Scalable Multi-Instance Learning

Xiu-Shen Wei, Jianxin Wu and Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023
Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing 210023
{*weixs, wujx, zhouzh*}*@lamda.nju.edu.cn*

***Abstract*—Multi-instance learning (MIL) has been widely applied to diverse applications involving complicated data objects such as images and genes. However, most existing MIL algorithms can only handle small- or moderate-sized data. In order to deal with the large scale problems in MIL, we propose an efficient and scalable MIL algorithm named miFV. Our algorithm maps the original MIL bags into a new feature vector representation, which can obtain bag-level information, and meanwhile lead to excellent performances even with linear classifiers. In consequence, thanks to the low computational cost in the mapping step and the scalability of linear classifiers, miFV can handle large scale MIL data efficiently and effectively. Experiments show that miFV not only achieves comparable accuracy rates with state-of-the-art MIL algorithms, but has hundreds of times faster speed than other MIL algorithms.**

***Keywords*-multi-instance learning; scalability; efficiency; large scale data**

I. Introduction

During the investigation of drug activity prediction [1], the multi-instance learning (MIL) framework was formally proposed and naturally applied to this problem. In contrast to traditional supervised learning, MIL receives a set of bags labeled positive or negative, rather than receiving a set of instances which have labels. In addition, instances in the MIL bags have no label information. The task of MIL is to train a classifier that labels new bags, and MIL has already been widely applied in diverse applications, e.g., image categorization [2], text categorization [3], face detection [4], computer-aided medical diagnosis [5], web mining [6], etc.

Over the last few years, many effective MIL algorithms have been developed [7]. These algorithms achieve decent accuracy rates in different MIL applications, which might partly attribute to the fact that objects are represented as bags in MIL, which can naturally encode the original objects. However, directly processing and classifying the complicated bag representation means that the complexity of MIL's hypothesis space also becomes much larger. This fact leads to an undesired outcome: most existing MIL algorithms are usually time-consuming and incapable of handling large scale MIL problems. The real world applications of MIL, however, consistently request scalable multi-instance learning algorithms to handle millions of complex objects or examples (e.g., images, genes, etc).

In order to deal with large scale MIL problems, a natural idea is to convert the bag representation of an object to a simpler one, i.e., a vector representation. The conversion procedure should be very efficient and should keep as much information as possible, in order to achieve a scalable and accurate multi-instance learning machine. There have been some related methods in this line of research, including the CCE method [8] and the MILES algorithm [9]. However, as we will empirically show in this paper, neither the efficiency nor the accuracy of these methods is mature enough for handling large scale multi-instance learning problems.

In this paper, we propose an efficient and scalable MIL algorithm which is miFV (multi-instance learning based on the Fisher Vector representation). In miFV, bags can be mapped by its mapping function into a new feature vector representation. The major difference between miFV and CCE/MILES is that miFV encodes more information into the new feature vector. Moreover, miFV is efficient to compute, and leads to excellent results even with linear classifiers.

In consequence, thanks to the low computational cost and scalability of the mapping function and the linear classifiers, miFV can handle large scale MIL data efficiently. As shown by the results in our experiments, on small- and medium-scale MIL problems, miFV not only achieves comparable performances with state-of-the-art MIL algorithms, but has hundreds of, even thousands of times faster speed than these MIL algorithms. Moreover, on large scale MIL problems, the training process of most existing MIL algorithms did not terminate after a few days, while miFV can finish training within a few hours and achieve good classification accuracy.

The rest of this paper is organized as follows. In Section II, we introduce some related work about MIL. The proposed algorithm is presented in Section III. In Section IV, we present our experimental results. In Section V, we conclude the paper with discussions on future issues.

II. Related Work

Many multi-instance learning algorithms have been developed during the past decade [7], to name a few: Diverse Density, EM-DD, Citation-kNN, MI-Kernel, miSVM and MISVM, DD-SVM, MIBoosting, RIPPER-MI, MI-LR, MILES, MissSVM, miGraph, MIMEL, etc. These MIL algorithms solve MIL problems in diverse ways. For example, miSVM [3] is a SVM based MIL method, which searches max-margin hyperplanes to separate positive and negative instances in bags. MIBoosting [10] was proposed as a

boosting approach for MIL with the assumption that all instances contribute equally and independently to a bag's label. miGraph [11] treats instances in bags in a non-i.i.d. way and solves MIL problems by graph kernels. Existing MIL algorithms have achieved satisfactory accuracy rates. However, most of them can only handle small- or moderate-sized data, and only have limited scalability.

In order to deal with MIL problems more efficiently, some researchers had tried to solve MIL by using the similarity that represents the closeness of the bag to specific target points in the original instance space, e.g., DD-SVM [12] and MILES [9]. Some had tried to treat MIL as a special case of semi-supervised learning, e.g., MissSVM [13]. However, due to the bag representation of MIL, the complexity of the hypothesis space of existing MIL algorithms is still too large to learn very efficiently. And, handling large scale MIL data is not practical yet. For example, the MILES method took 1.2 minutes (i.e., 72 seconds) to learn on the Musk2 dataset, which is a small scale problem with only 102 bags [9].[1]

A more efficient MIL algorithm CCE [8] was proposed in order to solve MIL problems with lower computational complexity by using a histogram-based vector representation via constructive clustering. However, the information CCE extracts from bags is very limited when compared with the higher-order statistics in miFV, which makes its performance significantly worse than that of miFV. One related property of CCE is that in order to incorporate more information from the bag representation, CCE used an ensemble classifier based on multiple clusterings, which significantly increases its computational complexity and renders it incapable of handling large scale problems.

III. The Proposed Algorithm

In this paper, we propose a scalable multi-instance learning algorithm, named as miFV (multi-instance learning based on the Fisher Vector representation). The method converts the bag representation into a vector form very efficiently, while maintaining useful information inside the bags, as will be verified by our experiments later in Section IV.

The FV (Fisher Vector) representation [14] is an approach in computer vision to encode a set of local patch descriptors extracted from one image, e.g., SIFT, into a high dimensional vector and to pool them into an image-level signature. Because of its efficiency, effectiveness and scalability, FV becomes state-of-the-art approach in computer vision and has shown excellent performances in many applications, e.g., large scale image retrieval and image categorization. Moreover, in these years, some studies illustrate its effectiveness and efficiency in both theoretical and practical aspects [14].

An image is represented as a set of local patches. This set representation is related to, but different from the bag representation in MIL. In a bag of MIL, every instance has a label (although unknown during the learning process), indicating whether it is a positive or negative instance. In a set representation of an image, however, the local patches do not have semantic meanings. For example, in an image labelled as "tiger", none of the local patches extracted from this image can be treated as a "tiger". However, since FV maps a set of items into one single vector, we may borrow ideas from FV to implement scalable multi-instance learning.

Here, we first give an introduction to the Fisher Kernel, and then propose the miFV algorithm.[2]

A. Fisher Kernel Basics

The Fisher Kernel is a powerful framework which combines the strengths of generative and discriminative approaches to pattern classification [15].

Let $S = \{\boldsymbol{s}_t, t = 1, \ldots, T\}$ be a sample of T observations $\boldsymbol{s}_t \in \mathcal{S}$. Let p be a probability density function which models the generative process of elements in $\mathcal{S}$ with parameters λ. Then the samples S can be described by the gradient vector:

$$G_\lambda^S = \nabla_\lambda \log p(S|\lambda). \tag{1}$$

Intuitively, the gradient describes how the parameters p should be modified to better fit the data S. Note that, the dimensionality of G_λ^S only depends on the number of parameters in p, rather than on the sample size T. In other words, it transforms a variable length set S into a fixed length vector G_λ^S, which is amenable for the mapping function $\mathcal{M}_f$ in miFV to map the bags with different numbers of instances into a fixed-length feature vector.

In [15], the Fisher Kernel (FK) was proposed to measure the similarity between two samples S_1 and S_2:

$$\mathcal{K}_{FK}(S_1, S_2) = {G_\lambda^{S_1}}' F_\lambda^{-1} G_\lambda^{S_2}, \tag{2}$$

where F_λ is the Fisher information matrix of p:

$$F_\lambda = E_{\boldsymbol{s}\sim p}[\nabla_\lambda \log p(\boldsymbol{s}|\lambda) \nabla_\lambda \log p(\boldsymbol{s}|\lambda)']. \tag{3}$$

Since F_λ is symmetric and positive definite, it has a Cholesky decomposition $F_\lambda^{-1} = {L_\lambda}' L_\lambda$. The FK in (2) can be rewritten explicitly as a dot-product:

$$\mathcal{K}_{FK}(S_1, S_2) = {\boldsymbol{f}_\lambda^{S_1}}' \boldsymbol{f}_\lambda^{S_2}, \tag{4}$$

where

$$\boldsymbol{f}_\lambda^S = L_\lambda G_\lambda^S = L_\lambda \nabla_\lambda \log p(S|\lambda). \tag{5}$$

The normalized gradient vector presented in (5) is the Fisher Vector (FV). In consequence, using a non-linear kernel machine with the $\mathcal{K}_{FK}$ kernel is equivalent to using a linear kernel machine with the FV (i.e., $\boldsymbol{f}_\lambda^S$) as feature vector.

[1] As a comparison, miFV only uses less than 0.1 second on this dataset, cf. Section IV.

[2] The code of miFV is available at http://lamda.nju.edu.cn/code_SMIL.ashx

B. Representing Bags with Fisher Vectors

Here we treat a bag X_i as a sample S mentioned above. In the traditional multi-instance learning assumption, the instances in bags are *independently and identically distributed*. In the same way, the $\boldsymbol{s}_t$'s from $\mathcal{S}$ are generated independently by p. Thus, a natural choice of p is a Gaussian Mixture Model (GMM). We can estimate the parameters of the GMM p on the training bags using Maximum Likelihood Estimation (MLE). Pseudo code of miFV is shown in Algorithm 1.

We denote the parameters of the K-component GMM by $\lambda = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, k = 1, \ldots, K\}$, where ω_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are respectively the mixture weight, mean vector and covariance matrix of the k^{th} Gaussian. In what follows, for a bag $X_i = \{\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{ij}, \ldots, \boldsymbol{x}_{i,n_i}\}$, let $\mathcal{L}(X_i|\lambda) = \log p(X_i|\lambda)$. Because of the independence assumption and the GMM model, we can rewrite this equation as follows:

$$\mathcal{L}(X_i|\lambda) = \sum_{j=1}^{n_i} \log p(\boldsymbol{x}_{ij}|\lambda) = \sum_{j=1}^{n_i} \log \sum_{k=1}^{K} \omega_k p_k(\boldsymbol{x}_{ij}|\lambda), \tag{6}$$

where the component p_k denotes the k^{th} Gaussian:

$$p_k(\boldsymbol{x}_{ij}|\lambda) = \frac{\exp\{-\frac{1}{2}(\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k)\}}{(2\pi)^{D/2} |\boldsymbol{\Sigma}_k|^{1/2}}. \tag{7}$$

In addition, the mixture weights are subject to the constraint:

$$\forall_k : \omega_k \geq 0, \sum_{k=1}^{K} \omega_k = 1, \tag{8}$$

to ensure that $p(\boldsymbol{x}_{ij}|\lambda)$ is a valid distribution. As assumed in [16], the covariance matrices are diagonal and $\boldsymbol{\sigma}_k^2$ is the variance vector.

In the following, we describe how the mapping function $\mathcal{M}_f$ in the miFV algorithm maps a bag into a Fisher Vector. The gradients of a single instance $\boldsymbol{x}_{ij}$ w.r.t. the parameters of the GMM model, $\lambda = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$, can be presented as

$$\nabla_{\omega_k} \log p(\boldsymbol{x}_{ij}|\lambda) = \gamma_j(k) - \omega_k, \tag{9}$$

$$\nabla_{\boldsymbol{\mu}_k} \log p(\boldsymbol{x}_{ij}|\lambda) = \gamma_j(k) \left(\frac{\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k^2} \right), \tag{10}$$

$$\nabla_{\boldsymbol{\sigma}_k} \log p(\boldsymbol{x}_{ij}|\lambda) = \gamma_j(k) \left[\frac{(\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k)^2}{\boldsymbol{\sigma}_k^3} - \frac{1}{\boldsymbol{\sigma}_k} \right], \tag{11}$$

where $\gamma_j(k)$ is the soft assignment, which is also the probability for $\boldsymbol{x}_{ij}$ generated by the k^{th} Gaussian:

$$\gamma_j(k) = p(k|\boldsymbol{x}_{ij}, \lambda) = \frac{\omega_k p_k(\boldsymbol{x}_{ij}|\lambda)}{\sum_{t=1}^{K} \omega_t p_t(\boldsymbol{x}_{ij}|\lambda)}. \tag{12}$$

Note that, the division and exponentiation of vectors should be understood as term-by-term operations in (10)-(12).

After obtaining the gradients, the remaining step is to compute L_λ. The methods proposed in [16] [14] supply us a closed form of L_λ and meanwhile it can be solved efficiently. Thus, the normalized gradients are presented as following [16] [14]:

$$f_{\omega_k}^{X_i} = \frac{1}{\sqrt{\omega_k}} \sum\nolimits_{j=1}^{n_i} \left(\gamma_j(k) - \omega_k \right), \tag{13}$$

$$\boldsymbol{f}_{\boldsymbol{\mu}_k}^{X_i} = \frac{1}{\sqrt{\omega_k}} \sum\nolimits_{j=1}^{n_i} \gamma_j(k) \left(\frac{\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right), \tag{14}$$

$$\boldsymbol{f}_{\boldsymbol{\sigma}_k}^{X_i} = \frac{1}{\sqrt{\omega_k}} \sum\nolimits_{j=1}^{n_i} \gamma_j(k) \frac{1}{\sqrt{2}} \left[\frac{(\boldsymbol{x}_{ij} - \boldsymbol{\mu}_k)^2}{\boldsymbol{\sigma}_k^2} - 1 \right]. \tag{15}$$

Recall that the dimension of the instances in bag X_i is d. We can find that $f_{\omega_k}^{X_i}$ in (13) is a scalar, while $\boldsymbol{f}_{\boldsymbol{\mu}_k}^{X_i}$ and $\boldsymbol{f}_{\boldsymbol{\sigma}_k}^{X_i}$ are d-dimensional vectors. Thus, the FV $\boldsymbol{f}_\lambda^{X_i}$ which can describe a bag X_i should be concatenated by $f_{\omega_k}^{X_i}$, $\boldsymbol{f}_{\boldsymbol{\mu}_k}^{X_i}$ and $\boldsymbol{f}_{\boldsymbol{\sigma}_k}^{X_i}$, for all $k = 1, \ldots, K$ Gaussian components. In the following, each element of $\boldsymbol{f}_\lambda^{X_i}$ is sign square rooted by $[\boldsymbol{f}_\lambda^{X_i}]_j \leftarrow \text{sign}([\boldsymbol{f}_\lambda^{X_i}]_j)\sqrt{|[\boldsymbol{f}_\lambda^{X_i}]_j|}$ [14]. Then the new feature vector $\boldsymbol{f}_\lambda^{X_i}$ is subsequently L_2-normalized. Thus, the mapping function $\mathcal{M}_f$ maps the bag X_i into a $(2d+1)K$-dimensional normalized FV, i.e., $\boldsymbol{f}_\lambda^{X_i}$.

Finally, we feed $(\boldsymbol{f}_\lambda^{X_i}, y_i)$ to a standard supervised learner, e.g., a support vector machine, to learn a classification model $\mathcal{F}$. A bag $X_{i'}$ in the testing set will be firstly mapped into a new feature vector $\boldsymbol{f}_\lambda^{X_i'}$ by $\mathcal{M}_f$. Then, we can get the bag-level prediction via $\mathcal{F}(\boldsymbol{f}_\lambda^{X_i'})$.

Algorithm 1 The miFV algorithm

Input:
  Training data $\{(X_1, y_1), \ldots, (X_i, y_i), \ldots, (X_{N_B}, y_{N_B})\}$
Train:
  Estimate parameters $\lambda = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$ of the GMM p on the training bags with MLE
  for $i = 1$ **to** N_B **do**
    Map the bag X_i into a FV $\boldsymbol{f}_\lambda^{X_i} \leftarrow \mathcal{M}_f(X_i, p)$
    $[\boldsymbol{f}_\lambda^{X_i}]_j \leftarrow \text{sign}([\boldsymbol{f}_\lambda^{X_i}]_j)\sqrt{|[\boldsymbol{f}_\lambda^{X_i}]_j|}$
    $\boldsymbol{f}_\lambda^{X_i} \leftarrow \boldsymbol{f}_\lambda^{X_i} / \parallel \boldsymbol{f}_\lambda^{X_i} \parallel_2$
  end for
  Use new training set $\{(\boldsymbol{f}_\lambda^{X_1}, y_1), \ldots, (\boldsymbol{f}_\lambda^{X_{N_B}}, y_{N_B})\}$ to learn a classifier $\mathcal{F}$
Test:
  for all test bags $X_{i'}$ $(i' \in \{1, 2, \ldots, N_B'\})$ **do**
    Map the bag $X_{i'}$ into a FV $\boldsymbol{f}_\lambda^{X_{i'}} \leftarrow \mathcal{M}_f(X_{i'}, p)$
    $[\boldsymbol{f}_\lambda^{X_{i'}}]_j \leftarrow \text{sign}([\boldsymbol{f}_\lambda^{X_{i'}}]_j)\sqrt{|[\boldsymbol{f}_\lambda^{X_{i'}}]_j|}$
    $\boldsymbol{f}_\lambda^{X_{i'}} \leftarrow \boldsymbol{f}_\lambda^{X_{i'}} / \parallel \boldsymbol{f}_\lambda^{X_{i'}} \parallel_2$
  end for
  Output the prediction $\mathcal{F}(\boldsymbol{f}_\lambda^{X_{i'}})$

C. Efficiency and Scalability of miFV

As illustrated in the previous section, FV is efficient to compute. Moreover, there is no need to use costly kernels

to implicitly project these very high-dimensional gradient vectors into a still higher dimensional space [16], [14]. Hence, it leads to excellent accuracy rates even with linear classifiers. In the same way, miFV based on the FV representation is also computed efficiently to represent the bags with the FVs. In addition, linear SVM and stochastic gradient descent methods can solve large scale classification problems with a huge number of instances and features very efficiently. Because it is difficult to make formal complexity analysis of this complex framework, we empirically validate the efficiency and scalability of miFV in our experiments.

IV. Experiments

A. Data Sets and Experimental Setup

Our experiments are performed on five MIL benchmark data sets, four moderate-sized data sets for image and document classification, and finally one large scale data set for video annotation. On these data sets, we compare miFV with six state-of-the-art MIL algorithms: *MIWrapper* [17], *CCE* [8], *EM-DD* [18], *miSVM* [3], *MIBoosting* [10] and *miGraph* [11]. In addition, *Simple-MI* is also a baseline method [7], which represents a bag with the mean vector of all the instances in that bag. For Simple-MI and miFV, we take LIBLINEAR as the final linear classifier.

We first evaluate miFV on five benchmark data sets popularly used in the studies of MIL, including *Musk1*, *Musk2*, *Elephant*, *Fox* and *Tiger*. In addition, two famous categories, i.e., *Course* and *Faculty*, in WebKB are used in experiments. More details of these data sets can be found in [19]. On each of the seven data sets above, we run ten times 10-fold cross validation and report the average results.

As image categorization is one of the most successful applications of MIL, two image data sets (*1000-Image* and *2000-Image*) for classification are also used in our experiments. We treat each image as a bag and employ the *SBN* bag generator [20] to extract instances/patches from each bag/image. For these image data sets, we use the same experimental routine as described in [11]. The experiment is repeated five times and the average results are reported.

Finally, we use a well known and large scale data set in computer vision, i.e., the development (DEV) set of *TRECVID 2005* for the video annotation task, to validate the scalability of miFV. The data set contains 61901 sub-shots associated with one or more concepts in the 39 concepts. We treat each sub-shot as a bag and each frame in one sub-shot as an instance. Each bag/sub-shot obtains 11 instances/keyframes of 1000-dimention. We split the data set into three parts, i.e., training data, validation data and test data. These three parts have 40616, 9331 and 11954 bags, respectively. For each concept, if we treat it as the positive label, then the other concepts become the negative ones. Thus, we can get 39 MIL datasets, which are sub-problems of the original dataset. For each subdataset, we build a binary classifier five times with five random balanced under-samplings to solve the class imbalance problem. Detailed characteristics of these data sets are summarized in Table I.

Table I
Detailed characteristics of the data sets. Note that, because we use the one-against-one strategy for the *1000-Image* and *2000-Image* data sets, "♯ positive" ("♯ negative") for them presents the number of positive (negative) bags in each round.

Dataset	♯ attribute	♯ bag			♯ instance
		♯ positive	♯ negative	♯ total	
Musk1	166	47	45	92	476
Musk2	166	39	63	102	6,598
Elephant	230	100	100	200	1,220
Fox	230	100	100	200	1,320
Tiger	230	100	100	200	1,391
Course	320	674	674	1,348	3,528
Faculty	361	795	795	1,590	4,248
1000-Image	121	100	100	1,000	3,000
2000-Image	121	100	100	2,000	3,000
TRECVID 2005	1,000	–	–	61,901	680,911

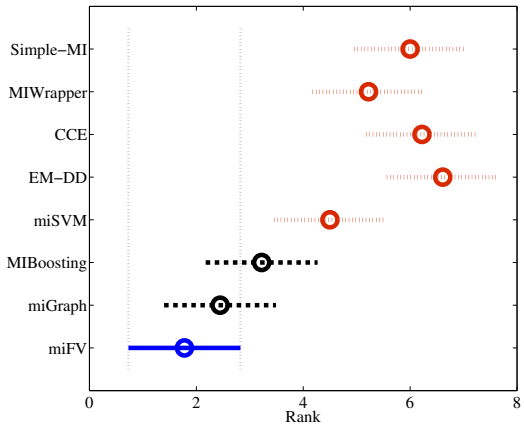

Figure 1. Friedman test results.

B. Accuracy Comparison

We first report the comparison results on the nine MIL data sets (i.e., all data sets except for *TRECVID 2005*) in Table II. As shown in the table, miFV achieves the best or the second best performance in almost all the cases. To have an overall evaluation of the performances of the algorithms over these nine data sets, we performed the Friedman test. As shown in Figure 1, the average rank of miFV on these data sets is 1st, and miFV performs significantly better than Simple-MI, MIWrapper, CCE, EM-DD and miSVM. Meanwhile, it can achieve comparable performances with two state-of-the-art MIL algorithms, i.e., MIBoosting and miGraph.

C. Efficiency Comparison

Because our goal is to handle large scale MIL problems, it is crucial to study the efficiency of the proposed algorithm. We only report the time cost of training time in Figure 2. The time cost of test time has almost identical trend of training time. Note that the vertical axes are *shown in log-scale* in

Table II
COMPARISON RESULTS (MEAN±STD.) ON 9 DATA SETS. THE HIGHEST AVERAGE ACCURACY OF EACH COLUMN IS MARKED IN BOLD AND WITH •; THE SECOND HIGHEST ONE OF EACH COLUMN IS JUST IN BOLD.

Algorithm \ Data set	Musk1	Musk2	Elephant	Fox	Tiger	Course	Faculty	1000-Image	2000-Image
Simple-MI	.832±.123	.853±.111	.801±.088	.546±.092	.778±.092	.896±.007	.910±.008	.844±.090	.818±.099
MIWrapper	.849±.106	.796±.106	.827±.088	.582±.102	.770±.092	.929±.009	.906±.004	.847±.086	.831±.092
CCE	.831±.027	.713±.024	.793±.021	.599±.027	.760±.012	.936±.006	.934±.006	.805±.102	.801±.095
EM-DD	.849±.098	.869±.108	.771±.098	.609±.101	.730±.096	.538±.120	.410±.008	.741±.145	.739±.139
miSVM	.874±.120	.836±.088	.822±.073	.582±.102	.789±.089	.915±.010	.915±.014	.854±.148	.849±.139
MIBoosting	.837±.120	.790±.088	.827±.073	**.638±.102** •	.784±.089	.938±.019	**.941±.017**	**.910±.060** •	**.898±.063** •
miGraph	**.889±.073**	**.903±.086** •	**.869±.078** •	.616±.079	**.801±.083**	**.980±.001** •	.854±.006	.896±.070	**.896±.072**
miFV	**.909±.089** •	**.884±.094**	**.852±.081**	**.621±.109**	**.813±.083** •	**.968±.009**	**.961±.007** •	**.899±.070**	.882±.070

Table III
RESULTS OF THE *TRECVID 2005* DATA SET. NOTE THAT, THE BEST RESULT OF EACH CRITERION IS MARKED IN BOLD; THE OTHER MIL ALGORITHMS, I.E., *CCE*, *EM-DD*, *miSVM*, *MIBoosting* AND *miGraph*, COULD NOT RETURN RESULTS IN 48 HOURS.

	mi.p.↑	mi.r.↑	mi.f.↑	ma.p.↑	ma.r.↑	ma.f.↑
Simple-MI	.229	.725	.348	.249	.710	.350
MIWrapper	.243	**.733**	.365	.273	**.718**	.372
miFV	**.275**	.723	**.398**	**.325**	.706	**.412**

these figures. All the experiments are performed on a machine with 4×3.10 GHz CPUs and 8GB main memory.

Obviously, except for Simple-MI, miFV is the most efficient MIL algorithm on all the data sets. EM-DD is the most time-consuming one, followed by miSVM. For CCE, because it trains an ensemble of classifiers based on multiple clusterings, it is not efficient enough and even worse than EM-DD and miGraph on the four moderate-sized data sets. Compared with the accuracy-wise comparable algorithms, i.e., MIBoosting and miGraph, miFV has hundreds of times faster speed. Especially for the four moderate-sized data sets, the efficiency of miFV is more prominent than other state-of-the-art MIL algorithms.

D. Scalability

Here, we present the results of the large scale data set (*TRECVID 2005*) in Table III. Because this data set contains 39 subdatasets, we evaluate the general performances on six commonly used criteria: *micro-averaged precision* (mi.p.), *micro-averaged recall* (mi.r.), *micro-averaged F-score* (mi.f.), *macro-averaged precision* (ma.p.), *macro-averaged recall* (ma.r.) and *macro-averaged F-score* (ma.f.). We can find that miFV achieves the best performance on four criteria, especially on mi.f. and ma.f. in Table III. Due to their high computational complexity, the other five MIL algorithms could not return result in 48 hours, even for 25% sampling examples of the original training data. By contrast, miFV can return results in several hours.

E. Parameter Analysis

In practice, before using the mapping function in miFV, we can use Principal Component Analysis (PCA) to reduce noise within the original instances in bags, or reduce dimension of them if necessary. In addition, as aforementioned, miFV has one important parameter, i.e., the number of Gaussian components. In this section, we report the results of miFV with different parameter values in Table IV. We first fix the value of PCA energy on 1.0. As shown in Table IV, miFV can achieve the best performance in most cases when the number of Gaussian components is 1. Then, the value of the number of centroids is fixed. For the two image data sets, miFV achieves the best performance when we do not use PCA. However it can get the best performance when PCA is used on the two document classification data sets. That may be related to the noises in the document data sets, which can be overcome by using PCA to get better performance.

V. CONCLUSION

Multi-instance learning has achieved great success in applications with complicated objects such as image and gene categorization. However, existing MIL algorithms are usually time-consuming to deal with large scale problems. In this paper, we propose an efficient and scalable MIL algorithm, i.e., miFV, which transforms the bag form of MIL into a new feature vector representation. On one hand, miFV has hundreds of, even thousands of times faster speed than state-of-the-art MIL algorithms; on the other hand, miFV can achieve comparable performances with other MIL algorithms. In the future, it will be interesting to consider introducing label information into the bag re-representation process.

Acknowledgment: This research was supported by NSFC (61333014). Authors want to thank Dan Zhang for the WebKB datasets, Yu-Feng Li and Nan Li for reading draft, and anonymous reviewers for helpful comments. J. Wu is the corresponding author of the paper.

REFERENCES

[1] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the Multiple Instance Problem with Axis-Parallel Rectangles," *Artificial Intelligence*, vol. 89, no. 1-2, pp. 31–71, 1997.

[2] X.-F. Song, L.-C. Jiao, S.-Y. Yang, X.-R. Zhang, and F.-H. Shang, "Sparse Coding and Classifier Ensemble based Multi-Instance Learning for Image Categorization," *Signal Processing*, vol. 93, pp. 1–11, 2013.

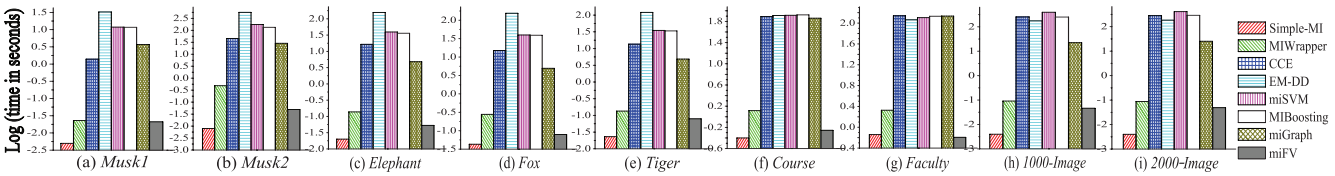

Figure 2. Comparison of mean time cost of **trainning time** on nine data sets. The time cost of test time has almost identical trend of training time.

Table IV
PARAMETER ANALYSIS RESULTS (MEAN±STD.) ON NINE DATA SETS. THE HIGHEST AVERAGE ACCURACY OF EACH COLUMN IS MARKED IN BOLD.

♯ of centers \ Data set	Musk1	Musk2	Elephant	Fox	Tiger	Course	Faculty	1000-Image	2000-Image
1	.875±.106	.861±.106	**.852±.081**	**.560±.099**	**.789±.091**	**.943±.008**	**.930±.010**	.879±.075	.875±.072
2	**.909±.089**	**.864±.096**	.829±.091	.542±.096	.765±.097	.932±.008	.923±.013	**.899±.070**	**.882±.070**
3	.888±.098	.844±.123	.806±.093	.538±.128	.712±.107	.932±.009	.919±.009	.881±.070	.879±.073
4	.889±.897	.835±.113	.781±.096	.554±.113	.708±.115	.932±.008	.921±.016	.882±.068	.877±.073
5	.864±.104	.831±.131	.764±.109	.531±.122	.686±.107	.930±.006	.922±.012	.881±.073	.878±.070
Data set / PCA energy	**Musk1**	**Musk2**	**Elephant**	**Fox**	**Tiger**	**Course**	**Faculty**	**1000-Image**	**2000-Image**
1.0	**.909±.089**	**.864±.096**	.829±.091	.542±.096	.765±.097	.932±.008	.923±.013	**.899±.070**	**.882±.070**
0.9	.840±.116	.843±.116	**.851±.079**	.595±.103	.795±.084	**.968±.009**	**.960±.008**	.836±.107	.833±.097
0.8	.763±.117	.752±.126	.836±.087	**.621±.109**	**.813±.086**	.964±.010	.959±.005	.793±.094	.798±.095

[3] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support Vector Machines for Multiple-Instance Learning," in *Advances in Neural Information Processing Systems 15*. Cambridge, MA: MIT Press, 2003, pp. 561–568.

[4] C. Zhang and P. Viola, "Multi-Instance Learning Pruning for Learning Efficient Cascade Detectors," in *Advances in Neural Information Processing Systems 20*. Cambridge, MA: MIT Press, 2008, pp. 1681–1688.

[5] G. Fung, M. Dundar, B. Krishnapuram, and R. B. Rao, "Multiple Instance Learning for Computer Aided Diagnosis," in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 425–432.

[6] B. Li, W. Xiong, and W. Hu, "Web Horror Image Recognition based on Context-Aware Multi-Instance Learning," in *Proc. IEEE 11th Int'l Conf. Data Mining*, Vancouver, Canada, 2011, pp. 1158–1163.

[7] J. Amores, "Multiple Instance Classification: Review, Taxonomy and Comparative Study," *Artificial Intelligence*, vol. 201, pp. 81–105, 2013.

[8] Z.-H. Zhou and M.-L. Zhang, "Solving Multi-Instance Problems with Classifier Ensemble based on Constructive Clustering," *Knowledge and Information Systems*, vol. 11, pp. 155–170, 2007.

[9] Y. Chen, J. Bi, and J.-Z. Wang, "MILES: Multiple-Instance Learning via Embedded Instance Selection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1931–1947, 2006.

[10] X. Xu and E. Frank, "Logistic Regression and Boosting for Labeled Bags of Instances," in *Proc. 8th Pacific-Asia Conf. Knowledge Discovery and Data Mining*, Sydney, Australia, 2004, pp. 272–281.

[11] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li, "Multi-Instance Learning by Treating Instances As Non-I.I.D. Samples," in *Proc. 26th Int'l Conf. Machine Learning*, Montreal, Canada, 2009, pp. 1249–1256.

[12] Y.-X. Chen and J. Z. Wang, "Image Categorization by Learning and Reasoning with Regions," *J. Machine Learning Research*, vol. 5, pp. 913–939, 2004.

[13] Z.-H. Zhou and J.-M. Xu, "On the Relation between Multi-Instance Learning and Semi-Supervised Learning," in *Proc. 24th Int'l Conf. Machine Learning*, Corvallis, OR, 2007, pp. 1167–1174.

[14] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image Classification with the Fisher Vector: Theory and Practice," *Int'l J. Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[15] T. Jaakkola and D. Haussler, "Exploiting Generative Models in Discriminative Classifiers," in *Advances in Neural Information Processing Systems 11*. Cambridge, MA: MIT Press, 1999, pp. 487–493.

[16] F. Perronnin and C. Dance, "Fisher Kernels on Visual Vocabularies for Image Categorization," in *Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, 2007, pp. 1–8.

[17] E. T. Frank and X. Xu, "Applying propositional learning algorithms to multi-instance data," University of Waikato, Department of Computer Science, University of Waikato, Hamilton, NZ, Tech. Rep., 2003.

[18] Q. Zhang and S. A. Goldman, "EM-DD: An Improved Multiple-Instance Learning Technique," in *Proc. 16th IEEE Int'l Conf. Data Eng.*, San Diego, California, USA, 2000, pp. 233–243.

[19] D. Zhang, J. He, and R. Lawrence, "MI2LS: Multi-Instance Learning from Multiple Information Sources," in *Proc. 19th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, Chicago, Illinois, 2013, pp. 149–157.

[20] O. Maron and A. L. Ratan, "Multiple-Instance Learning for Natural Scene Classification," in *Proc. 18th Int'l Conf. Machine Learning*, Williamstown, MA, 2001, pp. 425–432.