Large-Scale Patent Classification with Min-Max Modular Support Vector Machines

Xiao-Lei Chu, Chao Ma, Jing Li, Bao-Liang Lu∗ Senior Member, IEEE, Masao Utiyama, and Hitoshi Isahara

Abstract—Patent classification is a large-scale, hierarchical, imbalanced, multi-label problem. The number of samples in a real-world patent classification typically exceeds one million, and this number increases every year. An effective patent classifier must be able to deal with this situation. This paper discusses the use of min-max modular support vector machine (M3-SVM) to deal with large-scale patent classification problems. The method includes three steps: decomposing a large-scale and imbalanced patent classification problem into a group of relatively smaller and more balanced two-class subproblems which are independent of each other, learning these subproblems using support vector machines (SVMs) in parallel, and combining all of the trained SVMs according to the minimization and the maximization rules. M3-SVM has two attractive features which are urgently needed to deal with large-scale patent classification problems. First, it can be realized in a massively parallel form. Second, it can be built up incrementally. Results from experiments using the NTCIR-5 patent data set, which contains more than two million patents, have confirmed these two attractive features, and demonstrate that M3-SVM outperforms conventional SVMs in terms of both training time and generalization performance.

I. INTRODUCTION

CURRENT patent classification mainly relies on human experts. The whole process is inefficient and imprecise. Automatic classification systems based on machine learning techniques can greatly reduce the workload, and human experts could provide a further breakdown, if needed, of automatic classification. Moreover, patent classification is a fundamental of patent analysis. Through the analysis of competitors’ patents, valuable information pertaining to market strategy, product development direction, and so on can be obtained. Related patents can also be mined for specific information such as the identity of technical leaders and the key technologies in a field.

Because of its great importance, automatic patent classification has received much attention [1], [2], [3], [4], [5], [6]. The European Patent Office (EPO) tried a variety of preprocessing methods on patent data, such as assigning different weights to patent sections and, utilizing the co-citation between patents. They also redefined the evaluation standards for patent classification, and pointed out that precision levels of the order of 80% are required for practical usage [1]. Larkey [2] [3] designed a system for searching and classifying U.S. patent documents based on query, and used a K-nearest-neighbor algorithm in this system because of its scalability to larger data sets. The Japan Patent Office [4] developed a patent classification system based on keywords, and used about 310,000 patents in the training procedure. They achieved 96% classification accuracy in 38 categories and 82.5% classification accuracy in 2,815 subcategories. Their experiments also indicated that in order to ensure higher classification accuracy, each subcategory should contain at least 1,000 training samples. Fall et. al. compared the most commonly used classifiers, such as naive Bayes, k-NN, support vector machines (SVM), neural networks, and decision rules, and reported that SVM provides the best performance [5], [6].

The above research mainly dealt with small data sets, but real-world patent classification problems may contain more than a million samples. The time and space complexities of SVM limit its application in such large-scale patent classification.

In this paper, we use min-max modular support vector machines (M3-SVM) [7], [8], [9], [10] to tackle large-scale patent classification problems. M3-SVM includes three steps: 1) a large-scale pattern classification problem is divided into a number of smaller and more balanced subproblems according to decomposition strategies; 2) all of the subproblems are solved by support vector machines in a massively parallel way; 3) all of the trained support vector machines are combined according to minimization and maximization rules [11]. To speed up training and maintain the generalization performance of M3-SVM for patent classification, a prior-knowledge-based task decomposition strategy is used. We also analyze the incremental learning ability of M3-SVM. Results from experiments using the NTCIR-5 database [12] demonstrate that M3-SVM could surpass traditional SVM in terms of both classification accuracy and training time. We have also found that if a newly introduced training data set is incrementally learned, the training time can be greatly shortened.

The rest of the paper is organized as follows. Section II provides background knowledge regarding patents and their classification. In Section III, the min-max modular network model is briefly introduced, and the incremental learning ability of M3-SVM is analyzed. In Section IV, the experiments using a large-scale patent data set are described and experimental results are presented in detail. The conclusion and future work are presented in Section V.
II. PATENT CLASSIFICATION BACKGROUND

The International Patent Classification (IPC) provides a common classification for patents and inventions, including published patent applications, utility models and certificates. The IPC is a hierarchically structured system including SECTION, CLASS, SUBCLASS, and GROUP layers as illustrated in Fig. 1. The SECTION layer is the top layer, and it has eight categories denoted by letters from A to H. The CLASS layer is under the SECTION layer and has 120 categories. The CLASS layer is expressed as a SECTION label followed by two digits; for example, ‘A01’. The third layer is the SUBCLASS layer, which is represented as a CLASS label followed by a capital letter; for example, ‘A01B’. The number of categories in the SUBCLASS layer is 630. SUBCLASS can be further divided into GROUP, but in general current research has mainly concentrated on the top three layers since the definitions of layers below SUBCLASS are still frequently changed.

Patent documents are generally stored in XML format as shown in Fig. 2. A patent is composed of three main sections: Abstract, Claim, and Description, and other descriptive information such as Title and IPC.

III. MIN-MAX MODULAR SUPPORT VECTOR MACHINE

Before using $M^3$-SVM, we should divide a $K$-class problem into $K(K-1)/2$ two-class subproblems according to a one-against-one strategy. The work procedure of $M^3$-SVM consists of three steps: task decomposition, SVM training, and module combination. Figure 3 shows this idea of fine decomposition and module combination for a two-class problem. $\text{Min}^{+\ast}$ expresses the MIN unit for the $i$th subset of the positive class.

A. Task Decomposition

Let $T_{ij}$ be the given training data set for a two-class classification problem,

$$T_{ij} = \{(X_i^{(i)}, +1)\}_{i=1}^{L_i} \cup \{(X_j^{(j)}, -1)\}_{j=1}^{L_j}$$

for $i = 1, \ldots, K$ and $j = i + 1, \ldots, K$ (1)

where $X_i^{(i)} \in X_i$ and $X_i^{(j)} \in X_j$ are the training inputs belonging to class $C_i$ and class $C_j$, respectively; $X_i$ is the set of training inputs belonging to class $C_i$; $L_i$ denotes the number of data in $X_i$; $\bigcup_{i=1}^{K} X_i = X$; and $\sum_{i=1}^{K} L_i = L$. In this paper, the training data in a two-class subproblem are called positive training data if their desired outputs are +1. Otherwise, they are called negative training data.

Although the two-class subproblems defined by Eq. (1) are smaller than the original $K$-class problem, this partition may not be adequate for parallel learning. To speed up training, all the large and imbalanced two-class subproblems should be further divided into smaller and more balanced two-class subproblems.

Assume that $X_i$ is partitioned into $N_i$ subsets in the form

$$X_i = \{(X_i^{(i)})\}_{i=1}^{N_i}$$

for $j = 1, \ldots, N_i$ and $i = 1, \ldots, K$, (2)
where $1 \leq N_i \leq L_i$ and $\cup_{j=1}^{N_j} X_{ij} = X_i$.

After partitioning $X_i$ into $N_i$ subsets, every two-class subproblem $T_{ij}$ defined by Eq. (1) can be further divided into $N_i \times N_j$ smaller and more balanced two-class subproblems as follows:

$$T_{ij}^{(u,v)} = \{(X_{iu}^{(v)}, +1)\}_{i=1}^{L_i^{(v)}} \cup \{(X_{jv}^{(v)}, -1)\}_{j=1}^{L_j^{(v)}}$$

for $u = 1, \ldots, N_i$, $v = 1, \ldots, N_j$,

$$i = 1, \ldots, K, \text{ and } j = i + 1, \ldots, K$$

where $X_{iu}^{(v)} \in X_{iu}$ and $X_{jv}^{(v)} \in X_{jv}$ are the training inputs belonging to class $C_i$ and class $C_j$, respectively; $\sum_{u=1}^{N_i} L_i^{(v)} = L_i$; and $\sum_{v=1}^{N_j} L_j^{(v)} = L_j$.

After task decomposition, each of the two-class subproblems can be treated as a completely independent, non-communicating problem in the learning phase. Therefore, all the two-class subproblems defined by Eq. (3) can be efficiently learned in a massively parallel way.

From Eqs. (1) and (3), we see that a $K$-class problem is divided into

$$\sum_{i=1}^{K-1} \sum_{j=i+1}^{K} N_i \times N_j$$

two-class subproblems. The number of training data for each of the two-class subproblems is about

$$[L_i/N_i] + [L_j/N_j]$$

Since $[L_i/N_i] + [L_j/N_j]$ is independent of the number of classes $K$, the size of each of the two-class subproblems is much smaller than the original $K$-class problem for reasonable values of $N_i$ and $N_j$.

### B. Module Combination

After training, all the individual SVMs are integrated into a $M^3$-SVM with MIN and MAX units according to two combination principles: the minimization principle and the maximization principle [11].

#### Minimization Principle:

Suppose a two-class problem $B$ is divided into $P$ smaller two-class subproblems, $B_i$ for $i = 1, \ldots, P$, and also suppose that all the two-class subproblems have the same positive training data and different negative training data. If the $P$ two-class subproblems are correctly learned by the corresponding $P$ individual SVMs, $M_i$ for $i = 1, \ldots, P$, then the combination of the $P$ trained SVMs with a MIN unit will produce the correct output for all the training inputs in $B$, where the function of the MIN unit is to find a minimum value from its multiple inputs. The transfer function of the MIN unit is given by

$$q(x) = \min_{i=1}^{P} M_i(x)$$

where $x$ denotes the input variable.

#### Maximization Principle:

Suppose a two-class problem $B$ is divided into $P$ smaller two-class subproblems, $B_i$ for $i = 1, \ldots, P$, and also suppose that all the two-class subproblems have the same negative training data and different positive training data. If the $P$ two-class subproblems are correctly learned by the corresponding $P$ individual SVMs, $M_i$ for $i = 1, \ldots, P$, then the combination of the $P$ trained SVMs with a MAX unit will produce the correct output for all the training input in $B$, where the function of the MAX unit is to find a maximum value from its multiple inputs. The transfer function of the MAX unit is given by

$$q(x) = \max_{i=1}^{P} M_i(x)$$

### C. Incremental Learning

Patent classification is a typical incremental learning problem, because new patents are issued continuously. Incremental learning could be used to learn the novel knowledge from in-coming patents, and models which have been trained can be reused. $M^3$-SVM has such incremental learning ability if a decomposition strategy, such as prior-knowledge-based decomposition, can be used to divide each new data set independently of the learned datasets whereas random decomposition can-not be accommodated within the incremental learning architecture. Figure 4 shows $M^3$-SVM models after addition of a new data set. Previously trained models are reused in $M^3$-SVM.

First, the number of training models in the original data set can be easily calculated using

$$M^{(1)} = \sum_{0<i<j\leq K} \left( \left\lceil \frac{L_i}{a_i} \right\rceil \times \left\lceil \frac{L_j}{a_j} \right\rceil \right)$$

where $L_i$ is the sample number in the $i$th category, and $a_i$ is the sample number in each subset of the $i$th category.
where $L_i'$ is the sample number in the $i$th category in the newly introduced data set, and $a_i$ is the sample number in each subset of the $i$th category in the newly introduced data set. Assuming the newly introduced data set is equal in sizes to the original data set, we let all $a_i$ and $a_i'$ be equal to $a$ and all $L_i$ and $L_i'$ be equal to $L$. Equations (8) and (9) then become

$$M^{(1)} = \frac{K(K - 1)}{2} \frac{L_1}{a^2}$$

$$M^{(2)} = \frac{4K(K - 1)}{2} \frac{L_i}{a^2}$$

The number of models which should be incrementally learned is

$$M^{(2)} - M^{(1)} = \frac{3K(K - 1)}{2} \frac{L_1}{a^2}$$

After we introduce another data set (equal in size to the first data set),

$$M^{(3)} = \frac{9K(K - 1)}{2} \frac{L_i}{a^2}$$

The number of models which should be incrementally learned becomes

$$M^{(3)} - M^{(2)} = \frac{5K(K - 1)}{2} \frac{L_i}{a^2}$$

Thus, we can infer that after the introduction of the $i$th new data set (equal in size to the first data set),

$$M^{(i)} = i^2 W$$

where $W = \frac{K(K - 1)}{2} \frac{L_i}{a^2}$ is a constant. The number of models which should be incrementally learned is

$$M^{(i)} - M^{(i-1)} = i^2 W - (i - 1)^2 W = (2i - 1)W$$

Equations (15) and (16) show that when incremental learning is used, the number of models which should be incrementally learned decreases from $O(i^2)$ to $O(i)$.

This theoretical analysis has been verified by our experimental results, which are given in Section IV.

IV. PATENT CLASSIFICATION EXPERIMENTS

A. Data Set

The data set used in our experiment was collected from the NTCIR-5 patent data set [12] which follows the IPC taxonomy. There are about 350,000 new patents in per year, and new patents from a 7-year period were used in our experiment. The total number of patents is 2,399,884. We use the patents from the first five years’ as training data, and those from the final two years’ as test data. The number of patents in each year and in each class are listed in Table I.

We used the hierarchical text classification model to solve the patent classification problem, and focused on using M3-SVMs to solve the large-scale problem in the SECTION layer. The SECTION layer has eight categories from A to H, and the distribution of patents in eight categories is listed in Table I. Note that a patent has one main category label and may also have several compensatory labels. Here, we simplify the multi-label problem into a unique label problem by only considering the main label of the patent.

B. Feature Selection Method

First, the raw Japanese patent was segmented using chasen [13], a Japanese morphological analyzer, and all auxiliary words and punctuation were removed. A patent consists mainly of three sections - Abstract, Claim, and Description. In our experiment, the three sections were weighted equally and wholly indexed into a single vector using the TFIDF method. Other researchers have proved that $\chi_{avg}$, $\chi_{max}$ and Information Gain [14] are the top feature selection methods for text classification. We compared these three methods in our experiment. The results showed that the three methods are approximately equal in performance. We
also tuned the dimension parameter from 2,500 to 160,000, and found that 5,000 was the smallest value that enabled close to the top performance. Thus, we used the $\chi_{avg}$ method and a dimension size of 5000 in the experiments discussed in this paper.

C. Classifier

The M$^3$ framework can use support vector machine, K nearest neighbor, or neural network classifiers as the base classifier. In the experiments of other researchers, the support vector machine has proved to be the optimal classifier for most text classification problems, so we used SVM$^{light}$ [15], [16] in our experiment as the baseline algorithm and base classifier of M$^3$. SVM$^{light}$ is designed to solve large-scale text classification problems. It optimizes the speed of both the training and the testing, while also ensuring classification accuracy. Because of the outstanding capability of SVM$^{light}$, we considered it the most appropriate classifier for our experiment. We also used the linear kernel for SVM, because the linear kernel enables the shortest training and test times. We attempted to use other types of kernel, but the training seemed endless in the same experimental environment.

D. Decomposition Strategies

Effective task decomposition is vital for M$^3$-SVMs to achieve satisfactory performance [7]. We use three decomposition strategies in our experiments.

1) Random task decomposition is quite straightforward. After the subset sizes are decided, patents are randomly selected to form subsets. In our experiment, we set the subset size to 2000 based on our experience.

2) Year&Random task decomposition first divides patents into subsets according to year, and then each year’s patents are further divided into subsets using the Random Algorithm according to the predefined subset size(2000 in our experiments).

3) Year&CLASS task decomposition first divides patents into subsets according to year, and then each year’s patents are further divided into subsets according to taxonomy. The CLASS layer is used in our experiments. The CLASS number in each SECTION is shown in Table II.

Among the three decomposition strategies, Year&CLASS decomposition provided the best performance, as shown in Fig. 5. In the following discussion, we therefore only consider M$^3$-SVM based on Year&CLASS decomposition.

E. Baseline Comparison

As noted, SVM$^{light}$ performed well for most of the text classification problems, including our patent data set, in terms of both speed and classification accuracy. We compared M$^3$-SVM based on Year&CLASS decomposition with SVM$^{light}$, and found that our method could surpass even SVM$^{light}$ in the following four regards:

1) Classification Accuracy: We use the 1998-1999 patents as a test set, and the 1997, 1996-1997, 1995-1997, 1994-1997, and 1993-1997 sets in turn as the training set. Figure 5 shows the classification results with M$^3$-SVM and SVM$^{light}$. As the training data number increased, the performance of M$^3$-SVM surpassed that of SVM$^{light}$. In addition, the performance of SVM$^{light}$ dropped sharply with the introduction of distant years’ patents, which would complicate the decision boundary. M$^3$-SVM can simplify the decision boundary by dividing the original complex problem into a group of much simpler subproblems.

2) Parameters Independence: The support vector machine algorithm can achieve advanced performance only when the training parameters are properly set. For large-scale classification problems, it could require several days or more to finish one round of training, so parameter tuning is unrealistic for this kind of problem. Although we can select part of the training data to use for parameter tuning, whether this subset will accurately represent the whole data set is not guaranteed.

We used the SVM$^{light}$ classifier, the performance of SVM$^{light}$ changed dramatically along with the parameter tuning. In contrast, M$^3$-SVM could provide satisfactory performance regardless of parameter changes.

### Table I: Number of patents per year in eight sections

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>30,583</td>
<td>31,316</td>
<td>28,357</td>
<td>25,444</td>
<td>22,475</td>
<td>32,427</td>
<td>33,126</td>
<td>203,728</td>
</tr>
<tr>
<td>B</td>
<td>65,338</td>
<td>68,474</td>
<td>68,130</td>
<td>68,278</td>
<td>62,436</td>
<td>68,148</td>
<td>69,648</td>
<td>470,652</td>
</tr>
<tr>
<td>C</td>
<td>30,147</td>
<td>31,834</td>
<td>34,163</td>
<td>37,996</td>
<td>35,700</td>
<td>31,198</td>
<td>31,494</td>
<td>233,132</td>
</tr>
<tr>
<td>D</td>
<td>4,904</td>
<td>5,228</td>
<td>5,794</td>
<td>6,127</td>
<td>5,604</td>
<td>4,642</td>
<td>4,968</td>
<td>37,267</td>
</tr>
<tr>
<td>E</td>
<td>18,604</td>
<td>18,000</td>
<td>16,114</td>
<td>13,690</td>
<td>11,099</td>
<td>18,604</td>
<td>18,810</td>
<td>114,922</td>
</tr>
<tr>
<td>F</td>
<td>30,296</td>
<td>31,188</td>
<td>29,358</td>
<td>28,258</td>
<td>26,871</td>
<td>31,403</td>
<td>32,938</td>
<td>210,112</td>
</tr>
<tr>
<td>G</td>
<td>70,692</td>
<td>81,691</td>
<td>81,677</td>
<td>88,716</td>
<td>95,679</td>
<td>79,158</td>
<td>83,942</td>
<td>588,555</td>
</tr>
<tr>
<td>H</td>
<td>72,589</td>
<td>72,164</td>
<td>72,544</td>
<td>81,486</td>
<td>86,834</td>
<td>75,305</td>
<td>80,594</td>
<td>541,516</td>
</tr>
<tr>
<td>TOTAL</td>
<td>330,954</td>
<td>339,895</td>
<td>336,137</td>
<td>349,995</td>
<td>346,498</td>
<td>340,885</td>
<td>355,520</td>
<td>2,399,884</td>
</tr>
</tbody>
</table>

### Table II: Class number in each IPC section

<table>
<thead>
<tr>
<th>SECTION</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLASS number</td>
<td>15</td>
<td>36</td>
<td>19</td>
<td>8</td>
<td>7</td>
<td>17</td>
<td>13</td>
<td>5</td>
</tr>
</tbody>
</table>
This is shown in Fig. 6, and calculated mean and standard deviation values are shown in Table III. In general, M³-SVM provided the best performance without parameter selection, and this capability is important for large-scale problems.

3) Time Cost: We also compared the training and test times between M³-SVM and SVM\textsuperscript{light}. SVM\textsuperscript{light} provides greatly improved training and test times compared with those of traditional SVM classifiers. Our experiments were done on a Lenovo cluster composed of three fat nodes and thirty thin nodes. Each fat node had 32G RAM and two 3.2GHz Intel(R) Xeon(TM) CPUs, while each thin node had 8G RAM and two 2.0GHz Intel(R) Xeon(TM) CPUs with each CPU having four cores. The SVM\textsuperscript{light} experiments were done on the fat nodes, and the M³-SVM experiments were done on the thin nodes. The results are shown in Fig. 7. We found that although running on slower CPUs, M³-SVM can greatly reduce the training time. However, M³-SVM needed more testing time than SVM\textsuperscript{light}. For a fixed subproblem size, the M³-SVM model number will grow with the training set size, so the test time will also increase. In the SVM\textsuperscript{light} test process with a linear kernel, though, only the distance between the test vectors and the dividing hyperplane needs to be calculated, which is independent of the training set size. Although the test time was greater, M³-SVM needed only 2ms to test a patent, and can be used in real-time. Moreover, we used just 240 CPU cores to do the computation in parallel in the M³-SVM experiment, but all the training of subproblems could be done in parallel. Using the Year&CLASS decomposition method, the original five years’ of patents were divided into almost 150,000 subsets according to CLASS as shown in Table II. With enough computation nodes, the M³-SVM training and test times...
could be greatly reduced to $\frac{240}{150,000} \approx \frac{1}{625}$, much shorter than those of SVM$^{light}$. The scalability of M$^3$-SVMs is illustrated in Fig. 8.

4) Incremental Learning: We also tested the M$^3$-SVM incremental learning ability by successively adding one year’s patents to the training set. The results are shown in Fig. 9. As we expected, the training time of incremental learning grew linearly, while the training time of no incremental learning grew quadratically.

V. CONCLUSION

We have used M$^3$-SVM to address a large-scale patent classification problem on the IPC top layer. M$^3$-SVM provides better generation performance than conventional SVM,

<table>
<thead>
<tr>
<th>TABLE III</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MEAN AND STANDARD DEVIATION VALUES FOR M$^3$-SVM AND SVM$^{light}$</strong></td>
</tr>
<tr>
<td>classifier</td>
</tr>
<tr>
<td>Macro precision</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Macro recall</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Macro F1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Micro F1</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Fig. 8. Training and test times are inversely proportional to the number of CPU cores.

Fig. 9. Incremental learning ability of M^3-SVMs.

and more stable performance than conventional SVM with regard to parameter tuning. The training time of M^3-SVM can be greatly reduced. Experimental results showing that training and test times are inversely proportion to the number of CPU cores demonstrate the scalability of the parallel M^3-SVM system. We also verified the incremental learning ability of M^3-SVM, which is a valuable property for large-scale patent. Consequently, the training time is reduced from quadratic complexity to linear complexity. In the future, the multi-label and hierarchical structure of patents will be considered with the goal of solving the patent classification problem integrally.

VI. ACKNOWLEDGEMENTS

We sincerely thank Zhifei Ye and Feng Zhou for their helpful advices, and Cong Hui for his coding help. This work was supported in part by the National Natural Science Foundation of China via the grant NSFC 60773090, the Microsoft Laboratory for Intelligent Computing and Intelligent Systems of Shanghai Jiao Tong University, and the Okawa Foundation Research Grant.

REFERENCES