Vietnam J Comput Sci (2014) 1:107–115
DOI 10.1007/s40595-013-0013-2

REGULAR PAPER

Parallel multiclass stochastic gradient descent algorithms for classifying million images with very-high-dimensional signatures into thousands classes

Thanh-Nghi Do

Received: 26 October 2013 / Accepted: 19 December 2013 / Published online: 21 January 2014

Abstract The new parallel multiclass stochastic gradient descent algorithms aim at classifying million images with very-high-dimensional signatures into thousands of classes. We extend the stochastic gradient descent (SGD) for support vector machines (SVM-SGD) in several ways to develop the new multiclass SVM-SGD for efficiently classifying large image datasets into many classes. We propose (1) a balanced training algorithm for learning binary SVM-SGD classifiers, and (2) a parallel training process of classifiers with several multi-core computers/grid. The evaluation on 1000 classes of ImageNet, ILSVRC 2010 shows that our algorithm is 270 times faster than the state-of-the-art linear classifier LIBLINEAR.

Keywords Support vector machine · Stochastic gradient descent · Multiclass · Parallel algorithm · Large-scale image classification

1 Introduction

Visual classification is one of the important research topics in computer vision and machine learning. Low-level local features and bag-of-words model (BoW) are the core of state-of-the-art visual classification systems. The usual pipeline for visual classification task involves three steps: (1) extracting features, (2) building codebook and encoding features, and (3) training classifiers. Most of the methods based on this pipeline have been only evaluated on small datasets, e.g. Caltech 101 [1], Caltech 256 [2], and PASCAL VOC [3] that can fit into desktop memory. In step 3, most researchers may choose either linear or non-linear support vector machines (SVM) classifiers that can be trained in a few minutes.

However, the emergence of ImageNet dataset [4] poses more challenges for training classifiers. ImageNet is much larger in scale and diversity than the other benchmark datasets. The current release ImageNet has grown a big step in terms of the number of images and the number of classes, as shown in Fig. 1—it has more than 14 million images for 21,841 classes (more than 1,000 images for each class on average).

With millions of images, training an accurate classifier may take weeks or even years [5,6]. Therefore, the recent works in large-scale learning classifiers have focused on building linear classifiers for large-scale visual classification tasks. In many test cases, linear SVM classifier is a trade-off between training time and classification accuracy [7]. Shalev-Shwartz et al. [8] and [9] propose stochastic gradient descent algorithms for SVM (denoted by SVM-SGD) that shows the promising results for large-scale binary classification problems. An extension of SVM-SGD [10] uses the one-versus-all strategy for dealing with large-scale images in very-high-dimensional signatures and thousands classes. However, the current version of SVM-SGD does not take into account the benefits of high-performance computing (HPC). On ILSVRC 2010, it takes very long time to train 1000 binary classifiers. Therefore, it motivates us to study how to speed-up SVM-SGD for large-scale visual classification tasks. In this paper, we extend the binary SVM-SGD in several ways to develop the new parallel multiclass SVM-SGD algorithms for efficiently classifying large image datasets into many classes. The idea is to build

1. A balanced training algorithm for binary SVM-SGD classifiers,

T.-N. Do (✉)
College of Information Technology, Can Tho University,
Can Tho, Vietnam
e-mail: dtnghi@cit.ctu.edu.vn

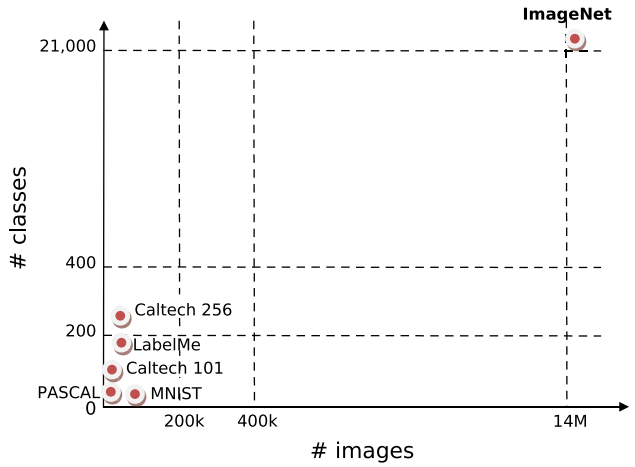

Fig. 1 A comparison of ImageNet with other benchmark datasets

2. A parallel training process of classifiers with several multi-core computers/grid.

The remainder of this paper is organized as follows: Sect. 2 briefly reviews the related work on large-scale visual classification. Section 3 introduces stochastic gradient descents for SVM. In Sect. 4, we present its improvement for large number of classes and describe how to speed-up the training process of SVM-SGD by using our balanced training algorithm and take into account the benefits of HPC. Section 5 presents numerical results before the conclusion and future work.

2 Related work

Many previous works on visual classification have relied on bag-of-words model, BoW [11], local feature quantization and SVM. These models may be enhanced by multi-scale spatial pyramids [12] on BoWs or histogram of oriented gradient [13] features. Some recent works consider exploiting the hierarchical structure of dataset for image recognition and achieve impressive improvements in accuracy and efficiency [14]. Related to classification is the problem of detection, often treated as repeated one-versus-all classification in sliding windows [3,15]. In many cases, such localization of objects might be useful for improving classification accuracy performance. However, in the context of large-scale visual classification with hundreds or thousands of classes, these common approaches become computationally intractable.

To address this problem, [16] study semi-supervised learning on 126 hand-labeled Tiny Images categories, [17] show classification experiments on a maximum of 315 categories. [18] do research with landmark classification on a collection of 500 landmarks and 2 million images. On a small subset of ten classes, they have improved BoW classification by increasing the visual vocabulary up to 80 K visual words. Furthermore, the current released ImageNet makes the complexity of large-scale visual classification become a big challenge. To tackle this challenge, many researchers are beginning to study strategies on how to improve the accuracy performance and avoid using high-cost nonlinear kernel SVMs for training classifiers. The recent prominent works for these strategies are proposed in [5,6,19,20] where the data are first transformed by a nonlinear mapping induced by a particular kernel and then efficient linear classifiers are trained in the resulting space. They argue that the classification accuracy of linear classifiers with high-dimensional image representations is similar to low-dimensional BoW with non-linear kernel classifiers. Therefore, many previous works in large-scale visual classification have converged on building linear classifiers using the state-of-the-art linear classifier LIBLINEAR [21]. However, the recent works in [8] and [9] show empirically that SVM-SGD is faster than LIBLINEAR on many benchmark datasets. The recent work [20] and [10] study the impact of high-dimensional Fisher vectors on large-datasets. They show that the larger the training dataset, the higher the impact of the dimensionality on the classification accuracy. To get the state-of-the-art results on ILSVRC 2010, they make use of the spatial pyramids to increase the dimensionality of Fisher vector and then exploit Product Quantizer [22] to compress the data before training classifiers. With this method, the training data can fit into the main memory of a single computer (48 GB). To training classifiers, they employ Stochastic Gradient Descent [9] with early stopping, reweighting data, regularization, step size and the computation of dot product is parallelized on 16 cores of their computer. Our approach is quite different with this work in which we train classifiers with a sampling strategy and a smooth hinge loss function and parallelize the training task of binary classifiers on several multi-core computers.

A grid with several multi-core computers bring to us many advantages. Advanced technologies designed for the systems where several processes have access to shared or distributed memory space are becoming popular choice for high-performance computing algorithms. Therefore, it motivates us to investigate parallel solutions and demonstrate how SVM-SGD can benefit from modern platforms. Furthermore, in the case of large number of classes, we propose the balanced training algorithm that can be applied to speed-up the training process of classifiers without compromising classification accuracy. Our experiments show very good results and confirm that the balanced training algorithm and parallel solutions are very essential for large-scale visual classification in terms of training time and classification accuracy.

3 Support vector machines and stochastic gradient descent

3.1 Support vector machines (SVM)

Let us consider a linear binary classification task, as depicted in Fig. 2, with m datapoints x_i $(i = 1, \ldots, m)$ in the n-dimensional input space R^n, having corresponding labels $y_i = \pm 1$. For this problem, the SVM algorithms [23] try to find the best separating plane (denoted by the normal vector $w \in R^n$ and the scalar $b \in R$), i.e. furthest from both class $+1$ and class -1. It can simply maximize the distance or margin between the supporting planes for each class ($x.w - b = +1$ for class $+1$, $x.w - b = -1$ for class -1). The margin between these supporting planes is $2/\|w\|$ (where $\|w\|$ is the 2-norm of the vector w). Any point x_i falling on the wrong side of its supporting plane is considered to be an error, denoted by z_i $(z_i \geq 0)$. Therefore, the SVM has to simultaneously maximize the margin and minimize the error. The standard SVMs pursue these goals with the quadratic programming of (1).

$$\begin{aligned} &\min\ \Psi(w, b, z) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} z_i \\ &s.t.: y_i(w.x_i - b) + z_i \geq 1 \\ &z_i \geq 0, \end{aligned} \tag{1}$$

where the positive constant C is used to tune errors and margin size.

The plane (w, b) is obtained by solving the quadratic programming (1). Then, the classification function of a new datapoint x based on the plane is

$$\text{predict}(x) = \text{sign}(w.x - b) \tag{2}$$

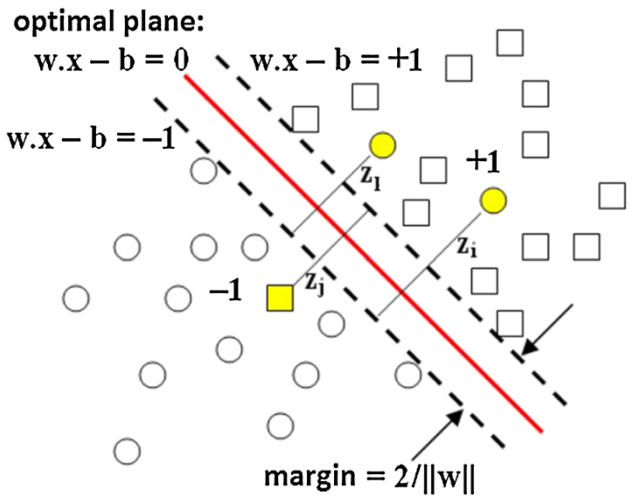

Fig. 2 Linear separation of the datapoints into two classes

SVM can use some other classification functions, for example a polynomial function of degree d, a RBF (Radial Basis Function) or a sigmoid function. To change from a linear to non-linear classifier, one must only substitute a kernel evaluation in (1) instead of the original dot product. More details about SVM and other kernel-based learning methods can be found in [24].

Unfortunately, the computational cost requirements of the SVM solutions in (1) are at least $O(m^2)$, where m is the number of training datapoints, making classical SVM intractable for large datasets.

3.2 SVM with stochastic gradient descent (SGD)

We can reformulate the SVM problem in quadratic programming (1) in an unconstraint problem. We can ignore the bias b without generality loss. The constraints $y_i(w.x_i) + z_i \geq 1$ in (1) are rewritten as follows:

$$z_i \geq 1 - y_i(w.x_i) \tag{3}$$

The constraints (3) and $z_i \geq 0$ are rewritten by the hinge loss function:

$$z_i = max\{0, 1 - y_i(w.x_i)\} \tag{4}$$

Substituting for z from the constraint in terms of w into the objective function Ψ of the quadratic programming (1) yields an unconstrained problem (5):

$$\min\ \Psi(w, [x, y]) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{m}\sum_{i=1}^{m} max\{0, 1 - y_i(w.x_i)\} \tag{5}$$

And then, [8,9] proposed the stochastic gradient descent method to solve the unconstrained problem (5). The stochastic gradient descent for SVM (denoted by SVM-SGD) updates w on T epochs with a learning rate η. For each epoch t, the SVM-SGD uses a single randomly received datapoint (x_i, y_i) to compute the sub-gradient $\nabla_t \Psi(w, [x_i, y_i])$ and update w_{t+1}.

As mentioned in [8,9], the SVM-SGD algorithm quickly converges to the optimal solution due to the fact that the unconstrained problem (5) is convex games on very large datasets. The algorithmic complexity of SVM-SGD is linear with the number of datapoints. An example of its effectiveness is given with the classification into two classes of 780,000 datapoints in 47,000-dimensional input space in 2 s on a PC and the test accuracy is similar to standard SVM.

4 Extentions of SVM-SGD to large number of classes

Most SVM algorithms are only able to deal with a two-class problem. There are several extensions of a binary classifi-

cation SVM solver to multi-class (k classes, $k \geq 3$) classification tasks. The state-of-the-art multi-class SVMs are categorized into two types of approaches. The first one is considering the multi-class case in one optimization problem [25–27]. The second one is decomposing multi-class into a series of binary SVMs, including one-versus-all [23], one-versus-one [28] and Decision Directed Acyclic Graph [29].

In practice, one-versus-all, one-versus-one are the most popular methods due to their simplicity. Let us consider k classes ($k > 2$). The one-versus-all strategy builds k different classifiers where the ith classifier separates the ith class from the rest. The one-versus-one strategy constructs $k(k-1)/2$ classifiers, using all the binary pairwise combinations of the k classes. The class is then predicted with a majority vote.

When dealing with very large number of classes, e.g. thousands classes, the one-versus-one strategy is too expensive because it needs to train millions binary classifiers. Therefore, the one-versus-all strategy becomes popular in this case. However, the multiclass SVM-SGD algorithm using one-versus-all leads to the two problems:

1. the SVM-SGD algorithm deals with the imbalanced datasets for building binary classifiers,
2. the SVM-SGD algorithms also takes very long time to train very large number of binary classifiers in sequential mode using a single processor.

A recent multiclass SVM-SGD algorithm proposed by [10] uses the one-versus-all strategy for classifying large-scale images in very-high-dimensional signatures and thousands classes. The recommendations for this algorithm include early stopping, reweighting used to adjust the sample data, regularization, step size.

And then, our multiclass SVM-SGD algorithm also uses the one-versus-all approach to train independently k binary classifiers. We propose two ways for creating the new multiclass SVM-SGD algorithm being able to handle very large number of classes in high speed. The first one is to build balanced training of binary classifiers with a sampling strategy and a smooth hinge loss function. The second one is to parallelize the training task of all classifiers with several multi-core machines/grids.

4.1 Balanced training SVM-SGD

In the one-versus-all approach, the learning task of SVM-SGD tries to separate the ith class (positive class) from the $k-1$ others classes (negative class). For very large number of classes, e.g. 1,000 classes, this leads to the extreme imbalance between the positive and the negative class. The problem is well known as the class imbalance. The problem of the SVM-SGD algorithm comes from the update rule using a random received datapoint. The probability for a positive datapoint sampled is very small (about 0.001) compared with the large chance for a negative datapoint sampled (e.g. 0.999). And then, the SVM-SGD concentrates mostly on the errors produced by the nagative datapoints. Therefore, the SVM-SGD has difficulty to separate the positive class from the rest.

As summarized by the review papers of [30–32], and the very comprehensive papers of [33,34], solutions to the class imbalance problems were proposed both at the data and algorithmic level. At the data level, these algorithms change the class distribution, including over-sampling the minority class [35] or under-sampling the majority class [36,37]. At the algorithmic level, the solution is to re-balance the error rate by weighting each type of error with the corresponding cost.

Our balanced training SVM-SGD simultaneously uses the two approaches. Furthermore, the class prior probabilities in this context are highly unequal (e.g. the distribution of the positive class is 0.1 % in the 1,000 classes classification problem), and over-sampling the minority class is very expensive. Therefore, our balanced training SVM-SGD uses under-sampling the majority class (negative class). The balanced training SVM-SGD also modifies the updating rule using the skewed misclassification costs.

Although the SVM-SGD algorithm has impressive convergence properties due to the fact that the unconstrained problem (5) is convex games on very large datasets. The hinge loss function $L(w, [x_i, y_i]) = \max\{0, 1 - y_i(w.x_i)\}$ in the unconstrained problem (5) is discontinuously in the derivative at $y_i(w.x_i) = 1$, and then the SGD's convergence rate still cannot be faster than $O(\ln(T)/T)$ as mentioned in [38]. One way to resolve this issue is to use a surrogate smooth loss function of the hinge loss function in the unconstrained problem (5), this leads to achieve the optimal rate $O(1/T)$, illustrated in [39,40]. Then, we propose to substitute the hinge loss $L(w, [x_i, y_i]) = max\{0, 1 - y_i(w.x_i)\}$ in the unconstrained problem (5) by the smooth hinge loss [41], as follows:

$$L^s(w, [x_i, y_i]) = \begin{cases} \frac{1}{2} - y_i(w.x_i) & y_i(w.x_i) \leq 0 \\ \frac{1}{2}[1 - y_i(w.x_i)]^2 & 0 < y_i(w.x_i) < 1 \\ 0 & 1 \leq y_i(w.x_i) \end{cases} \tag{6}$$

And then our balanced training SVM-SGD for binary classification tasks (described in 1) updates w on T epochs. For each epoch t, the reduced dataset D' is created by the full set of positive class D_+ and under-sampling the negative class D'_-, the SVM-SGD randomly picks a datapoint (x_i, y_i) from the reduced dataset D' to compute the sub-gradient $\nabla_t \Psi(w, [x_i, y_i])$ (according to the smooth hinge loss (6) and update w_{t+1} (using the skewed misclassification costs) as

follows:

$$w_{t+1} = w_t - \eta_t \nabla_t \Psi(w, [x_i, y_i]) = w_t - \eta_t(\lambda w_t + \nabla_t L^s(w, [x_i, y_i])) \quad (7)$$

$$\text{with } \nabla_t L^s(w, [x_i, y_i]) = \begin{cases} -\frac{1}{|D_c|} y_i x_i & y_i(w_t.x_i) \leq 0 \\ -\frac{1}{|D_c|} y_i x_i [1 - y_i(w_t.x_i)] & 0 < y_i(w_t.x_i) < 1 \\ 0 & 1 \leq y_i(w_t.x_i) \end{cases} \quad (8)$$

where $|D_c|$ is the cardinality of the class $c \in \pm 1$.

Algorithm 1: Balanced training SVM-SGD for binary classification tasks

input :
training data of the positive class D_+
training data of the negative class D_-
positive constant $\lambda > 0$
number of epochs T
output:
SVM-SGD model w

init $w_1 = 0$
for $t \leftarrow 1$ **to** T **do**
  creating the reduced dataset D' from the full set of positive class D_+ and sampling without replacement D'_- from dataset D_- (with $|D'_-| = \sqrt{|D_-| \times |D_+|}$)
  setting $\eta_t = \frac{1}{\lambda t}$
  for $i \leftarrow 1$ **to** $|D_+|$ **do**
    randomly pick a datapoint $[x_i, y_i]$ from reduced set D'
    if $(y_i(w_t.x_i) \leq 0)$ **then**
      $w_{t+1} = w_t - \eta_t(\lambda w_t - \frac{1}{|D_{y_i}|} y_i x_i)$
    else if $(y_i(w_t.x_i) < 1$ **then**
      $w_{t+1} = w_t - \eta_t\{\lambda w_t - \frac{1}{|D_{y_i}|} y_i x_i [1 - y_i(w_t.x_i)]\}$
    else
      $w_{t+1} = w_t - \eta_t \lambda w_t$
    end
  end
end
return w_{t+1}

We remark that the margin can be seen as the minimum distance between two convex hulls, H_+ of the positive class and H_- of the negative class (the farthest distance between the two classes). Under-sampling the negative class (D'_-) done by balanced training SVM-SGD provides the reduced convex hull of H_-, called H'_-. And then, the minimum distance between H_+ and H'_- is larger than H_+ and H_- (full dataset). It is easier to achieve the separating boundary than learning on the full dataset. Therefore, the training task of balanced SVM-SGD is fast to converge to the solution.

4.2 Parallel multiclass SVM-SGD training

For k classes problems, the multiclass SVM-SGD algorithm trains independently k binary classifiers. Although balanced training SVM-SGD deals with binary classification tasks with high speed, the multiclass SVM-SGD algorithm does not take the benefits of high-performance computing.

Our investigation aims at speedup training tasks of multiclass SVM-SGD with several multi-processor computers. The idea is to learn k binary classifiers in parallel.

The parallel programming is currently based on two major models, Message Passing Interface, MPI [42] and Open Multiprocessing, OpenMP [43]. MPI is a standardized and portable message-passing mechanism for distributed memory systems. MPI remains the dominant model (high performance, scalability, and portability) used in high-performance computing today. However, MPI process loads the whole dataset into memory during learning tasks, making it wasteful. The simplest development of parallel multiclass SVM-SGD algorithms is based on the shared memory multiprocessing programming model OpenMP. However, OpenMP is not guaranteed to make the most efficient computing. Finally, we present a hybrid approach that combines the benefits from both OpenMP and MPI models. The parallel learning for multiclass SVM-SGD is described in Algorithm 2. The number of MPI processes depends on the memory capacity of the HPC system used.

Algorithm 2: Hybrid MPI/OpenMP parallel multiclass SVM-SGD algorithm

input :
D the training dataset with k classes
P the number of MPI processes
output:
SVM-SGD model

Learning:
$MPI - PROC_1$
#pragma omp parallel for
for $c_1 \leftarrow 1$ **to** k_1 **do** `/* class c1 */`
  training SVM-SGD$(c_1 - vs - all)$
end
:
$MPI - PROC_P$
#pragma omp parallel for
for $c_P \leftarrow 1$ **to** k_P **do** `/* class cP */`
  training SVM-SGD$(c_P - vs - all)$
end

5 Experiments

We have implemented two parallel versions of SVM-SGD:

1. OpenMP version of balanced training SVM-SGD (Par-MC-SGD)
2. Hybrid MPI/OpenMP version of balanced training SVM-SGD (mpi-Par-MC-SGD)

Franc and Sonnenburg [44] have shown in their experiments that OCAS even in the early optimization steps shows often faster convergence than that seen so far in this domain prevailing approximative methods. Therefore, in this section we compare parallel algorithms of SVM-SGD with both OCAS and LIBLINEAR in terms of training time and classification accuracy.

OCAS. This is an optimized cutting plane algorithm for SVM [45] with the default parameter value $C = 1$.

LIBLINEAR. This is the linear SVM from [21] with default parameter value $C = 1$.

Par-MC-SGD, mpi-Par-MC-SGD. These are parallel balanced training SVM-SGD using $T = 20$ epochs and regularization term $\lambda = 0.0001$.

Our experiments are run on machine Linux 2.6.39-bpo.2-amd64, Intel(R) Xeon(R), CPU X5560, 2.8 GHz, 16 cores, and 96 GB main memory.

5.1 Dataset

The Par-MC-SGD, mpi-Par-MC-SGD algorithms are designed for large-scale datasets, so we have evaluated the performance of our approach on the three following datasets.

ImageNet 10. This dataset contains the 10 largest classes from ImageNet (24,807 images with size 2.4 GB). In each class, we sample 90 % images for training and 10 % images for testing (with random guess 10 %). First, we construct BoW of every image using dense SIFT descriptor (extracting SIFT on a dense grid of locations at a fixed scale and orientation) and 5,000 codewords. Then, we use feature mapping from [46] to get the high-dimensional image representation in 15,000 dimensions. This feature mapping has been proven to give a good image classification performance with linear classifiers [46]. We end up with 2.6 GB of training data.

ImageNet 100. This dataset contains the 100 largest classes from ImageNet (183,116 images with size 23.6 GB). In each class, we sample 50 % images for training and 50 % images for testing (with random guess 1 %). We also construct BoW of every image using dense SIFT descriptor and 5,000 codewords. For feature mapping, we use the same method as we do with ImageNet 10. The final size of training data is 8 GB.

ILSVRC 2010. This dataset contains 1,000 classes from ImageNet with 1.2M images (126 GB) for training, 50 K images (5.3 GB) for validation and 150 K images (16 GB) for testing. We use BoW feature set provided by [47] and the method reported in [48] to encode every image as a vector in 21,000 dimensions. We take $\leq$900 images per class for training dataset, so the total training images is 887,816 and the training data size is 12.5 GB. All testing samples are used to test SVM models. Note that the random guess performance of this dataset is 0.1 %.

5.2 Training time

We have only evaluated the training time of SVM classifiers excluding the time needed to load data from disk. As shown in Figs. 3 and 4, on small and medium datasets as ImageNet 10, ImageNet 100, our parallel versions show a very good speed-up in training process, compared with OCAS and LIBLINEAR (Tables 1, 2).

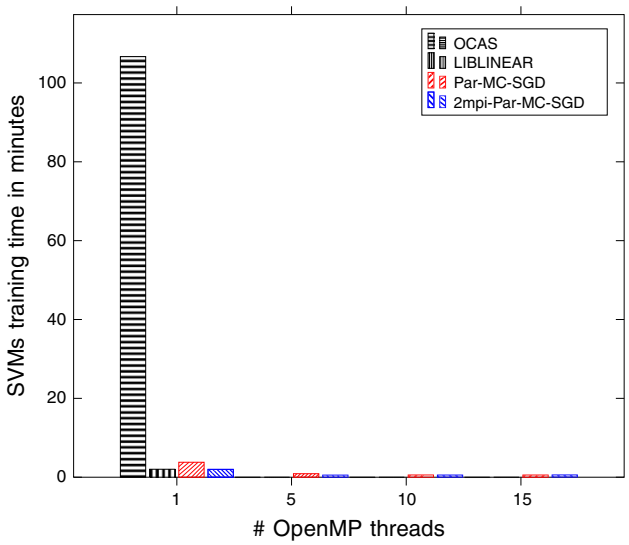

Fig. 3 SVMs training time with respect to the number of threads on ImageNet 10

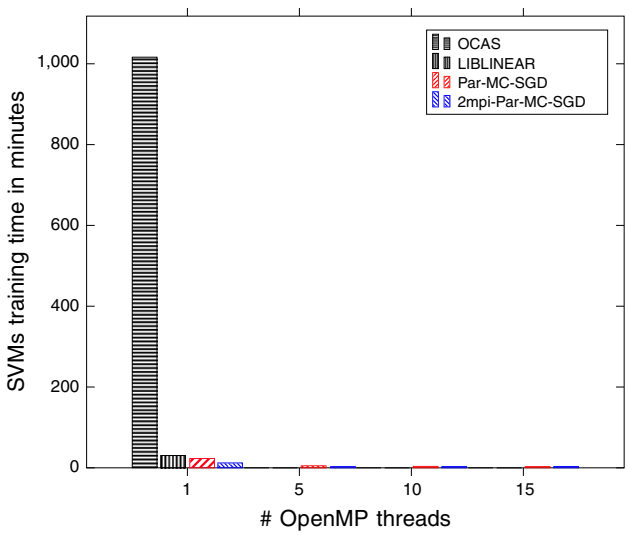

Fig. 4 SVMs training time with respect to the number of threads on ImageNet 100

Table 1 SVMs training time (minutes) on ImageNet 10

	# OpenMP threads			
Method	1	5	10	15
OCAS	106.67			
LIBLINEAR	2.02			
Par-MC-SGD	3.80	0.89	0.57	0.55
2mpi-Par-MC-SGD	1.99	0.54	0.54	**0.62**

Bold value indicates the best result

Table 2 SVMs training time (minutes) on ImageNet 100

	# OpenMP threads			
Method	1	5	10	15
OCAS	1016.35			
LIBLINEAR	30.41			
Par-MC-SGD	23.00	5.17	3.55	3.12
2mpi-Par-MC-SGD	12.15	3.11	2.78	**2.84**

Bold value indicates the best result

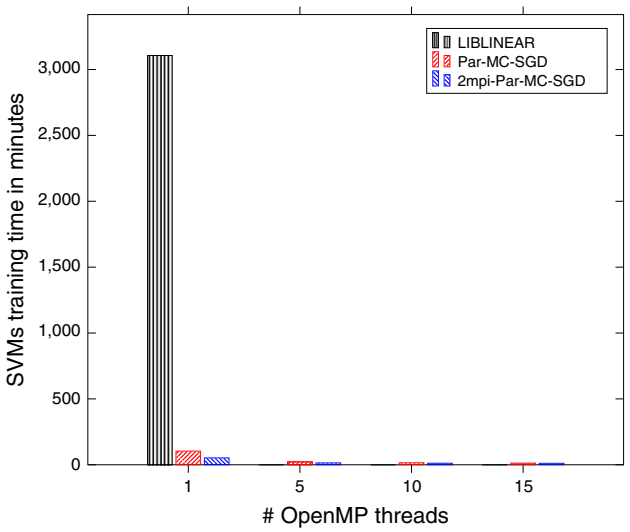

Fig. 5 SVMs training time with respect to the number of threads on ILSVRC 2010

ILSVRC 2010. Our implementations achieve a significant speed-up in training process when performing on large dataset ILSVRC 2010.

Balanced training SVM-SGD. As shown in Fig. 5, the balanced training version of SVM-SGD (Par-MC-SGD running with 1 thread) has a very fast convergence speed in training process: it is 30 times faster than LIBLINEAR (Table 3).

OpenMP balanced training SVM-SGD. On a multi-core machine, OpenMP version of balanced training SVM-SGD (Par-MC-SGD) achieves a significant speed-up in training process with 15 OpenMP threads. As shown in Fig. 5, our implementation is 249 times faster than LIBLINEAR

Table 3 SVMs training time (minutes) on ILSVRC 2010

	# OpenMP threads			
Method	1	5	10	15
OCAS	–			
LIBLINEAR	3106.48			
Par-MC-SGD	103.62	22.01	15.33	12.48
2mpi-Par-MC-SGD	51.97	14.11	11.69	**11.50**

Bold value indicates the best result

(Table 3). Due to the restriction of our computer (16 cores), we set the maximum number of OpenMP threads to 15. We can set more than 15 OpenMP threads, but according to our observation there is very few significant speed-up in training process because there is no more available core.

Hybrid MPI/OpenMP balanced training SVM-SGD. Although OpenMP balanced training SVM-SGD shows a significant speedup in training process, it does not ensure that the program achieves the most efficient high-performance computing on multi-core computer. Therefore, we explore this challenge using a combination of MPI and OpenMP models. With this approach, our implementation (mpi-Par-MC-SGD) achieves the impressive parallelization performance results on our computer. The program first loads the whole training data into computer memory and each MPI process can work with its local data independently. However, we cannot increase the number of MPI processes exceed the memory capacity of computer, because each MPI process occupies the main memory during its computation process, resulting in an increase in the overall memory requirement. OpenMP has been proven to work effectively on shared memory systems. It is used for fine-grained parallelization within each MPI process. Consequently, in each MPI process we can increase the number of OpenMP threads without demanding more extra memory. As shown in Fig. 5, our implementation achieves a significant performance in training process with 2 MPI processes and 15 OpenMP threads. It is 270 times faster than LIBLINEAR. On ILSVRC 2010, we need only 12 min to train 1000 binary classifiers, compared with LIBLINEAR (∼2 days and 4 h), as shown in Table 3. This result confirms that our approach has a great ability to scaleup to full ImageNet dataset with more than 21,000 classes.

5.3 Classification accuracy

As shown in Fig. 6, on the small datasets ImageNet 10 and medium dataset ImageNet 100, Par-MC-SGD provides very competitive performances when compared with LIBLINEAR.

We achieve a very good classification result on ILSVRC 2010. This is a large dataset with a large number of classes (1,000 classes) and a huge number of samples (more than 1

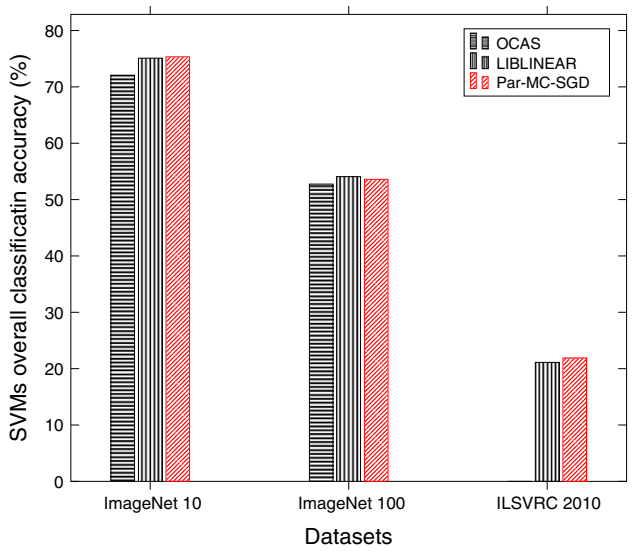

Fig. 6 Overall accuracy of SVM classifiers

Table 4 SVMs overall classification accuracy (%)

Dataset	ImageNet 10	ImageNet 100	ILSVRC 1000
OCAS	72.07	52.75	–
LIBLINEAR	75.09	54.07	21.11
Par-MC-SGD	**75.33**	53.60	**21.90**

Bold value indicates the best result

million samples), and the density of dataset is about 6 % (the proportion of non-zero elements of dataset in percent). Thus, it is very difficult for many state-of-the-art SVM solvers to obtain a high rate in classification performance. In particular, with the feature set provided by ILSVRC 2010 competition the state-of-the-art system [5] reports an accuracy of approximately 19 %, but it is far above random guess (0.1 %). And now our approach provides a significantly higher accuracy rate than [5] with the same feature set (21.90 vs. 19 %), as shown in Table 4. The relative improvement is more than 15 %. Moreover, we also compare our implementation with the current state-of-the-art of linear SVM classifiers LIBLINEAR to validate our approach. As shown in Table 4, Par-MC-SGD outperforms LIBLINEAR (+0.79 %, the relative improvement is more than 3.7 %).

Note that Par-MC-SGD runs much faster than LIBLINEAR while yielding higher rate in classification accuracy.

6 Conclusion and future work

We have developed the extended versions of SVM-SGD in several ways to efficiently deal with large-scale datasets with very large number of classes like ImageNet. The primary idea is to build the balanced classifiers with a sampling strategy and a smooth hinge loss function and then parallelize the training process of these classifiers with several multi-core computers.

Our approach has been evaluated on the 10, 100 largest classes of ImageNet and ILSVRC 2010. On ILSVRC 2010, our implementation is 270 times faster than LIBLINEAR. Therefore, we can achieve higher performances using more resources (CPU cores, computer, etc.). Furthermore, with our sampling strategy we significantly speed-up the training process of the classifiers while yielding a high performance in classification accuracy. We need only 12 min, to train 1,000 binary classifiers. Obviously, this is a roadmap towards full dataset with 21,000 classes of ImageNet. However, when the training data are larger, SVM-SGD requires a large amount of main memory due to loading the whole training data into main memory. This issue will be addressed in the next step. We may study the approach as reported in [49] and another possibility is to compress the training data and handle it on the fly [20]. In the near future we intend to provide more empirical test on full dataset with 21,000 classes of ImageNet and comparisons with parallel versions of LIBLINEAR.

References

1. Li, F.F., Fergus, R., Perona, P.: Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. Comput. Vis. Image Underst. **106**(1), 59–70 (2007)
2. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. In: Technical Report CNS-TR-2007-001, California Institute of Technology (2007)
3. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Intern. J. Comput. Vis. **88**(2), 303–338 (2010)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: Imagenet: a large-scale hierarchical image database. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 248–255. (2009)
5. Deng, J., Berg, A.C., Li, K., Li, F.F.: What does classifying more than 10, 000 image categories tell us? In: European Conference on Computer Vision. pp. 71–84. (2010)
6. Lin, Y., Lv, F., Zhu, S., Yang, M., Cour, T., Yu, K., Cao, L., Huang, T.S.: Large-scale image classification: fast feature extraction and svm training. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1689–1696. (2011)
7. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 886–893. (2005)
8. Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal estimated sub-gradient solver for svm. In: Proceedings of the Twenty-

Fourth International Conference Machine Learning, pp. 807–814. ACM (2007)
9. Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. In Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems. NIPS Foundation, (http://books.nips.cc) vol. 20, pp. 161–168. (2008)
10. Perronnin, F., Akata, Z., Harchaoui, Z., Schmid, C.: Towards good practice in large-scale learning for image classification. In: CVPR. pp. 3482–3489. (2012)
11. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: In Workshop on Statistical Learning in Computer Vision. ECCV, pp. 1–22. (2004)
12. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 2169–2178. (2006)
13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society pp. 886–893. (2005)
14. Griffin, G., Perona, D.: Learning and using taxonomies for fast visual categorization. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society (2008)
15. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: IEEE 12th International Conference on Computer Vision. IEEE, pp. 606–613. (2009)
16. Fergus, R., Weiss, Y., Torralba, A.: Semi-supervised learning in gigantic image collections. In: Advances in Neural Information Processing Systems. pp. 522–530. (2009)
17. Wang, C., Yan, S., Zhang, H.J.: Large scale natural image classification by sparsity exploration. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. IEEE, pp. 3709–3712. (2009)
18. Li, Y., Crandall, D.J., Huttenlocher, D.P.: Landmark classification in large-scale image collections. In: IEEE 12th International Conference on Computer Vision. IEEE, pp. 1957–1964. (2009)
19. Perronnin, F., Sánchez, J., Liu, Y.: Large-scale image categorization with explicit data embedding. In: CVPR. pp. 2297–2304. (2010)
20. Sánchez, J., Perronnin, F.: High-dimensional signature compression for large-scale image classification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 1665–1672. (2011)
21. Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C.: LIBLINEAR: a library for large linear classification. J. Mach. Learn. Res. **9**(4), 1871–1874 (2008)
22. Jégou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. IEEE Trans. Pattern Anal. Mach. Intell. **33**(1), 117–128 (2011)
23. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
24. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press (2000)
25. Weston, J., Watkins, C.: Support vector machines for multi-class pattern recognition. In: Proceedings of the Seventh European Symposium on Artificial Neural Networks. pp. 219–224. (1999)
26. Guermeur, Y.: Svm multiclasses, théorie et applications (2007)
27. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. J. Mach. Learn. Res. **2**, 265–292 (2001)
28. Krebel, U.: Pairwise classification and support vector machines. In: Support Vector Learning, Advances in Kernel Methods. pp. 255–268. (1999)
29. Platt, J., Cristianini, N., Shawe-Taylor, J.: Large margin dags for multiclass classification. Adv. Neural. Inf. Process. Syst. **12**, 547–553 (2000)
30. Japkowicz, N. (ed.): AAAI'Workshop on Learning from Imbalanced Data Sets. Number WS-00-05 in AAAI Tech Report (2000)
31. Weiss, G.M., Provost, F.: Learning when training data are costly: the effect of class distribution on tree induction. J. Artif. Intell. Res. **19**, 315–354 (2003)
32. Visa, S., Ralescu, A.: Issues in mining imbalanced data sets—a review paper. In: Midwest Artificial Intelligence and Cognitive Science Conf., Dayton, USA, pp. 67–73. (2005)
33. Lenca, P. anf Lallich, S., Do, T.N., Pham, N.K.: A comparison of different off-centered entropies to deal with class imbalance for decision trees. In: The Pacific-Asia Conference on Knowledge Discovery and Data Mining, LNAI 5012, pp. 634–643. Springer (2008)
34. Pham, N.K., Do, T.N., Lenca, P., Lallich, S.: Using local node information in decision trees: coupling a local decision rule with an off-centered. In: International Conference on Data Mining, Las Vegas, Nevada, USA, pp. 117–123. CSREA Press (2008)
35. Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W.: Smoteboost: improving prediction of the minority class in boosting. In: In Proceedings of the Principles of Knowledge Discovery in Databases, PKDD-2003. pp. 107–119. (2003)
36. Liu, X.Y., Wu, J., Zhou, Z.H.: Exploratory undersampling for class-imbalance learning. IEEE Trans. Syst. Man Cybern. Part B **39**(2), 539–550 (2009)
37. Ricamato, M.T., Marrocco, C., Tortorella, F.: Mcs-based balancing techniques for skewed classes: an empirical comparison. In: ICPR. pp. 1–4. (2008)
38. Shamir, O.: Making gradient descent optimal for strongly convex stochastic optimization. CoRR abs/1109.5647 (2011)
39. Ben-David, S., Loker, D., Srebro, N., Sridharan, K.: Minimizing the misclassification error rate using a surrogate convex loss. In: ICML. (2012)
40. Ouyang, H., Gray, A.G.: Stochastic smoothing for nonsmooth minimizations: accelerating sgd by exploiting structure. CoRR abs/1205.4481 (2012)
41. Rennie, J.D.M.: Derivation of the f-measure. http://people.csail.mit.edu/jrennie/writing. Accessed Feb 2004
42. MPI-Forum.: MPI: a message-passing interface standard
43. Board, OpenMP Architecture Review: OpenMP application program interface version **3** (2008)
44. Franc, V., Sonnenburg, S.: Optimized cutting plane algorithm for support vector machines. In: International Conference on Machine Learning. pp. 320–327. (2008)
45. Franc, V., Sonnenburg, S.: Optimized cutting plane algorithm for large-scale risk minimization. J. Mach. Learn. Res. **10**, 2157–2192 (2009)
46. Vedaldi, A., Zisserman, A.: Efficient additive kernels via explicit feature maps. IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 480–492 (2012)
47. Berg, A., Deng, J., Li, F.F.: Large scale visual recognition challenge 2010. In: Technical Report (2010)
48. Wu, J.: Power mean svm for large scale visual classification. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 2344–2351. (2012)
49. Do, T.N., Nguyen, V.H., Poulet, F.: Gpu-based parallel svm algorithm. J. Front. Comput. Sci. Technol. **3**(4), 368–377 (2009)