Fast Parallel SVMs using Data Augmentation

Hugh Perkins, Minjie Xu, Jun Zhu and Bo Zhang
Department of Computer Science
Tsinghua University
Beijing, 100084 China
ngls11@mails.tsinghua.edu.cn,xuj-10@mails.tsinghua.edu.cn,
dcszj@mail.tsinghua.edu.cn,dcszb@mail.tsinghua.edu.cn

ABSTRACT

As one of the most popular classifiers, linear SVMs still have challenges in dealing with very large-scale problems, even though linear or sub-linear algorithms have been developed recently on single machines. Although parallel computing methods have also been developed to this end, most of them rely on solving local sub-optimization problems. In this paper, we develop a novel parallel algorithm for learning large-scale linear SVMs. Our approach is based on an equivalent formulation which casts the primal problem of SVMs as a Bayesian inference problem for whose solution we develop very efficient parallel EM and MCMC sampling algorithms. We provide empirical results for our algorithms, and provide extensions for SVR, nonlinear kernel SVMs, as well as the Crammer and Singer multi-class SVMs. Our approach is very promising in its own right and is also a very useful technique to parallelize a broader family of more general maximum-margin models.[1]

1. SVM AS BAYESIAN INFERENCE

In this section we present the fundamental theories our extensions and distributed algorithms are built upon.

1.1 SVM: the Basics

We first focus on standard linear SVMs for binary classification. Let $\mathcal{D} = \{(\mathbf{x}_d, y_d)\}_{d=1}^D$ be the training data, where $y_d \in \{1, -1\}$. The goal of SVMs is to learn a linear discriminant function

$$f(\mathbf{x}; \mathbf{w}, \nu) = \mathbf{w}^\top \mathbf{x} + \nu.$$

For notation simplicity, we absorb the offset parameter ν into $\mathbf{w}$ by introducing an additional feature dimension with fixed unit value. To find the optimal $\mathbf{w}$, the canonical learning problem of SVMs with a tolerance on training errors is formulated as a constrained optimization problem

$$\min_{\mathbf{w},\boldsymbol{\xi}} \ \frac{1}{2}\lambda\|\mathbf{w}\|_2^2 + 2\sum_d \xi_d$$
$$\text{s.t.}: \forall d, \ \begin{cases} y_d \mathbf{w}^\top \mathbf{x}_d \geq 1 - \xi_d \\ \xi_d \geq 0 \end{cases},$$

Note that the constant factor 2 in the training error term can be absorbed into λ, yet we leave it for the simplicity of the deduction later. Equivalently, the problem can be formulated as an unconstrained form

$$\min_{\mathbf{w}} \frac{1}{2}\lambda\|\mathbf{w}\|_2^2 + 2\sum_d \max(0, 1 - y_d \mathbf{w}^\top \mathbf{x}_d). \tag{1}$$

1.2 SVM: the MAP estimate

Problem (1) can also be viewed as a MAP estimate of a probabilistic model, where the posterior distribution is

$$p(\mathbf{w}|\mathcal{D}) \propto q_0(\mathbf{w}) q(\mathbf{y}|\mathbf{w}, \mathbf{X})$$

and $q_0(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \lambda^{-1} I)$ and $q(\mathbf{y}|\mathbf{w}, \mathbf{X}) = \prod_d q(y_d|\mathbf{w}, \mathbf{x}_d)$ with

$$q(y_d|\mathbf{w}, \mathbf{x}_d) = \exp(-2\max(0, 1 - y_d \mathbf{w}^\top \mathbf{x}_d)). \tag{2}$$

Note that we factorize the posterior into q_0 and q merely for the simplicity of subsequent denotation and that normally they can be intrinsically different from the *genuine* prior and likelihood as can be induced from the probabilistic model (even up to a constant factor). Hence we call q_0 and q *dele*-prior and *dele*-likelihood respectively ("dele" for "delegate").

The benefit of the MAP formulation is that it allows us to take advantage of many existing techniques developed for inference in probabilistic models and thus grants more flexibility for the solution. Specifically, Polson and Scott [1] show that the dele-likelihood can be represented as a scale mixture of Gaussians, namely

Lemma 1. *Scale mixture for hinge loss*

$$\exp(-2\max(0, 1 - y_d \mathbf{w}^\top \mathbf{x}_d)) = \int_0^\infty \frac{1}{\sqrt{2\pi\gamma_d}} \exp\Big(-\frac{(1 + \gamma_d - y_d \mathbf{w}^\top \mathbf{x}_d)^2}{2\gamma_d}\Big) d\gamma_d \tag{3}$$

This directly inspires an augmented representation with $\boldsymbol{\gamma} =$

[1]**This is an excerpt from the unpublished full version. Here we provide the core technical section for reference. MATLAB code is available here: http://ml.cs.tsinghua.edu.cn/~minjie/code.shtml**

$(\gamma_1, \ldots, \gamma_D)$ such that

$$p(\mathbf{w}, \boldsymbol{\gamma}|\mathcal{D}) \propto q_0(\mathbf{w}) \prod_d q(y_d, \gamma_d|\mathbf{w}, \mathbf{x}_d)$$

$$q(y_d|\mathbf{w}, \mathbf{x}_d) = \int_0^\infty q(y_d, \gamma_d|\mathbf{w}, \mathbf{x}_d) d\gamma_d$$

$$q(y_d, \gamma_d|\mathbf{w}, \mathbf{x}_d) = \phi(1 - y_d \mathbf{w}^\top \mathbf{x}_d| - \gamma_d, \gamma_d)$$

where $\phi(\cdot|\mu, \sigma^2)$ is the Gaussian density function.

1.3 MCMC Sampling for SVM

Based on this augmented representation, we are able to design MCMC methods for $p(\mathbf{w}, \boldsymbol{\gamma}|\mathcal{D})$, from which the optimal SVM solution that maximizes $p(\mathbf{w}|\mathcal{D})$ is relatively more probable to get sampled.

Specifically, we use Gibbs sampling and have the following conditional distributions [1]

$$p(\mathbf{w}|\boldsymbol{\gamma}, \mathcal{D}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{4}$$

$$p(\gamma_d^{-1}|\mathbf{w}, y_d, \mathbf{x}_d) = \mathcal{IG}(|1 - y_d \mathbf{w}^\top \mathbf{x}_d|^{-1}, 1), \tag{5}$$

where

$$\boldsymbol{\Sigma} = \Big(\lambda I + \sum_d \frac{1}{\gamma_d} \mathbf{x}_d \mathbf{x}_d^\top\Big)^{-1}, \ \boldsymbol{\mu} = \boldsymbol{\Sigma}\Big(\sum_d y_d (1 + \frac{1}{\gamma_d}) \mathbf{x}_d\Big) \tag{6}$$

and $\mathcal{IG}$ is the inverse Gaussian distribution [1].

1.4 EM algorithm for SVM

The EM algorithm is useful when directly maximizing the posterior $p(\mathbf{w}|\mathcal{D})$ is intractable but it is easy to alternate between the following two steps which converges to a local maximum of the posterior.

$$\text{E-step: } Q^{(m)}(\mathbf{w}) = \int \log p(\mathbf{w}, \boldsymbol{\gamma}|\mathcal{D}) p(\boldsymbol{\gamma}|\mathcal{D}, \mathbf{w}^{(m)}) d\boldsymbol{\gamma} \tag{7}$$

$$\text{M-step: } \mathbf{w}^{(m+1)} = \operatorname*{argmax}_{\mathbf{w}} Q^{(m)}(\mathbf{w}) \tag{8}$$

One can prove that the algorithm above monotonically increases the genuine posterior distribution of interest $p(\mathbf{w}|\mathcal{D})$ after each iteration, just as traditional EM does likelihood.

Deduction details omitted to save space, we summarize the results as follows

$$\text{E-step (update } \boldsymbol{\gamma}\text{):} \quad \gamma_d^{(m)} = |1 - y_d \mathbf{w}^{(m)\top} \mathbf{x}_d| \tag{9}$$

$$\text{M-step (update } \mathbf{w}\text{):} \quad \mathbf{w}^{(m+1)} = \boldsymbol{\mu}^{(m+1)}(\boldsymbol{\gamma}^{(m)}) \tag{10}$$

where $\boldsymbol{\mu}$ is calculated just as in Eq. (6).

Although normally EM is not guaranteed to obtain the global optimum, for our specific $p(\mathbf{w}|\mathcal{D})$ which is concave w.r.t $\mathbf{w}$, global optimum is actually expected. Furthermore, EM is a deterministic algorithm and enjoys a straightforward stopping criterion when compared with MCMC sampling.

2. EXTENSIONS

In this section we extend the idea above to SVR, nonlinear kernel SVMs, and the Crammer and Singer multi-class SVMs.

2.1 Learning Nonlinear Kernel SVMs

According to the representer theorem, the solution to problem (1) has the form

$$\mathbf{w} = \sum_d \alpha_d y_d \mathbf{x}_d, \tag{11}$$

which is a linear combination of $\mathbf{X}$. We can naturally extend it to the nonlinear case by using a feature mapping function h and learn the nonlinear SVM by solving

$$\min_{\mathbf{w}} \frac{1}{2}\lambda \|\mathbf{w}\|_2^2 + 2\sum_d \max(0, 1 - y_d \mathbf{w}^\top h(\mathbf{x}_d)), \tag{12}$$

the solution to which can be represented accordingly as

$$\mathbf{w} = \sum_d \alpha_d y_d h(\mathbf{x}_d) = H \operatorname{diag}(\mathbf{y}) \boldsymbol{\alpha}, \tag{13}$$

where $H = [h(\mathbf{x}_1) \ h(\mathbf{x}_2) \ \cdots \ h(\mathbf{x}_D)]$.

Substituting Eq. (13) into (12), we get the dual problem

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\lambda \boldsymbol{\alpha}^\top \operatorname{diag}(\mathbf{y}) K \operatorname{diag}(\mathbf{y}) \boldsymbol{\alpha} + 2\sum_d \max(0, 1 - y_d \boldsymbol{\alpha}^\top \operatorname{diag}(\mathbf{y}) K_d^\top), \tag{14}$$

where K is the Gram matrix and K_d is the dth row. If the feature map function h is a reproducing kernel, i.e., $h(\mathbf{x}) = k(\cdot, \mathbf{x})$, problem (14) becomes a kernel SVM and each entry of K is a dot product, that is

$$K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = h(\mathbf{x}_i)^\top h(\mathbf{x}_j).$$

The Gram matrix K is positive definite for any reproducing kernel, e.g. the most commonly used Gaussian kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\Big(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\Big)$$

Let $\boldsymbol{\omega} = \operatorname{diag}(\mathbf{y})\boldsymbol{\alpha}$, then $\mathbf{w} = \sum_d \omega_d h(\mathbf{x}_d)$ and the problem becomes

$$\min_{\boldsymbol{\omega}} \frac{1}{2}\lambda \boldsymbol{\omega}^\top K \boldsymbol{\omega} + 2\sum_d \max(0, 1 - y_d \boldsymbol{\omega}^\top K_d^\top), \tag{15}$$

Observing the similarity between problem (15) and (1), we reformulate it as MAP just as we did (1), with $q_0(\boldsymbol{\omega}) = \mathcal{N}(0, (\lambda K)^{-1})$ and $q(\mathbf{y}|\boldsymbol{\omega}, \mathbf{X}) = \prod_d q(y_d|\boldsymbol{\omega}, \mathbf{x}_d)$, where

$$q(y_d|\boldsymbol{\omega}, \mathbf{x}_d) = \exp(-2\max(0, 1 - y_d \boldsymbol{\omega}^\top K_d^\top)). \tag{16}$$

LEMMA 2. *Scale mixture for kernel hinge loss*

$$\exp(-2\max(0, 1 - y_d \boldsymbol{\omega}^\top K_d^\top)) = \int_0^\infty \frac{1}{\sqrt{2\pi\gamma_d}} \exp\Big(-\frac{(1 + \gamma_d - y_d \boldsymbol{\omega}^\top K_d^\top)^2}{2\gamma_d}\Big) d\gamma_d \tag{17}$$

Consequently for kernel SVMs, we have

$$q(\boldsymbol{\omega}|\boldsymbol{\gamma}, \mathcal{D}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{18}$$

$$p(\gamma_d^{-1}|\mathbf{w}, y_d, \mathbf{X}) = \mathcal{IG}(|\ell - y_d \boldsymbol{\omega}^\top K_d^\top|^{-1}, 1), \tag{19}$$

where

$$\boldsymbol{\Sigma} = \Big(\lambda K + \sum_d \frac{1}{\gamma_d} K_d^\top K_d\Big)^{-1}, \ \boldsymbol{\mu} = \boldsymbol{\Sigma}\Big(\sum_d y_d (1 + \frac{1}{\gamma_d}) K_d^\top\Big).$$

2.2 Support Vector Regression

For regression, where the response variable y are real-valued, the support vector regression (SVR) problem is defined as minimizing a regularized ϵ-insensitive loss [2]

$$\min_{\mathbf{w}} \frac{1}{2}\lambda \|\mathbf{w}\|_2^2 + 2\sum_d \max(0, |y_d - \mathbf{w}^\top \mathbf{x}_d| - \epsilon), \tag{20}$$

where ϵ is the precision parameter[2].

Naturally, we obtain the same q_0 as SVMs and

$$q(y_d|\mathbf{w},\mathbf{x}_d) = \exp(-2\max(0, |y_d - \mathbf{w}^\top \mathbf{x}_d| - \epsilon)), \quad (21)$$

and the augmentation is carried out by the following lemma

LEMMA 3. *Double scale mixture for ϵ-insensitive loss*

$$\begin{aligned}&\exp(-2\max(0, |y_d - \mathbf{w}^\top \mathbf{x}_d| - \epsilon))\\ =&\int_0^\infty \frac{1}{\sqrt{2\pi\gamma_d}} \exp\Big(-\frac{(\gamma_d + y_d - \mathbf{w}^\top \mathbf{x}_d - \epsilon)^2}{2\gamma_d}\Big) d\gamma_d\\ \times&\int_0^\infty \frac{1}{\sqrt{2\pi\omega_d}} \exp\Big(-\frac{(\omega_d - y_d + \mathbf{w}^\top \mathbf{x}_d - \epsilon)^2}{2\omega_d}\Big) d\omega_d \quad (22)\end{aligned}$$

PROOF. As $\epsilon \geq 0$, the following equality holds

$$\begin{aligned}&\max(0, |y_d - \mathbf{w}^\top \mathbf{x}_d| - \epsilon)\\ =&\max(0, y_d - \mathbf{w}^\top \mathbf{x}_d - \epsilon) + \max(0, -y_d + \mathbf{w}^\top \mathbf{x}_d - \epsilon). \quad (23)\end{aligned}$$

Therefore, for each term, we can do similar derivation as in Lemma 1 to get the double scale mixture formulation. □

Consequently for SVR, we have

$$p(\mathbf{w}|\boldsymbol{\gamma}, \boldsymbol{\omega}, \mathcal{D}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma) \quad (24)$$

$$p(\gamma_d^{-1}|\mathbf{w}, \boldsymbol{\omega}, y_d, \mathbf{x}_d) = \mathcal{IG}(|y_d - \mathbf{w}^\top \mathbf{x}_d - \epsilon|^{-1}, 1) \quad (25)$$

$$p(\omega_d^{-1}|\mathbf{w}, \boldsymbol{\gamma}, y_d, \mathbf{x}_d) = \mathcal{IG}(|y_d - \mathbf{w}^\top \mathbf{x}_d + \epsilon|^{-1}, 1), \quad (26)$$

where the covariance and mean are now

$$\boldsymbol{\Sigma} = \Big(\lambda I + \sum_d (\frac{1}{\gamma_d} + \frac{1}{\omega_d}) \mathbf{x}_d \mathbf{x}_d^\top\Big)^{-1}, \quad (27)$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma}\Big(\sum_d (\frac{y_d - \epsilon}{\lambda_d} + \frac{y_d + \epsilon}{\omega_d}) \mathbf{x}_d\Big). \quad (28)$$

2.3 Learning Multi-class SVMs

For multi-class classification, we have $y_d \in \{1, \cdots, M\}$. There are various strategies to perform multi-class classification with SVMs. Here we consider the approach proposed by Crammer and Singer (2001), where the generalized discriminant function is defined to be

$$f(y, \mathbf{x}; \mathbf{w}) = \mathbf{w}_y^\top \mathbf{x} \quad (29)$$

where $\mathbf{w}_y$ is the sub-vector corresponding to class label y. And the regularized risk minimization problem becomes

$$\min_{\mathbf{w}} \frac{1}{2}\lambda \|\mathbf{w}\|_2^2 + 2\sum_d \max_y (\Delta_d(y) - \Delta f_d(y; \mathbf{w})), \quad (30)$$

where $\Delta_d(y)$ is the cost of predicting y for the true label y_d and $\Delta f_d(y; \mathbf{w}) = f(y_d, \mathbf{x}_d; \mathbf{w}) - f(y, \mathbf{x}_d; \mathbf{w})$ is the margin. Both $\Delta_d(y)$ and $\Delta f_d(y; \mathbf{w})$ equal zero when $y = y_d$.

Then, the pseudo-prior and pseudo-likelihood is changed accordingly to

$$q_0(\mathbf{w}) = \prod_y q_0(\mathbf{w}_y) = \prod_y \mathcal{N}(\mathbf{w}_y|\mathbf{0}, \lambda^{-1} I) \quad (31)$$

$$q(y_d|\mathbf{w}, \mathbf{x}_d) = \exp(-2\max_y (\Delta_d(y) + \mathbf{w}_y^\top \mathbf{x}_d - \mathbf{w}_{y_d}^\top \mathbf{x}_d)) \quad (32)$$

In order for Lemma 1 to be applicable, we resort to an iterative procedure, which alternately infer weights $\mathbf{w}_y$ given the other weights $\mathbf{w}_{-y}$, for each class label y.

[2] ϵ is a small positive number, e.g., $1e^{-3}$ in our experiments

The local conditional distribution is

$$p(\mathbf{w}_y|\mathcal{D}, \mathbf{w}_{-y}) \propto q_0(\mathbf{w}_y) \prod_d \psi(\mathbf{w}_y; \mathbf{w}_{-y}, y_d, \mathbf{x}_d), \quad (33)$$

where $\psi(\mathbf{w}_y; \mathbf{w}_{-y}, y_d, \mathbf{x}_d) \propto q(y_d|\mathbf{w}, \mathbf{x}_d)$

$$\begin{aligned}&= \exp(-2(\max(\mathbf{w}_y^\top \mathbf{x}_d + \Delta_d(y), \zeta_d(y)) - \mathbf{w}_{y_d}^\top \mathbf{x}_d))\\ &\propto \begin{cases}\exp(-2\max(\mathbf{w}_y^\top \mathbf{x}_d - \rho_d^y, 0)) & (y \neq y_d)\\ \exp(-2\max(0, \rho_d^y - \mathbf{w}_y^\top \mathbf{x}_d)) & (y = y_d)\end{cases} \quad (34)\\ &= \exp(-2\max(0, \beta_d^y(\rho_d^y - \mathbf{w}_y^T \mathbf{x}_d))) \quad (35)\end{aligned}$$

where $\zeta_d(y) = \max_{y' \neq y}(\mathbf{w}_{y'}^\top \mathbf{x}_d + \Delta_d(y'))$ is independent of $\mathbf{w}_y$, $\rho_d^y = \zeta_d(y) - \Delta_d(y)$ and $\beta_d^y = \begin{cases}+1 & \text{for } y = y_d\\ -1 & \text{for } y \neq y_d\end{cases}$.

Hence we take

$$\psi(\mathbf{w}_y; \mathbf{w}_{-y}, y_d, \mathbf{x}_d) = \exp(-2\max(0, \beta_d^y(\rho_d^y - \mathbf{w}_y^T \mathbf{x}_d)))$$

and through a similar augmentation, we obtain the Gibbs sampling step for each augmented local conditional distribution $p(\mathbf{w}_y, \boldsymbol{\gamma}_y|\mathcal{D}, \mathbf{w}_{-y})$

$$p(\gamma_{yd}^{-1}|\mathbf{w}, y_d, \mathbf{x}_d) = \mathcal{IG}(|\rho_d^y - \mathbf{w}_y^\top \mathbf{x}_d|^{-1}, 1), \quad (36)$$

$$p(\mathbf{w}_y|\boldsymbol{\gamma}_y, \mathbf{w}_{-y}, \mathcal{D}) = \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y) \quad (37)$$

where

$$\boldsymbol{\Sigma}_y = \Big(\lambda I + \sum_d \frac{1}{\gamma_{yd}} \mathbf{x}_d \mathbf{x}_d^\top\Big)^{-1}, \quad (38)$$

$$\boldsymbol{\mu}_y = \boldsymbol{\Sigma}_y\Big(\sum_d (\frac{\rho_d^y}{\gamma_{yd}} + \beta_d^y) \mathbf{x}_d\Big). \quad (39)$$

Note that this is actually a hierarchical Gibbs sampling

1. to sample $p(\mathbf{w}|\mathcal{D})$, we carry out Gibbs sampling over $p(\mathbf{w}_y|\mathcal{D}, \mathbf{w}_{-y})$ alternately for $y = 1, \ldots, M$;
2. to sample each $p(\mathbf{w}_y|\mathcal{D}, \mathbf{w}_{-y})$, we use data augmentation to sample over $p(\mathbf{w}_y, \boldsymbol{\gamma}_y|\mathcal{D}, \mathbf{w}_{-y})$.

Accordingly, the EM algorithm for Crammer and Singer multi-class SVMs inherits this 2-layer structure:

1. to maximize $p(\mathbf{w}|\mathcal{D})$, we carry out blockwise coordinate descent to maximize $p(\mathbf{w}_y|\mathcal{D}, \mathbf{w}_{-y})$ alternately;
2. to maximize each $p(\mathbf{w}_y|\mathcal{D}, \mathbf{w}_{-y})$, we adopt the EM algorithm where

$$Q^{(m)}(\mathbf{w}_y) = \int \log p(\mathbf{w}_y, \boldsymbol{\gamma}_y|\mathcal{D}, \mathbf{w}_{-y}) p(\boldsymbol{\gamma}_y|\mathcal{D}, \mathbf{w}_y^{(m)}, \mathbf{w}_{-y}) d\boldsymbol{\gamma}_y$$

3. PARALLEL SVM

Below we show how to employ distributed computing into the sampling algorithms above. We focus on the classical linear binary SVMs for the ease of explanation. And exactly the same techniques apply as well to all the extensions we present in section 2, and also their EM algorithms.

Two key properties of the sampling process that are in favor of parallel computation are summarized as follows.

1. The scale variables $\boldsymbol{\gamma}$ are mutually independent from each other, and therefore its sampling step can be easily parallelized to multiple cores and multiple machines.
2. The training data $(\mathbf{x}_d, y_d)$ contribute to the global variables $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ through a simple summation operator (Eq. (6)). Thus a typical map-reduce architecture is directly applicable, as shown in Figure 1.

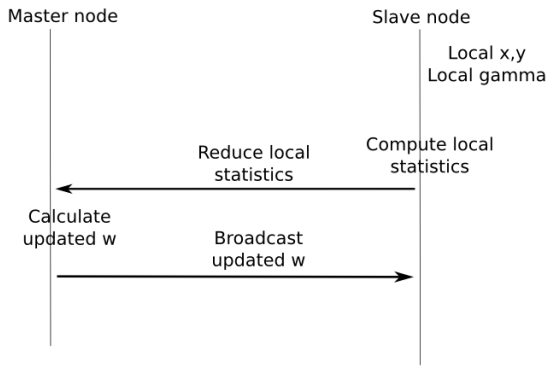

Figure 1: Map-reduce architecture for parallel sampling SVM

3.1 The Basic Procedure

Let P be the total number of processes and let $\mathcal{D}^p = \{(\mathbf{x}_d^p, y_d^p)\}_{d=1}^{D_p}$ be the data assigned to process p. Then each process performs the following computations

1. *draw scale parameters*: each p draws γ_{dp}^{-1} ($\forall 1 \leq d \leq D_p$) according to the distribution in Eq. (5).
2. *compute local statistics*: each p computes the following local statistics

$$\boldsymbol{\mu}^p = \sum_{d=1}^{D_p} (1 + \frac{1}{\gamma_{dp}}) y_d^p \mathbf{x}_d^p,$$
$$\boldsymbol{\Sigma}^p = \sum_{d=1}^{D_p} \frac{1}{\gamma_{dp}} \mathbf{x}_d^p \mathbf{x}_d^{p\top}. \quad (40)$$

Since $\boldsymbol{\Sigma}^p$ is symmetric, it suffices to compute only the upper or lower triangle and then submit to the master.

After process p has finished its local computation, it passes the local statistics $\boldsymbol{\mu}^p$ and $\boldsymbol{\Sigma}^p$ to the master process, which collects the results and performs the following aggregation operations

1. compute $\boldsymbol{\Sigma}^{-1} = \lambda I + \sum_p \boldsymbol{\Sigma}^p$.
2. after $\boldsymbol{\Sigma}^{-1}$ is updated, compute $\boldsymbol{\mu} = \boldsymbol{\Sigma}(\sum_p \boldsymbol{\mu}^p)$.

It is worth noting that all the slave processes perform exactly the same set of operations. Assume that we equally partition the large data set and all computing nodes are of the same capacity, then it can be expected that all the nodes have a high probability to finish their local job at roughly the same time. Therefore the latency due to synchronization is typically small. While in contrast, the existing parallel methods for SVMs by solving multiple smaller QP problems can suffer from large synchronization latency since the sub-QP problems varies a lot.

3.2 Notation

We will denote the parallel sampling SVM as PEMSVM. PEMSVM has the following options:

- linear ("LIN") vs kernelized ("KRN")
- EM ("EM") vs MCMC ("MC")

Step	Asymptotic time
Draw $\boldsymbol{\gamma}$	$O(NK/P)$
Calculate $\boldsymbol{\mu}_p$	$O(NK/P)$
Calculate $\boldsymbol{\Sigma}_p$	$O(NK^2/P)$
Reduce	$O(K^2 \log(P))$
Draw $\boldsymbol{\mu}$	$O(K^2 \log(K))$
Broadcast $\boldsymbol{\mu}$	$O(K^2 \log(P))$

Table 1: Asymptotic times for LIN-EM-CLS.

Step	Asymptotic time
Draw $\boldsymbol{\gamma}$	$O(N^2/P)$
Calculate $\boldsymbol{\mu}_p$	$O(N^2/P)$
Calculate $\boldsymbol{\Sigma}_p$	$O(N^3/P)$
Reduce	$O(N^2 \log(P))$
Draw $\boldsymbol{\mu}$	$O(N^2 \log(N))$
Broadcast $\boldsymbol{\mu}$	$O(N^2 \log(P))$

Table 2: Asymptotic times for KRN-EM-CLS.

- binary classification ("CLS") vs multiclass classification ("MLT") vs support vector regression ("SVR")

These three sets of options are orthogonal, so we can write a set of options for example as 'LIN-EM-CLS'.

N is the number of training instances, K is the number of features, M is the number of classes, and P is the number of processes.

3.3 Iteration time

3.3.1 EM

LIN.

LIN-EM-CLS comprises the steps shown in Table 1. Overall:

$$O(K^2[N/P + \log(P) + \log(K)])$$

Typically, the N/P term dominates, giving $O(NK^2/P)$, and parallelization is effective.

Where K or P are high, then the $\log(P)$ and $\log(K)$ terms can dominate. When this is the case, (further) parallelization is no longer effective.

KRN.

KRN-EM-CLS comprises the steps in 2. Overall:

$$O(N^2[N/P + \log(P) + \log(N)])$$

Typically, the N/P term dominates, giving $O(N^3/P)$, which shows effective parallelization.

When P or N are high, then the $\log(P)$ and $\log(N)$ terms can dominate, and (further) parallelization is no longer effective.

SVR.

The iteration time of SVR is asymptotically identical to CLS.

MLT.

The iteration time of MLT is multiplied by a factor of M, when compared to CLS.

3.3.2 MC

In MC, there is an additional stochastic sampling step for both $\boldsymbol{\gamma}$ and $\boldsymbol{\Sigma}$.

For LIN, the sampling step for $\boldsymbol{\gamma}$ is an inverse gamma on an N by 1 vector, which is $O(N)$. The sampling step for $\boldsymbol{\Sigma}$ involves a matrix inverse, on a K by K matrix. Even using Cholesky decomposition, this is asymptotically $O(K^3)$. Therefore, for MC, the asymptotic time becomes:

$$O(K^2[N/P + \log(P) + K])$$

Therefore, for LIN-MC, when K or $\log(P)$ become large relative to N/P, then (further) parallelization will no longer be effective.

4. REFERENCES

[1] N. G. Polson and S. L. Scott. Data augmentation for support vector machines. *Bayesian Analysis*, 6(1):1–24, 2011.

[2] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 2003.