BAG-SVM-SGD FOR DEALING WITH LARGE-SCALE MULTI-CLASS DATASETS

Thanh Nhi Do\textsuperscript{1,2}, Huu Hoa Nguyen\textsuperscript{1}, The Phi Pham\textsuperscript{1}

\textsuperscript{1} College of Information Technology
Can Tho University
\textsuperscript{2} UMI UMMISCO 209 (IRD/UPMC)
dmghi@cit.ctu.edu.vn

ABSTRACT: We propose the bagging support vector machines using stochastic gradient descent (Bag-SVM-SGD) for effectively classifying large-scale multi-class datasets. The Bag-SVM-SGD learns in the parallel way from under-sampling training dataset to create ensemble binary SVM-SGD classifiers used in the One-Versus-All (OVA) multi-class strategy for performing text classification tasks with million of datapoints in thousands of classes. The numerical test results on two large scale multi-class datasets (LSHTC4, Book) show that our Bag-SVM-SGD algorithm is faster and more accurate than the state-of-the-art linear algorithm LIBLINEAR. An example of its effectiveness is given with an accuracy of 62.41\% obtained in the classification of LSHTC4 dataset having 728,067 datapoints in 1,617,900 dimensions into 2,713 classes in 104.15 minutes using a PC Intel(R) Core i7-4790 CPU, 3.6 GHz, 4 cores.

Keywords: Support Vector Machines (SVM), Stochastic gradient descent (SGD), Bagging method, large-scale multi-class.

I. INTRODUCTION

There are more and more multimedia data stored electronically, with increasing number of internet users and mobile internet access sharing videos, songs or photos on Youtube, Facebook, Twitter, Amazon. This leads to very huge amount of data, there is a need for high performance classification algorithms in order to help us find what we are looking for. The automatic classification of texts plays an important role in information management tasks, including auto-tagging emails, news, detecting or discovering topics, filtering. The emergence of new benchmarks of text classification tasks yields huge classification challenges of very high-dimensional and large-scale multi-class. For example, LSHTC4 dataset [Partalas et al. 2015] has 2.4 million documents and 325,000 classes. Our Book dataset consists of 115,000 abstracts and 661 subjects. It is very hard for any machine learning algorithm to be able to handle such datasets. This challenge motivates us to study the bagging support vector machines using stochastic gradient descent (Bag-SVM-SGD) for effectively dealing with large-scale multi-class datasets.

We propose to extend the binary SVM-SGD algorithm [Bottou and Bousquet 2008; Shalev-Shwartz et al. 2007] to develop the bagging binary SVM-SGD classifiers learnt from under-sampling training dataset. These Bag-SVM-SGD models trained in the parallel way on multi-core computers are used in the One-VersusAll multi-class strategy to perform text classification tasks with million of datapoints in thousands of classes. The numerical test results on large scale multi-class datasets (LSHTC4, Book) show that our Bag-SVM-SGD algorithm is faster and more accurate than the state-of-the-art linear algorithm, LIBLINEAR [Fan et al. 2008].

The paper is organized as follows. Section 2 briefly introduces the SVM-SGD algorithm and our proposed Bag-SVM-SGD algorithm for dealing with large-scale multi-class datasets in multi-core computers. Section 3 shows the experimental results. We then conclude in section 4.

II. BAGGING SUPPORT VECTOR MACHINES FOR LARGE-SCALE MULTI-CLASS DATASETS

A. Support vector machine for binary classification

Let us consider a binary classification problem with the dataset $D=\{X, Y\}$ consisting of $m$ datapoints $X=[x_i]_{i=1,m}$ in the $n$-dimensional input space, having corresponding labels $Y=[y_i]_{i=1,m}$ being $\pm 1$. The SVM algorithm [Vapnik 2000] tries to find the best separating plane (denoted by $w$), i.e. furthest from both class $+1$ and class $-1$. It is accomplished through the maximization of the margin (or the distance) between the supporting planes for each class. 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, its error distance denoted by $z_i=I - y_i(w,x_i) \geq 0$. The error $z_i$ is rewritten by $L(w, \{x_i, y_i\}) = \max[0, I - y_i(w,x_i)]$. And then, SVM has to simultaneously maximize the margin and minimize the error. The SVM pursues these goals with the unconstrained problem (1).

$$
\min_{w} \Psi(w, x, y) = \frac{\lambda}{2} ||w||^2 + (1/m) \sum_{i=1}^{m} \max\{0, 1 - y_i (w,x_i)\}
$$

where $\lambda$ is a positive constant used to tune the trade-off between the margin size and the error.
And then, [Bottou and Bousquet 2008; Shalev-Shwartz et al. 2007] proposed the stochastic gradient descent (SGD) method to solve the unconstrained problem (1). The SGD for SVM (denoted by SVM-SGD) updates $w$ on $T$ epochs with a learning rate $\eta$. For each epoch $t$, the SVM-SGD uses a single randomly received datapoint $(x_t, y_t)$ to compute the sub-gradient $\nabla \Psi(w, [x_t, y_t])$ and update $w_{t+1}$ as follows:

$$w_{t+1} = w_t - \eta \nabla \Psi(w, [x_t, y_t])$$

(2)

As mentioned in [Bottou and Bousquet 2008; Shalev-Shwartz et al. 2007], the SVM-SGD algorithm quickly converges to the optimal solution due to the fact that the unconstrained problem (1) is convex optimization problems 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 binary classification of 780,000 datapoints in 47,000-dimensional input space in 2 seconds on a PC and the test accuracy is similar to standard SVM, e.g. LIBLINEAR [Fan et al. 2008].

**B. Support vector machine for multi-class**

There are two strategies to extend the binary SVM solver for dealing with the multi-class problems (with the number of classes $c \geq 3$). The first one is considering the multi-class case in one optimization problem [Guermeur 2007; Weston and Watkins 1999]. The second one is decomposing multi-class into a series of binary SVMs, including One-Versus-All [Vapnik 2000], One-Versus-One [Krebel 1999]. In practice, the most popular methods are One-Versus-All (ref. LIBLINEAR [Fan et al. 2008]), One-Versus-One (ref. LibSVM [Chang and Lin 2011]) and are due to their simplicity. The One-Versus-All strategy (as illustrated in Figure 1) builds $c$ different binary SVM models where the $i^{th}$ one separates the $i^{th}$ class from the rest. The One-Versus-One strategy (as illustrated in Figure 2) constructs $c(c-1)/2$ binary SVM models for all the binary pairwise combinations of the $c$ classes. The class is then predicted with the largest distance vote.

![Figure 1. Multi-class SVM (One-Versus-All)](image1)

![Figure 2. Multi-class SVM (One-Versus-One)](image2)

**C. Bagging svm-sgd for large-scale multi-class**

When dealing with very large number of classes, e.g. $c = 1,000$ classes, the One-Versus-One strategy is too expensive because it needs training $499,500$ of binary classifiers and using them in the classification (compared to $1,000$ binary models learned by the One-Versus-All strategy). Therefore, the One-Versus-All strategy is suited for handling this case. And then, we propose to use the One-Versus-All approach to train $c$ binary SVM-SGD classifiers. However, the multi-class 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 algorithm also takes very long time to train very large number of binary classifiers in sequential mode using a single processor.

Due to these problems, we propose the parallel bagging SVM-SGD algorithm (denoted by Bag-SVM-SGD) being able to efficiently handle the large number of datapoints in large-scale multi-class on standard personal computers (PCs). The first one is to build ensemble binary classifiers with under-sampling strategy. The second one is to parallelize the training task of all binary classifiers with multi-core machines.

**Bagging binary SVM-SGD classifier**

In the multi-class SVM-SGD algorithm using One-Versus-All approach, the learning task of binary SVM-SGD classifier is try to separate the class $c_i$ (positive class) from the $c-1$ other classes (negative class). For very large number of classes, this leads to extreme unbalance between the positive and the negative class. The problem of binary SVM-SGD comes from randomly sampling the single datapoint $(x_i, y_i)$ to compute the sub-gradient $\nabla P(w, \{x_i, y_i\})$ and update $w_{+i}$. Given a classification problem with 1,000 classes, 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 binary SVM-SGD classifier focuses mostly on the negative datapoints. Therefore, the binary SVM-SGD classifier has difficulty to separate the positive class from the negative class, well-known as the class imbalance problems. One of the most popular solutions for dealing with the imbalanced data [Japkowicz 2000; Visa and Ralescu 2005; Weiss and Provost 2003] is to change the data distribution, including over-sampling the minority class [Chawla et al. 2003] or under-sampling the majority class [Liu et al. 2009; Ricamato et al. 2008]. Nevertheless, over-sampling the minority class is very expensive due to large datasets with millions datapoints. Given the training dataset $D$ consists of the positive class $D^+$ ($|D^+|$ is the cardinality of the positive class $c_i$) and the negative class $D^-$ ($|D^-|$ is the cardinality of the negative class). Our bagging binary SVM-SGD trains $\kappa$ SVM-SGD classifiers $\{w_1, w_2, \ldots, w_\kappa\}$ from mini-batch using undersampling the majority class (negative class) to separate the positive class $c_i$ from the negative class. Since the original bagging [Breiman 1996] uses bootstrap sampling from the training dataset without regard to the class distribution. Our bagging SVM-SGD follows the idea of bagging in more appropriate strategy for dealing with class imbalance. At the $i^{th}$ iteration, the mini-batch $mB_i$ includes $n_p$ datapoints randomly sampling without replacement from the positive class $D^+$ and $n_p \sqrt{|D^-|^D^+}$ datapoints sampling without replacement from the negative class $D^-$, and then the learning algorithm learns $w_t$ from $mB_i$. Such a mini-batch improves the chance for a positive datapoint sampled. The Bag-SVM-SGD averages all classifiers $\{w_1, w_2, \ldots, w_\kappa\}$ to create the final model $w$ for separating the class $c_i$ from other ones. The binary Bag-SVM-SGD algorithm is described in algorithm 1.

![Algorithm 1](image-url)

**Parallel training of Bag-SVM-SGD**

The Bag-SVM-SGD algorithm independently trains $c$ binary classifiers for $c$ classes in multi-class SVM-SGD. This is a nice property for parallel learning. The main idea is to learn $c$ binary classifiers in the parallel way to speed-up
training tasks for large-scale multi-class datasets. The simplest development of parallel Bag-SVM-SGD described in algorithm 2 is based on the shared memory multiprocessing programming model OpenMP on multi-core computers.

```plaintext
input:
training dataset $D$ with $c$ classes
positive constant $\lambda > 0$
number of epochs $T$
number of SVM-SGD models $\kappa$

output:
hyperplanes \{w_1, w_2, \ldots, w_c\}

begin
#pragma omp parallel for schedule(dynamic)
for $c_i \leftarrow 1$ to $c$ do  /* class $c_i$ */
    $w_{c_i} = \text{Bag-SVM-SGD}(c_i, D, \lambda, T, \kappa)$
end
```

**Algorithm 2.** Parallel training ensemble binary Bag-SVM-SGD classifiers in One-Versus-All of large-scale multi-class SVM

### III. NUMERICAL TEST RESULTS

In order to evaluate the performance (accuracy and training time) of the Bag-SVM-SGD algorithm for classifying a large amount of data in large-scale multi-class, we have implemented the Bag-SVM-SGD in C/C++, OpenMP [OpenMP Architecture Review Board 2008]. We are interested in the best state-of-the-art linear SVM algorithm, LIBLINEAR (a library for large linear classification [Fan et al. 2008], the parallel version on multi-core computers). Therefore, we here report the comparison of the classification performance obtained by the Bag-SVM-SGD and the LIBLINEAR.

All experiments are run on a PC with Linux Fedora 20, Intel(R) Core i7-4790 CPU, 3.6 GHz, 4 cores and 32 GB main memory.

#### A. Description of datasets

The Bag-SVM-SGD algorithm is designed for dealing with a large amount number of data in large-scale multi-class, so we have evaluated its performance on the two following datasets.

**Book collection**

Book dataset is the real book collection at the Learning Resource Center of Can Tho University in Vietnam. It consists of 114,998 books in a quadruplet format: \langle Title; Abstract; Keywords; Subject \rangle. Due to the book information in Vietnamese and in English, there are not only one-syllable words but also multiple syllables ones. We use JvnTextPro [Nguyen et al. 2010] well-known as a good Vietnamese word segmentation, to perform the word splitting. The dictionary has 89,821 vocabulary words. The representation of books in the BoW model brings out the table with 114,998 rows, 89,821 columns. Furthermore, there are 661 subjects. Therefore, it yields huge classification challenges of very-high-dimensional and large-scale multi-class dataset. The dataset is randomly divided into training set with 100,000 rows and testing set with 14,998 rows (with random guess about 0.1513%).

**LSHTC4 dataset**

The LSHTC4 dataset [Partalas et al. 2015] is a benchmark for large-scale text classification. The dataset originates from the DBpedia site. The LSHTC4 is represented in the BoW model. The dictionary has 1,617,900 vocabulary words. The categories in the DBpedia site have the complex relationships. Since the LSHTC4 is multi-label benchmark, a text (datapoint) may belong to more than one category with respect to the hierarchy of categories. We convert the multi-label LSHTC4 to the multi-class one. The top category is assigned for a text. Furthermore, the dataset only includes the categories having more than 100 individuals. We obtain the new multi-class LSHTC4 dataset with 728,067 datapoints in 1,617,900 dimensions into 2,713 classes. And then, the dataset is randomly divided into training set with 628,067 rows and testing set with 100,000 rows (with random guess about 0.0369%).
B. Tuning parameters

With such large datasets in very high-dimensional, linear SVM models have given competitive performances compared to non-linear classifiers but training and testing are much faster [Doan et al. 2013; Yuan et al. 2012]. That is why we propose to use linear SVM models for classifying these large datasets in the experiments.

The training set is used to build the classification model and tune the parameters. Then, the classification results are reported on the testing set using the resulting models.

The positive constant \( C=100,000 \) (a trade-off between the margin size and the errors in learning SVM algorithms, the same tuning in [Do 2014; Do and Tran Nguyen 2016; Doan et al. 2015]) was used in LIBLINEAR.

Our Bag-SVM-SGD algorithm learns \( \kappa=50 \) binary SVM-SGD classifiers and regularization term \( \lambda=0.000002 \) to separate one class from other ones.

Due to the PC (Intel(R) Core i7-4790 CPU, 4 cores) used in the experimental setup, we try to vary the number of OpenMP threads (1, 4 threads) for all training tasks.

C. Classification results

Firstly, we would like to compare the training time of the Bag-SVM-SGD to the LIBLINEAR on two large-scale multi-class datasets as described above.

Training time

For Book collection having medium number of texts (100,000 texts) and large-scale multi-class (661 classes), the training time of the LIBLINEAR using 4 OpenMP threads is 5.57 minutes, compared to 0.46 minutes of the Bag-SVM-SGD. The Bag-SVM-SGD is 12.11 times faster than the LIBLINEAR.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># OpenMP threads</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIBLINEAR</td>
<td>12.34 5.57</td>
</tr>
<tr>
<td>Bag-SVM-SGD</td>
<td>1.47 0.46</td>
</tr>
</tbody>
</table>

Table 1. Training time (minutes) on Book collection

![Figure 3. Comparison of training time (minutes) on Book collection](image)

LSHTC4 is the large dataset with 600,000 texts in 1,617,900 dimensions into 2,713 classes. The LIBLINEAR can not deal with LSHTC4 due to the Segmentation fault (core dumped) error. The Bag-SVM-SGD has finished the training task in 381.59 minutes (using 1 OpenMP thread) and 104.15 minutes (using 4 OpenMP threads).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># OpenMP threads</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIBLINEAR</td>
<td>N/A</td>
</tr>
<tr>
<td>Bag-SVM-SGD</td>
<td>381.59 104.15</td>
</tr>
</tbody>
</table>

Table 2. Training time (minutes) on LSHTC4
Training time reported in Tables 1, 2 show that our Bag-SVM-SGD can reduce learning time linearly with the number of cores used in training tasks.

**Classification accuracy**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Dataset</th>
<th>Book</th>
<th>LSHTC4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIBLINEAR</td>
<td>64.61</td>
<td>N/A</td>
<td>62.41</td>
</tr>
<tr>
<td>Bag-SVM-SGD</td>
<td>64.69</td>
<td>62.41</td>
<td></td>
</tr>
</tbody>
</table>

*Table 3. Overall classification accuracy (%)*
Table 3 and Figure 5 present the classification results in terms of accuracy. The Bag-SVM-SGD achieves very competitive correctness compared to the LIBLINEAR on Book collection dataset. However, only Bag-SVM-SGD has finished the training task on LSHTC4 dataset with an accuracy of 62.28%.

These results show that our Bag-SVM-SGD has a great ability to scale-up to a large amount of datapoints in very high-dimensional input space and large-scale multi-class.

IV. CONCLUSION AND FUTURE WORKS

We have presented the parallel bagging support vector machines using stochastic gradient descent (Bag-SVM-SGD) on multi-core computers that achieves high performances for dealing with a large amount of data in large-scale multi-class. The main idea of the Bag-SVM-SGD is to learn in a parallel way from under-sampling training dataset to create ensemble binary SVM-SGD classifiers used in the One-Versus-All (OVA) multi-class strategy for performing classification tasks with millions of datapoints in millions of dimensions and thousands of classes. The numerical test results on two large scale multi-class datasets (LSHTC4, Book) show that our Bag-SVM-SGD algorithm is faster and more accurate than the state-of-the-art linear algorithm LIBLINEAR.

In the near future, we intend to develop a distributed implementation for large scale processing on an in-memory cluster-computing platform, Apache Spark [Zaharia et al. 2010] (running times or up to 100x faster than Hadoop MapReduce, or 10x faster on disk).

REFERENCES

THUẬT TOÁN BAG-SVM-SGD CHO XỬ LÝ DỮ LIỆU ĐA LỚP LỚN

Thành Nghị Đỗ, Hữu Hòa Nguyễn, Thế Phi Phạm

TÓM TẮT: Chúng tôi đề xuất thuật toán Bagging máy học vector hỗ trợ sử dụng phương pháp giảm gradient ngẫu nhiên (Bag-SVM-SGD) cho phân lớp hiểu quả tập dữ liệu đa lớp lớn. Giải thuật Bag-SVM-SGD thực hiện huấn luyện song song từ tập huấn luyện tạo thành từ việc lấy mẫu giảm tập dữ liệu gốc, để tạo ra tập hợp các bộ phân lớp nhị phân (SGD) được sử dụng trong chiến lược phân lớp đa lớp một-tắt cã, cho vấn đề phân lớp hàng trăm văn bản trong hàng nghìn câu của gổ LSHTC4 (Book) cho thấy rằng giải thuật Bag-SVM-SGD của chúng tôi đạt hiệu suất huấn luyện và chính xác khi so sánh với giải thuật huấn luyện LIBLINEAR được cho là hiệu quả nhất hiện nay. Một ví dụ về tính hiệu quả của giải thuật Bag-SVM-SGD là nó có thể hoàn thành huấn luyện mô hình phân lớp với độ chính xác 62,41% cho tập dữ liệu LSHTC4 có 720067 phân tử trong 1617900 câu và 2713 lớp trong 104,15 phút trên PC Intel(R) Core i7-4790 CPU, 3,6 GHz, 4 nhân.

Từ khóa: Máy học vector hỗ trợ (SVM), Phương pháp giảm gradient ngẫu nhiên (SGD), Bagging, dữ liệu đa lớp lớn.