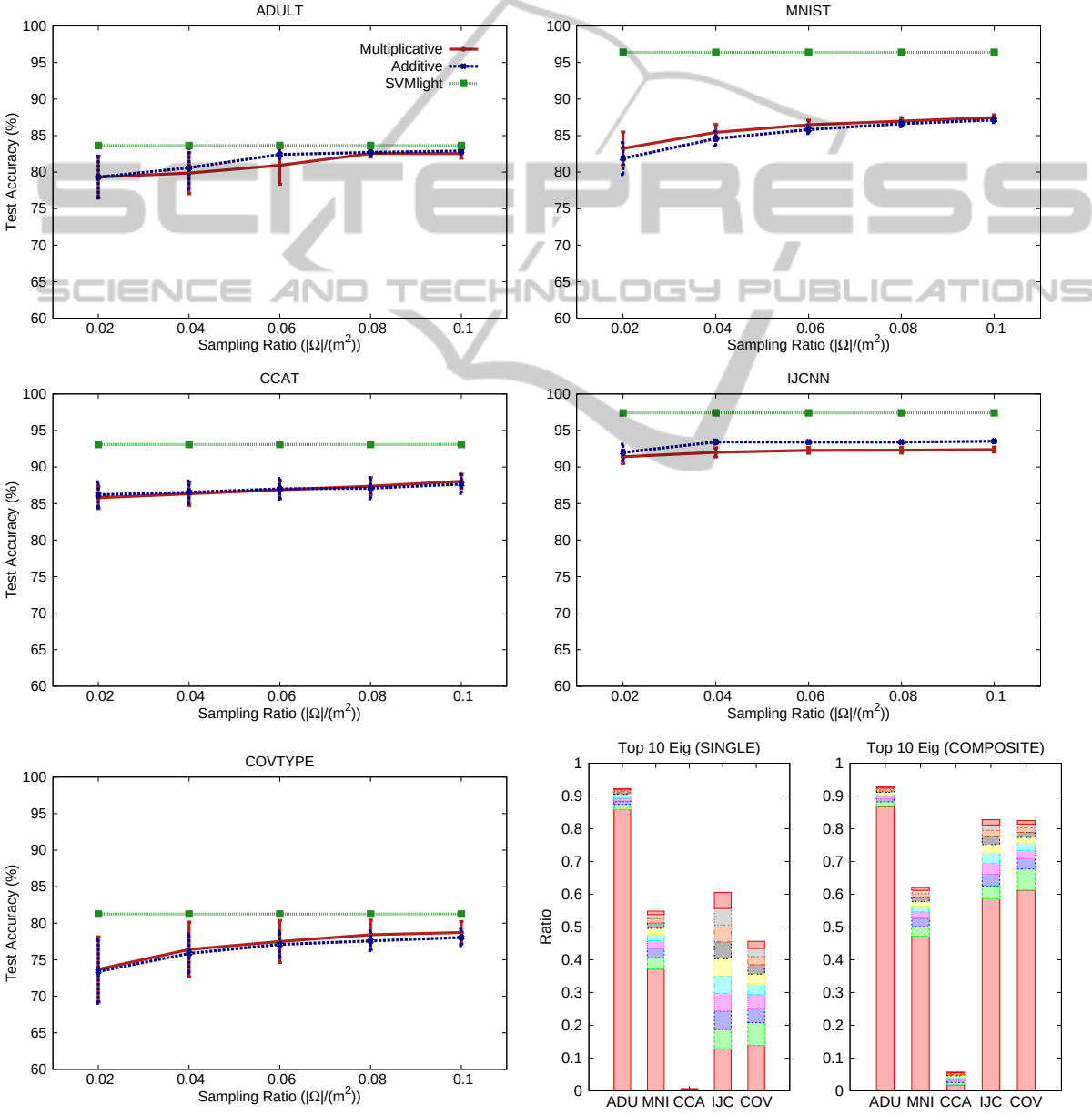

Figure 2: Prediction accuracy on test sets for 5k/5k subsets of the five UCI data sets, over different sampling ratios in kernel completion. The average and standard deviation over multiple trials with random Ω and $N = 3$ nodes are shown. The bottom-right plot illustrates the proportion of the entire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

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. $|\cup_n S_n^*|/m$ is the fraction of the union support vector sets to their corresponding training sets.

	ADDITIVE			ASSET	SVMLIGHT
	$\lvert\cup_n S_n^*\rvert/m$	2%	10%		
ADULT	0.61	81.4±1.00	84.2±0.18	80.0±0.02	84.9
MNIST	0.99	78.9±1.69	87.0±0.20	88.9±0.39	98.9
CCAT	0.84	87.2±1.00	92.0±0.35	73.7±1.00	95.8
IJCNN	0.56	96.0±0.35	96.5±0.23	90.9±0.88	99.3

trum in ADULT is concentrated in the top 10 eigenvalues, indicating that its kernel matrix has a very small numerically effective rank. This would be the reason why our method performed as good as SVMLIGHT for ADULT.

Comparing MULTIPLICATIVE to ADDITIVE, both showed similar prediction performance. However, higher concentration of the eigen spectrum of ADDITIVE indicated that it would make a good alternative to MULTIPLICATIVE, also considering the extra saving with ADDITIVE discussed in Section 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 (Lee et al., 2012), introduced in Section 4. Since ASSET admits only ADDITIVE kernels, we have omitted MULTIPLICATIVE in comparison. Among the several versions of ASSET in (Lee et al., 2012), we used the "Separate" version with central optimization. COVTYPE was excluded due to extra-long runtimes of SVMLIGHT and ours.

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 result was consistent to the discussion in Sections 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 further.

6 CONCLUSIONS

We have proposed a simple algorithm for learning consensus SVMs in sensor stations connected with band-limited communication channels. Our method makes use of decompositions of kernels, together with kernel completion to approximate unobserved entries of remote kernel matrices. The resulting SVMs performed well with relatively small numbers of sampled entries, when kernel matrices satisfied required conditions. A property of support vectors also helped us further reduce computational cost.

Using matrix completion, there is no need to identify and execute an optimal sampling strategy to have similar performance guarantees. Although sample complexity could be reduced by a small factor by identifying specific sample sets Ω for a given situation, such sets will depend on network topology and cost/noise models, perhaps with the need for central coordination.

Several aspects of our method remain to be investigated further. First, different types of kernels may involve different types of decomposition, having dissimilar characteristics in terms of matrix completion. Second, although parameters of SVMs can be tuned using small aggregated data, it would be desirable to tune parameters locally, or to consider parameter-free methods instead of SVMs. Also, despite the benefits of the ADDITIVE kernel, it requires more kernel parameters to be specified compared to the MULTIPLICATIVE kernel. Therefore when the budget for parameter tuning is limited, MULTIPLICATIVE would be preferred to ADDITIVE. 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, for instance.

Considering kernel completion in the context of privacy preserving learning would be an interesting branch, if the number of entries required for kernel completion to build a good classifier is less than the number required to recover private information, or if we can make kernel completion to fail unless it has right credentials by possibly tweaking the coherence

of kernel matrices.

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.

REFERENCES

Bache, K. and Lichman, M. (2013). UCI machine learning repository.

Bertsekas, D. P. and Tsitsiklis, J. N. (1997). *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth Annual Workshop on Computational Learning Theory*, pages 144–152.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122.

Candès, E. J. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772.

Candès, E. J. and Recht, B. (2012). Exact matrix completion via convex optimization. *Communications of the ACM*, 55(6):111–119.

Crammer, K., Dredze, M., and Pereira, F. (2012). Confidence-weighted linear classification for natural language processing. *Journal of Machine Learning Research*, 13:1891–1926.

Forero, P. A., Cano, A., and Giannakis, G. B. (2010). Consensus-based distributed support vector machines. *Journal of Machine Learning Research*, 11:1663–1707.

Ji, Y. and Sun, S. (2013). Multitask multiclass support vector machines: Model and experiments. *Pattern Recognition*, 46(3):914–924.

Joachims, T. (1999). Making large-scale support vector machine learning practical. In Schölkopf, B., Burges, C., and Smola, A., editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11, pages 169–184. MIT Press, Cambridge, MA.

Lanckriet, G., Cristianini, N., Bartlett, P., E. G., and L., Jordan, M. (2002). Learning the kernel matrix with semidefinite programming. In *Proceedings of the 19th International Conference on Machine Learning*.

Lanckriet, G. R. G., De Bie, T., Cristianini, N., Jordan, M. I., and Noble, W. S. (2004). A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635.

Lee, S. and Bockermann, C. (2011). Scalable stochastic gradient descent with improved confidence. In *Big Learning – Algorithms, Systems, and Tools for Learning at Scale*, NIPS Workshop.

Lee, S., Stolpe, M., and Morik, K. (2012). Separable approximate optimization of support vector machines for distributed sensing. In Flach, P., Bie, T., and Cristianini, N., editors, *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *Lecture Notes in Computer Science*, pages 387–402. Springer.

Lippi, M., Bertini, M., and Frasconi, P. (2010). Collective traffic forecasting. In *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part II*, pages 259–273.

Morik, K., Bhaduri, K., and Kargupta, H. (2012). Introduction to data mining for sustainability. *Data Mining and Knowledge Discovery*, 24(2):311–324.

Rakotomamonjy, A., Bach, F., Canu, S., and Grandvalet., Y. (2007). More efficiency in multiple kernel learning. In *Proceedings of the 24th International Conference on Machine Learning*.

Recht, B., Fazel, M., and Parrilo, P. A. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501.

Recht, B. and Ré, C. (2011). Parallel stochastic gradient algorithms for large-scale matrix completion. Technical report, University of Wisconsin-Madison.

Schoenberg, I. J. (1938). Metric spaces and positive definite functions. *Transactions of the American Mathematical Society*, 44(3):522–536.

Scholkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Shawe-Taylor, J. and Sun, S. (2011). A review of optimization methodologies in support vector machines. *Neurocomputing*, 74(17):3609–3618.

Stolpe, M., Bhaduri, K., Das, K., and Morik, K. (2013). Anomaly detection in vertically partitioned data by distributed core vector machines. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013*.

Whittaker, J., Garside, S., and Lindveld, K. (1997). Tracking and predicting a network traffic process. *International Journal of Forecasting*, 13(1):51–61.