International Journal of Innovative
Computing, Information and Control
Volume **11**, Number **3**, June **2015**

 ISSN 1349-4198
pp. 921–934

A FRAMEWORK OF HASHING FOR MULTI-INSTANCE MULTI-LABEL LEARNING

Man Liu and Xinshun Xu*

School of Computer Science and Technology
Shandong University
No. 1500, Shunhua Road, Jinan 250101, P. R. China
liuman.sdu@outlook.com; *Corresponding author: xuxinshun@sdu.edu.cn

Received October 2014; revised January 2015

Abstract. *Multi-instance multi-label learning (*Miml*) is a powerful framework, which deals with the problem that each example is represented as multiple instances and associated with multiple class labels. Previous works mostly focus on accuracy, while scalability for large scale datasets has been rarely addressed. In this paper, we present a novel framework – Multi-instance Multi-label Hashing (*MimlH*) to tackle both accuracy and scalability issues of* Miml *tasks, which means that it can not only get good accuracy, but also fast learning speed.* MimlH *leverages hashing technique. Specifically, it exploits the hashing approach in two perspectives – bag-level hashing and instance-level hashing, which replaces the dot-product kernel operator in the previous methods and effectively maps the entire samples into hamming space, speeding up the process of learning tremendously. Moreover, we also take the label information into account to enhance our framework. We evaluate our approach on two popular data sets of* Miml *task, which were derived from two real world applications – scene classification and text categorization. The experimental results show that the proposed framework performs better than previous works on accuracy and efficiency in a balanced way.*
Keywords: Multi-instance multi-label learning, Hashing, Scene classification, Text categorization

1. **Introduction.** Multi-instance multi-label learning (Miml) is a novel framework which was initially derived from scene classification. In such a task, one image has multiple labels owing to its complicated semantics; in addition, one image can be represented as multiple instances because different image regions often provide different hints for the labels [1]. Thus, in Miml, each example in the training set is associated with multiple instances as well as multiple labels, which provides a natural formulation for those real-world tasks involving ambiguous objects. Later, Miml was also applied to text categorization, bioinformatics, image annotation, etc. [2, 3].

Recent years, many powerful approaches have been proposed for Miml problems, including MimlBoost, MimlSvm, MimlSvmmi, MimlNN, M^{3}Miml, D-Miml, InsDif, SubCod, M3LDA, etc. [1-4]. Some of them solve the Miml problem in a degenerated version, where Miml is transformed into single-instance multi-label learning (Siml) or multi-instance single-label learning (Misl) first and then tackled by existing solutions. Others resolve it in a direct way, where the problem is formalized into a regularization one and then tackled directly. Moreover, Miml can also be used for Misl and Siml problems by transforming examples into Miml representation and then addressing it by the existing solutions.

From the above, we can find that Miml problems widely exist in our real world; some algorithms in Miml framework have been proposed for such problems. However, most of

them only focus on accuracy, while the scalability for large scale data sets has been rarely addressed. Thus, an MIML model will be much useful if it could consider both the accuracy and scalability of such tasks. Motivated by this, in this paper, we present a novel approach – Multi-instance Multi-label Hashing (MIMLH) to tackle both accuracy and scalability issues of MIML. MIMLH exploits the hashing approach to solve the problem in two perspectives – bag-level and instance-level, which replaces the dot-product kernel operator in the previous methods and then effectively maps the entire samples into hamming space, speeding up the process of learning tremendously. Moreover, we also take the label information into account to enhance our framework. First, we use the original dataset to get the predicted labels of the testing set. Then, we embed the original training labels and predicted testing labels into the initial features to construct a new dataset. Finally, we apply the new dataset to get the results. We evaluate our approach on two popular data sets of MIML problems, which were derived from two real world applications – scene classification and text categorization. The experimental results show that the proposed framework performs better than some state-of-the-art works on accuracy and efficiency in a balanced way.

The rest of this paper is organized as follows. Section 2 reviews the related works including MIML and hashing approaches. Then, we propose our novel framework – the Multi-instance Multi-label Hashing (MIMLH) in Section 3. Section 4 reports the experimental results on two popular data sets of MIML. Finally, Section 5 concludes this paper and indicates several issues for future work.

2. Problem Statement and Preliminaries.

Our work is closely related to multi-instance multi-label learning and hashing approaches. Thus, in this section, we briefly introduce some related works.

2.1. Multi-instance multi-label learning.

In multi-instance multi-label learning, letting $\mathcal{X} = \mathbb{R}^d$ denote the input space of instances and $\mathcal{Y} = \{1, 2, \ldots, L\}$ the set of class labels, then the MIML training examples can be represented as $\{(X_i, Y_i) \mid 1 \leq i \leq N\}$, where $X_i \subseteq \mathcal{X}$ is a bag of instances $\{x_1^i, x_2^i, \ldots, x_{n_i}^i\}$ and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with X_i. Here, n_i is the number of instances in X_i and l_i the number of labels in Y_i. MIML aims to learn a function $f_{MIML} : 2^{\mathcal{X}} \to 2^{\mathcal{Y}}$ from the training sets and study the ambiguity in both input space and output space. Apparently, the framework of MIML is closely related to the learning framework of multi-instance learning [5], multi-label learning [6, 7].

Multi-instance learning [5], or multi-instance single-label learning (MISL), was originated from drug activity prediction problem by Dietterich et al. The task of MISL is to learn a function $f_{MISL} : 2^{\mathcal{X}} \to \{+1, -1\}$ from a set of MISL training examples $\{(X_i, y_i) \mid 1 \leq i \leq N\}$, where $X_i \subseteq \mathcal{X}$ is a bag of instance $\{x_1^i, x_2^i, \ldots, x_{n_i}^i\}$ and $y_i \in \{+1, -1\}$ is the binary label of X_i. Multi-label learning [6, 7], or single-instance multi-label learning (SIML), was derived from the investigation of text categorization problems. The task of MIML is to learn a function $f_{SIML} : \mathcal{X} \to 2^{\mathcal{Y}}$ from a set of MIML training examples $\{(x_i, Y_i) \mid 1 \leq i \leq N\}$, where $x_i \in \mathcal{X}$ is an instance and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with x_i. MISL and SIML study the ambiguity in the input space and output space, respectively. A number of MISL and SIML learning algorithms have been proposed [8-16], and applied to many applications successfully, including text and image categorization [6, 17-23]. More related works on MISL and SIML can be found in [24, 25].

According to the above definitions, it can be seen that the traditional supervised learning (SISL) is a degenerated version of either MISL or SIML. Moreover, SISL, MISL, SIML

can all be regarded as degenerated versions of MIML. Therefore, using MISL or SIML as a bridge becomes an intuitive way to solve the MIML task [1, 4]. From this point of view, many typical methods have been proposed including MIMLBOOST, MIMLSVM, MIMLNN, MIMLSVM-mi, etc. Later, considering the information loss of the degenerated version reformulation, some direct ways have been put forward by explicitly exploiting the connections between the instances and labels, including regularization framework D-MIML, maximum-margin method M^{3}MIML, probabilistic generative model DBA approach, RankingLoss approach, topic-model M3LDA, etc. [2-4, 26, 27]. Moreover, MIML can also be used for multi-instance single-label learning and single-instance multi-label learning by transforming the given data sets into MIML samples firstly and then addressing them by the existing solutions. However, most of the above methods only focus on accuracy, while the scalability for large scale data sets has been rarely addressed.

2.2. **Hashing approaches.** Hashing is an effective technique for approximate nearest neighbor search with rapid speed. In the recent years, many hashing methods have been proposed, such as locality sensitive hashing (LSH) [28], spectral hashing (SpH) [29], and self-taught hashing (STH) [30]. By mapping data points into hamming space, hashing methods can obtain nearest neighbor search in sub-linear time. The critical factors for a successful code include three aspects: (1) easily computing for a novel input; (2) requiring a small number of bits to code the full dataset; (3) mapping similar items to similar binary codewords.

The intuition behind LSH [28, 31] is that at least one of the hash functions can hash nearby data points into a same bucket with high probability. Therefore, LSH uses a family of locality sensitive hash functions composed of linear projection over random directions in the feature space. It preserves the similarity of the items and could be easily computed. However, the precision improves with the increasement of the number of bits, which ends with very inefficient codes with long bits. Recently, for the possibility of performing real-time search due to the quick similarity computation by using bit XOR operation in the Hamming space, compact binary code approaches [32] such as spectral hashing (SH) [29], self-taught hashing (STH) [30] were proposed. Thus, how to generate the compact binary codes for the data points becomes the primary challenge. In the learning phase, spectral hashing applies spectral graph partitioning to get the hash codes of training data, which is similar to self-taught hashing. In addition, in order to calculate the binary codes for a new data point, spectral hashing assumes that the data are uniformly distributed in a hype-rectangle.

Rank correlation measures are known for their resilience to perturbations in numeric values and are widely used in many evaluation metrics. Such ordinal measures have been rarely applied in treatment of numeric features as a representational transformation. In this paper, we use the Winner-Take-All Hashing (WTA) [33], which is a family of algorithms where each WTA hash function defines an ordinal embedding and a related rank-correlation similarity measure. In brief, WTA is well suited as a basis for locality-sensitive hashing and offers a degree of invariance with respect to perturbations in numeric values [33].

3. Multi-instance Multi-label Hashing.

3.1. **The framework of MIMLH.** Figure 1 illustrates the flowchart of MIMLH, which can be separated into two stages: training stage and testing stage. For each stage, we describe it as two steps: without label step and with label step.

In summary, MIMLH works in the following way.

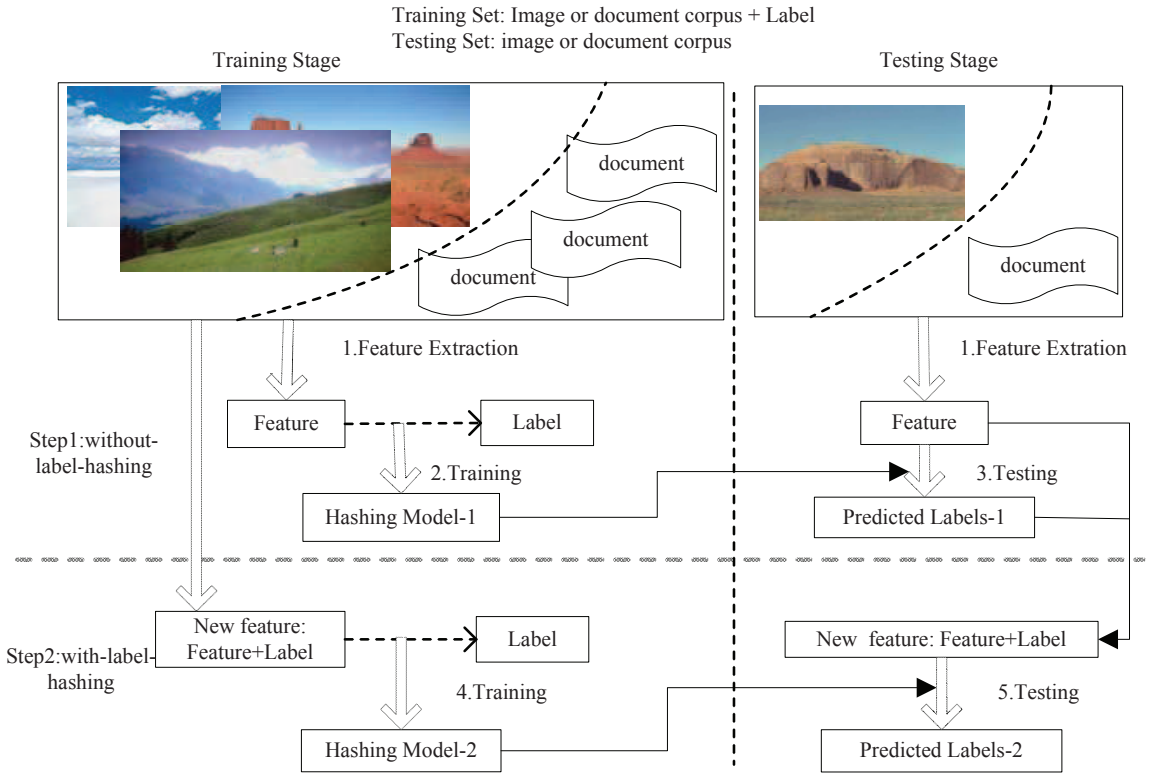


FIGURE 1. The framework of Multi-instance Multi-label Hashing

1. Each image/document is represented as a bag of instances with multi-labels. Leting $\mathcal{X} = \mathbb{R}^d$ denote the input space of instances and $\mathcal{Y} = \{1, 2, \ldots, L\}$ the set of class labels, then the MIML training examples can be represented as $\{(X_i, Y_i) \mid 1 \leq i \leq N\}$, where $X_i \subseteq \mathcal{X}$ is a bag of instances $\{x_1^i, x_2^i, \ldots, x_{n_i}^i\}$ and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with X_i. Here n_i is the number of instances in X_i and l_i the number of labels in Y_i.
2. In the training stage, we use the feature and labels of the training samples to learn a hashing model (Hashing model-1). Here the hashing model can be implemented in two perspectives, which will be illustrated in the next section, and now we denote it as:
$$f_{MIMLH} : 2^{\mathcal{X}} \to 2^{\mathcal{Y}} \tag{1}$$
3. In the testing stage, we use the learned hashing model to predict the test samples' labels (Predicted labels-1).
4. In the training stage, we embed the original labels to the training feature, ending in a reconstructed dateset with label information embedded in feature. The reconstructed dataset can be represented as $\{(X_i', Y_i) \mid 1 \leq i \leq N\}$, where $X_i' \subseteq \mathcal{X}'(\mathcal{X}' = \mathbb{R}^{d+L})$ is a bag of instances $\{(x_1^{i'}, Y_i), (x_2^{i'}, Y_i), \ldots, (x_{n_i}^{i'}, Y_i)\}$ and $Y_i \subseteq \mathcal{Y}$ is a set of labels $\{y_1^i, y_2^i, \ldots, y_{l_i}^i\}$ associated with X_i'.

   Then, we use the reconstructed dateset to learn a new hashing model (Hashing model-2), denoted as:
$$f'_{MIMLH} : 2^{\mathcal{X}'} \to 2^{\mathcal{Y}} \tag{2}$$
5. In the testing stage, we use the new hashing model to predict the new test samples' labels (Predicted labels-2).

In the end, we can get the "Predicted labels-1" from the without label hashing step and the "Predicted labels-2" from the with label hashing step. This means that we can get a better accuracy by introducing the label hashing step, and a better efficiency by using the hashing models.

Note that we use WTA [33] as the hashing method and the MimlSvm and MimlNN as the Miml methods in the experiments; however, other hashing methods and Miml methods can also be adapted to this framework.

3.2. **Two perspectives of hashing.** Here the hashing model can be implemented in two perspectives: bag-level and instance-level.

3.2.1. *Bag-level hashing.* Inspired by the original solutions, we can tackle the Miml problem by identifying its equivalence in the traditional supervised learning framework, e.g., using multi-label learning as the bridge [1, 4]. In brief, we first transform the Miml problem into Siml problem, and then hash the whole bag into binary codes. Consequently, we can solve it by a multi-label learning method. In this paper, we use the MlSvm and a two-layer Neural Network. We call them as MimlHSvmB and MimlHNNB, respectively.

In detail, we first perform k-medoids on training data, and divide them into k partitions whose medoids are $M_t(t = 1, 2, \ldots, k)$, respectively. Then, we transform the original multi-instance examples X_u into a k-dimensional numerical vector z_u, where the $i_{th}(i = 1, 2, \ldots, k)$ component of z_u is the distance between X_u and M_i, namely, $d_H(X_u, M_i)$, where $d_H(A, B)$ is the Hausdorff distance between A and B: $d_H(A, B) = \max\{\max_{a\in A} \min_{b\in B} \|a - b\|, \max_{b\in B} \min_{a\in A} \|b - a\|\}$. Thus, the original Miml examples $(X_u, Y_u)(u = 1, 2, \ldots, N)$ have been transformed into multi-label examples $(z_u, Y_u)(u = 1, 2, \ldots, N)$.

Then, we can use hashing approaches to map the instances into hamming space, representing the whole bag into binary codes. Here we exploit the WTA as the hashing method, which is represented as f_{hash}, and then we can obtain the hashed multi-label instances:

$$B_u = f_{hash}(z_u) \;\; u = 1, 2, \ldots, N \tag{3}$$

Then, given the data set, a multi-label learning function f_{MLL} can be learned, which can accomplish the desired Miml function. By using MlSvm and a two-layer Neural Network (NN) to implement f_{MLL}, we can get MimlHSvmB, MimlHNNB, separately. More details about MlSvm, NN can be found in [21, 38]. Specifically, the function can be written as:

$$f_{MIMLHB}(X_u) = f_{MLL}(B_u) = f_{MLL}\{f_{hash}(z_u)\} \tag{4}$$

3.2.2. *Instance-level hashing.* In this scheme, we first hash each instance into binary codes; then, use the traditional methods to resolve the transformed multi-instance multi-label problem by replacing the dot-product kernel operator in the traditional methods. By doing this, we can effectively map the entire samples into hamming space, and speed up the process of learning tremendously. Here, we use MimlSvm, MimlNN, as the the traditional Miml methods, separately. We call the entire solution as MimlHSvmI, MimlHNNI. We get the hashed multi-instance multi-label samples with following function:

$$B_i = \{b_1^i, b_2^i, \ldots, b_{n_i}^i\} = f_{hash}(\{x_1^i, x_2^i, \ldots, x_{n_i}^i\}) \tag{5}$$

where $i = 1, 2, \ldots, N$. Then, the entire process can be written as:

$$f_{MIMLHI}(X_i) = f_{MIML}(f_{hash}(\{x_1^i, x_2^i, \ldots, x_{n_i}^i\})) \tag{6}$$

These schemes of hashing are shown in Figure 2.

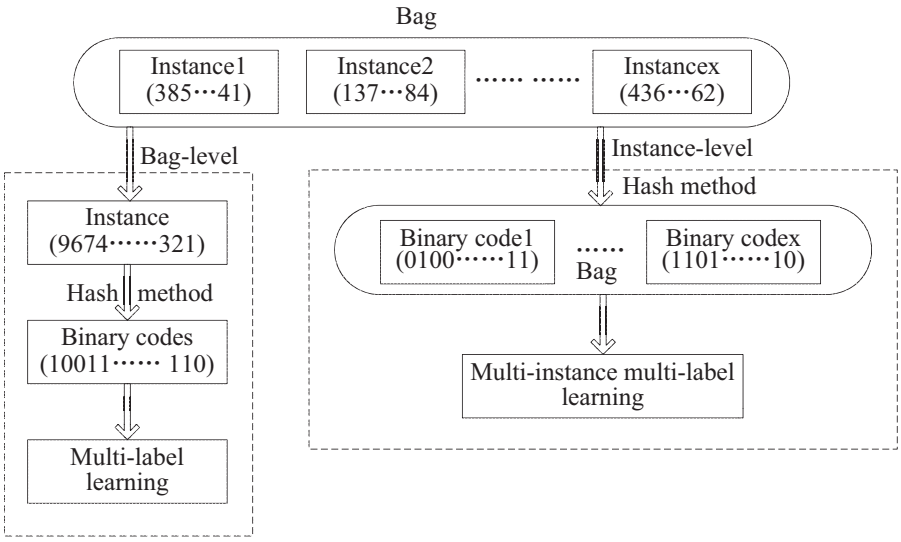


FIGURE 2. Two perspectives of hashing

3.3. **The details of hashing.** In this paper, we encode the original features as a high-dimensional sparse binary descriptor using a Winner-Take-All Hash [33]. WTA is well suited as a basis for locality-sensitive hashing, where the deterministic functions are non-linear and produce sparse descriptors. It has been shown to yield significant improvements on VOC 2010 using simple linear classifiers, which can be trained quickly. Besides, it is intuitive that the information stored in a WTA hash allows one to reconstruct a partial ordering of the coefficients in the hashed vector.

We use a subfamily of the hash functions derived in [33], where each WTA hash is defined by a sequence of N permutations of the elements in the original feature space. Each permutation consists of a list of indices of the vector; thus, we only need to retain the first K indices for each of the N permutations to implement a WTA hash function. For each (N*K)-length descriptor, it comprises N spans of length K. The term "winner take all" means that, each span consists of all zeros except the k^{th} entry, which is set to one to encode the index of the maximum value in the first K entries of the permutation. Each descriptor is compactly represented in $N * \lceil \log_2 K \rceil$ bits. Comparison is the only operation involved in the entire process, so the hashing scheme can be implemented completely with integer arithmetic. Moreover, it can be efficiently coded without branching and branch prediction penalties. The algorithm is summarized in **Algorithm 1**.

For instance, suppose that $N = 2$ and $K = 4$, the original vector is $[10, 5, 2, 6, 12, 3]$, and the permutations $[1, 4, 2, 5, 0, 3]$ and $[4, 5, 2, 0, 1, 3]$, the resulting WTA descriptor is [0001]: the first K indices of each permutation, $[1, 4, 2, 5]$ and $[4, 5, 2, 0]$, selecting $[5, 12, 2, 3]$ and $[12, 3, 2, 10]$, whose maximum values have the indices 1 and 0 for leftmost in the binary vector.

Each WTA hash function defines an ordinal embedding and a related rank-correlation similarity measure, which offers a degree of invariance with respect to perturbations in numeric values [33]. If the above vector transformed into $[22, 12, 6, 14, 26, 6]$ or $[11, 4, 3, 7, 13, 2]$, where the first is a scaled and offset version of the original vector while the last has each element perturbed, which results in the same or a different ranking of the elements but the same maximum of the first K elements, the results in the end are the same as the original code.

Algorithm 1 WTA-Hash

Input: A set of m permutations Θ; selected size K; input vector X.
Output: Sparse vector of codes C_X.

```
for each permutation θ_i in Θ do
    Permute elements of X according to θ_i to get X';
    Initialize i_th sparse code c_{x_i} to 0;
    set c_{x_i} to the index of the maximum value in X'(1...K):
    for j = 0 to K − 1 do
        if X'(j) > X'(c_{x_i}) then
            c_{x_i} = j
        end if
    end for
end for
```

$C_X = [c_{x_0}, c_{x_1}, \ldots, c_{x_{m-1}}]$, C contains m codes, each taking a value between 0 and $K-1$

4. **Experiments.** In this section, we demonstrate the effectiveness and efficiency of our proposed MimlH methods by experiments on two publicly available data sets: "miml-image-data" and "miml-text-data", which were derived from two applications of real-world Miml learning tasks [1, 2, 4], i.e., scene classification and text categorization problems. Note that the proposed framework can also work on other Miml tasks.

4.1. **Experimental setup.** The "miml-image-data" was derived from scene classification which was studied by Zhou and Zhang [1] in their investigation of the Miml framework. The data set is made up of 2,000 natural scene images collected from the COREL image collection and the Internet, belonging to the classes desert, mountains, sea, sunset, and trees, which are manually assigned to each image. In the data set, more than 22% of the data set belongs to more than one class, where the average number of labels per image is 1.24 ± 0.44. By using the SBN image bag generator [19], each image is represented as a bag of nine instances, where each instance is a 15-dimensional vector, corresponding to an image patch.

Moreover, we have also tested MimlH on text categorization problems. Specifically, the widely studied Reuters-21578 collection [34] is used in experiment, where the seven most frequent categories are considered. After removing documents whose label sets or main texts are empty and randomly removing documents with only one label, a text categorization data set containing 2,000 documents is obtained. Documents belonging to more than one class comprise over 15% of the data set and the average number of labels per document is 1.15 ± 0.37. Each document is represented as a bag of instances using the sliding window techniques [8], where each instance corresponds to a text segment enclosed in one sliding window of size 50 (overlapped with 25 words). "Function words" on the Smart stop-list [35] are removed from the vocabulary and the remaining words are stemmed. Instances in the bags adopt the "Bag-of-Words" representation based on term frequency [34, 36]. For the sake of effectiveness, dimensionality reduction is performed by retaining the top 2% words with the highest document frequency [37]. Eventually, each instance is represented as a 243-dimensional feature vector.

The characteristics of the above data sets are summarized in Table 1.

MimlH is compared with the original non-hash Miml methods on both accuracy and efficiency. We implement the bag-level solutions MimlHSvmB, MimlHNNB and instance-level solutions MimlHSvmI, MimlHNNI, which are represented in two versions including label information and without label information for each algorithms, compared with the

TABLE 1. Characteristics of the data sets

Data set	Number of examples	Number of classes	Number of features	Instances per bag			Labels per example (k)		
				min	max	mean±std	k=1	k=2	k≥3
Scene	2,000	5	15	9	9	9.00±0.00	1,543	442	15
Reuters	2,000	7	243	2	26	3.56±2.71	1,701	290	9

original non-hash methods, MIMLSVM, MIMLNN, separately, all of which are set to take the best parameters as reported in the primitive papers [1, 4]. Concretely, the Gaussian kernel with $\gamma = 0.2^2$ is used to implement MIMLSVM and the parameter k is set to be 20% of the number of training images, and the ratio and λ are set to be 0.4 and 0.5, respectively in MIMLNN.

Multi-instance multi-label learning algorithms make multi-label predictions, which can be evaluated according to five popular multi-label metrics, i.e., hamming loss, one-error, coverage, ranking loss and average precision. We also report the learning time of each method. Therefore, the performance of each compared algorithm can be evaluated according to the above six metrics. As for average precision, the bigger the value is, the better the performance is. While for the other five metrics, the smaller the value is, the better the performance is.

Use the same denotation as before, given test set $T = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_t, Y_t)\}$, the six criteria are defined as below. Here, $h(X_i)$ returns a set of proper labels of X_i, $h(X_i, y)$ returns a real-value indicating the confidence for y to be a proper label of X_i, and $rank^h(X_i, y)$ returns the rank of y derived from $h(X_i, y)$. The detailed definitions of these metrics are described as follows.

- The hamming loss evaluates how many times an object-label pair is misclassified, i.e., a proper label is missed or a wrong label is predicted, which can be defined as:

$$hamming - loss_T(h) = \frac{1}{t}\sum_{i=1}^{t}\frac{1}{|\mathcal{Y}|}|h(X_i) \triangle Y_i| \tag{7}$$

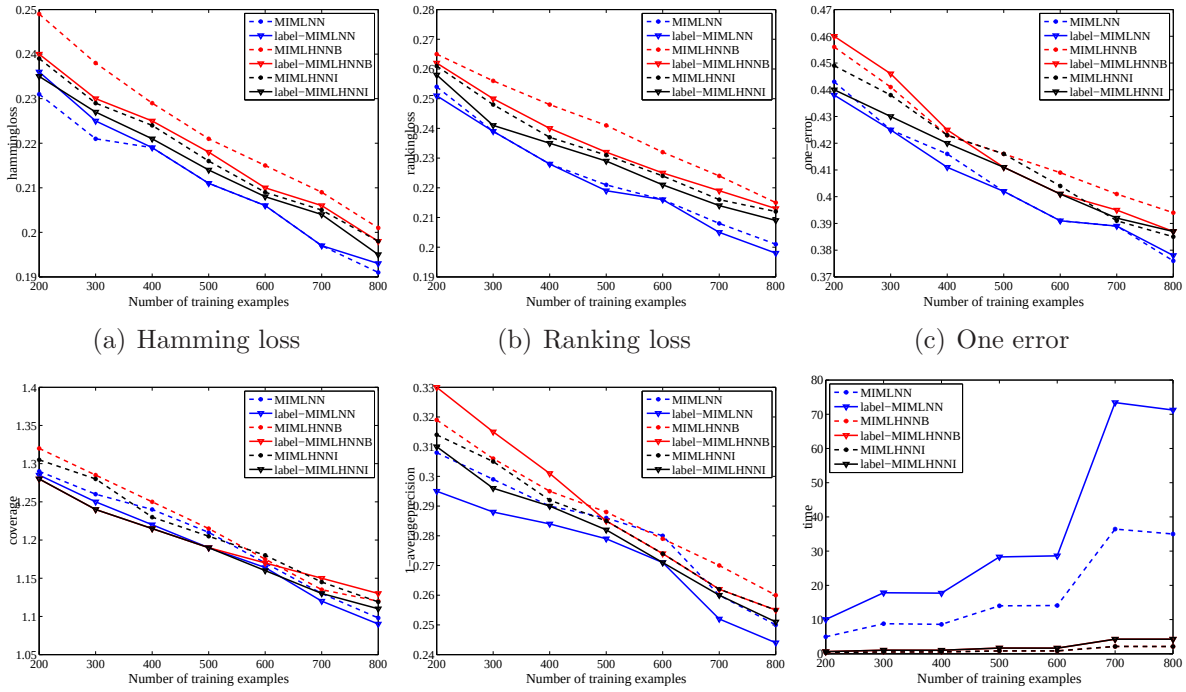
  where $\triangle$ stands for the symmetric difference between two sets. The performance is perfect when $hamming - loss_T(h) = 0$; the smaller the value of $hamming - loss_T(h)$ is, the better the performance of h is.
- The one-error evaluates how many times the top-ranked label is not a proper label of the object, which can be defined as:

$$one - error_T(h) = \frac{1}{t}\sum_{i=1}^{t}\|[\arg\max_{y\in\mathcal{Y}} h(X_i, y) \notin Y_i]\| \tag{8}$$

  The performance is perfect when $one - error_T(h) = 0$; the smaller the value of $one - error_T(h)$ is, the better the performance of h is.
- The coverage evaluates how far it is needed, on the coverage, to go down the list of labels in order to cover all the proper labels of the object, which can be defined as:

$$coverage_T(h) = \frac{1}{t}\sum_{i=1}^{t}\max_{y\in Y_i} rank^h(X_i, y) - 1 \tag{9}$$

  It is loosely associated with precision at the level of perfect recall. The smaller the value of $coverage_T(h)$ is, the better the performance of h is.



(a) Hamming loss (b) Ranking loss (c) One error

(d) Coverage (e) 1-average precision (f) Time

FIGURE 3. The performance of each method implemented by NN (on the scene classification data) changes as the number of training examples increases. In each subfigure, the lower the curve is, the better the performance of the algorithm is.

- The ranking-loss evaluates the average fraction of label pairs that are disordered for the object, which can be defined as:

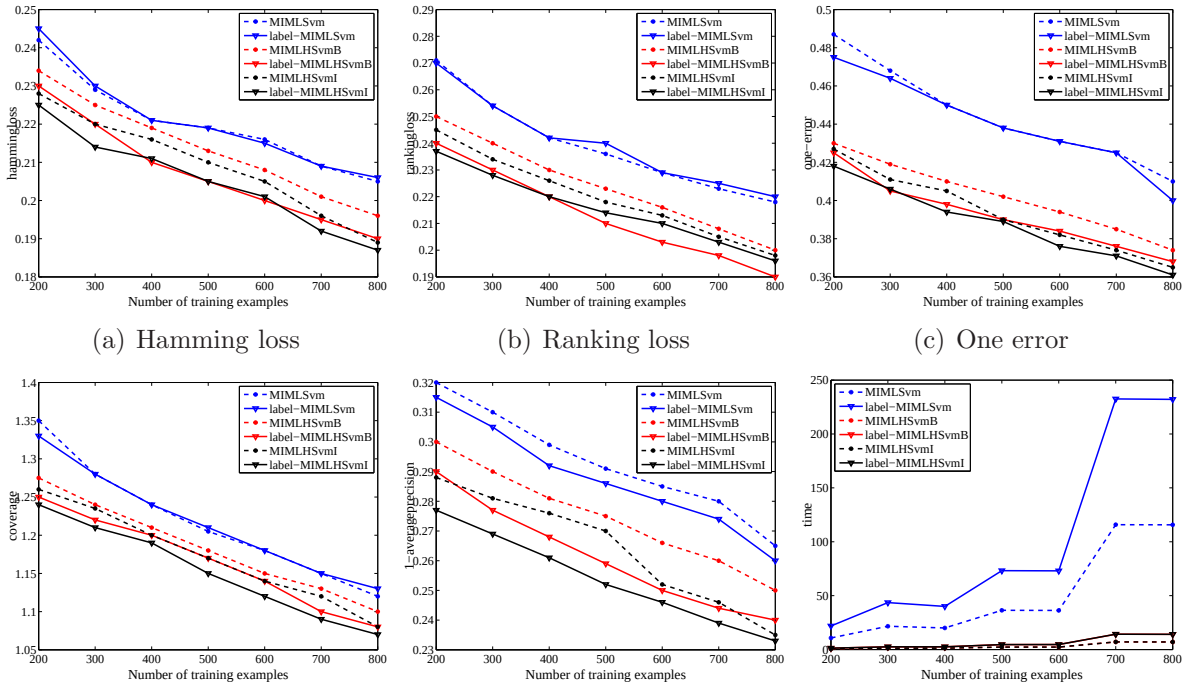
$$ranking - loss_T(h) = \frac{1}{t}\sum_{i=1}^{t}\frac{1}{|Y_i||\overline{Y_i}|}\left(|\{(y_1, y_2)\ |h(X_i, y_1) \leq h(X_i, y_2), (y_1, y_2) \in Y_i \times \overline{Y_i}\}|\right) \tag{10}$$

  The performance is perfect when $ranking - loss_T(h) = 0$. The smaller the value of $ranking - loss_T(h)$ is, the better the performance of h is.
- The average-precision evaluates the average fraction of labels ranked above a particular label $y \in Y_i$, which can be defined as:

$$average - precision_T(h) = \frac{1}{t}\sum_{i=1}^{t}\frac{1}{|Y_i|}\sum_{y\in Y_i}\frac{|\{y' \mid rank^h(X_i, y') \leq rank^h(X_i, y), y' \in Y_i\}|}{rank^h(X_i, y)} \tag{11}$$

  The performance is perfect when $average - precision_T(h) = 1$. The larger the value of $average - precision_T(h)$ is, the better the performance of h is.
- The time of each method is recorded to compare the efficiency of these approaches.

4.2. **Evaluation and discussion.** For both scene and Reuters data, we first randomly choose 1000 samples from the original data set as the test set. The remaining examples are then used to form the potential training set, where training set is created by randomly picking up N examples from the potential training set. In this paper, N ranges from 200 to 800 with an interval of 100. For each value of N, twenty different training sets are



(a) Hamming loss (b) Ranking loss (c) One error

(d) Coverage (e) 1-average precision (f) Time

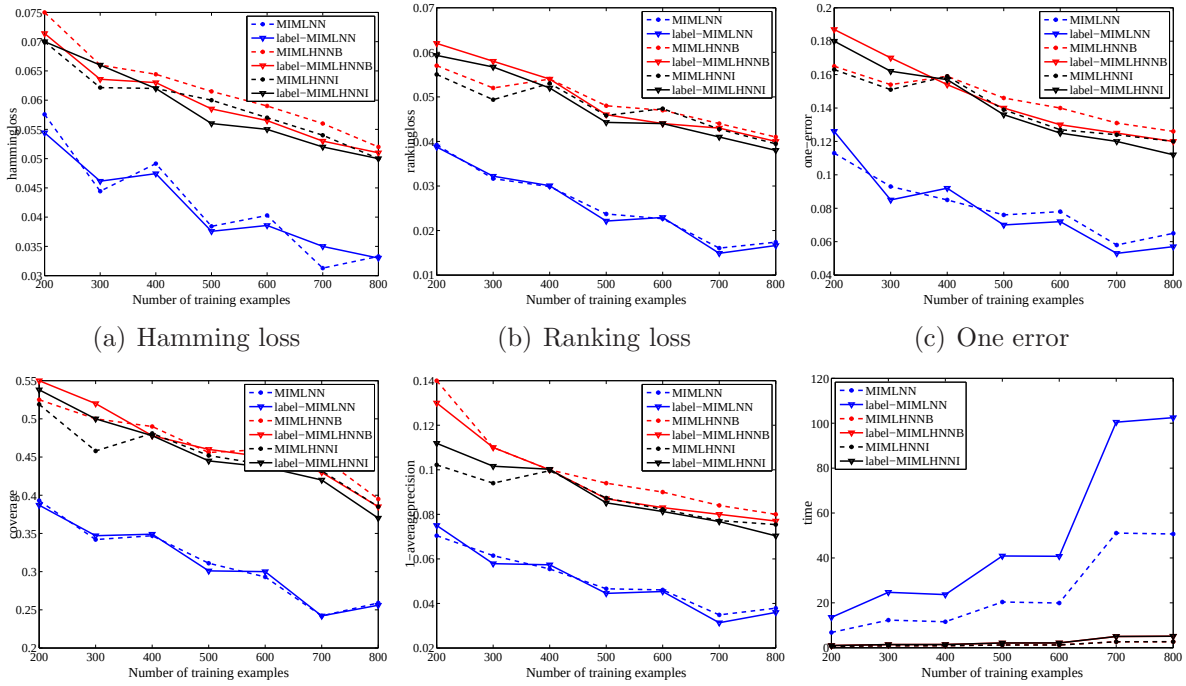
FIGURE 4. The performance of each method implemented by SVM (on the scene classification data) changes as the number of training examples increases. In each subfigure, the lower the curve is, the better the performance of the algorithm is.

created by repeating the pickup procedure. We report the average test performance of each algorithm trained on the twenty training sets.

Figure 3 and Figure 4 illustrate the performance of each method implemented by NN and SVM on the scene classification data. For each algorithm, when the training set size is fixed, the average values of the twenty independent runs are depicted. It is obvious that as the number of the training set increases, the performance turns better. Note that, for the sake of convenience, we plot the 1-average precision instead of average precision. Thus, in each subfigure, the lower the curve is, the better the performance of the algorithm is. Accordingly, Figure 5 and Figure 6 report the experimental results on the Reuters categorization data in the same way.

From these figures, we can see that the proposed framework performs in a similar accuracy and a much better speed compared with the previous works:

- First, each algorithm with label information performs better or similarly to the corresponding methods without label information on accuracy while with a nearly double time. In detail, on both data set, the algorithm with label information performs better than the original method with the bag-level hash and instance-level hash approach, and performs similarly to the corresponding non-hashing methods. We think it is due to that the label information is similar with the hash code and supplementing effective information for the hashing methods, while it is dissimilar with the original feature and ends in little improvement in performance.
- Second, hashing methods perform better than corresponding non-hash methods applying MIMLSVM while performing worse or similarly employing MIMLNN. It is



(a) Hamming loss (b) Ranking loss (c) One error

(d) Coverage (e) 1-average precision (f) Time

FIGURE 5. The performance of each method implemented by NN (on the Reuters data) changes as the number of training examples increases. In each subfigure, the lower the curve is, the better the performance of the algorithm is.

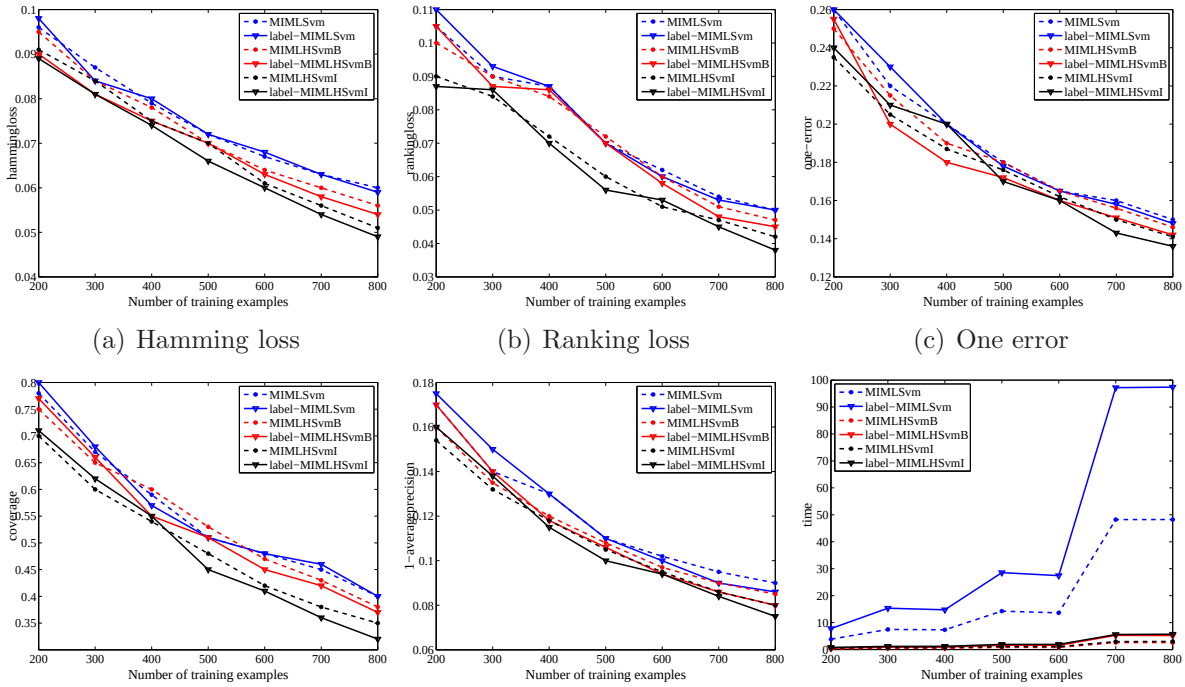
reasonable to have a worse accuracy with hashing method since it is an approximate solution to the original problem.

- Third, instance-level hashing performs better than the bag-level hashing. Instance-level hashing exploits more meticulous perspective than the bag-level one, resulting in better performance.
- Fourth, for all the methods, hashing methods speed up in training phase tremendously.

In conclusion, the experimental results show that the proposed framework gets similar accuracy to those of previous methods; however, it is much better than those methods on efficiency. Thus it performs better than those previous works on accuracy and efficiency in a balanced way.

5. **Conclusions.** In this paper, we propose the Multi-instance Multi-label Hashing (MIMLH) to tackle both accuracy and scalability issues of MIML. MIMLH exploits the hashing approach in two perspectives – bag-level hashing and instance-level hashing, which replaces the dot-product kernel operator in the previous methods, effectively maps the entire samples into hamming space, and speeds up the process of learning tremendously. Moreover, we also take the label information into account to enhance our framework. We evaluate the proposed approach on two popular data sets of MIML – scene classification and text categorization. The experimental results show that the proposed method outperforms those previous methods on accuracy and efficiency in a balanced way.

Note that MIMLH is a general learning framework, it can work on all MIML tasks, e.g., text categorization, and image annotation. In addition, it is promising to achieve



(a) Hamming loss (b) Ranking loss (c) One error

(d) Coverage (e) 1-average precision (f) Time

FIGURE 6. The performance of each method implemented by SVM (on the Reuters data) changes as the number of training examples increases. In each subfigure, the lower the curve is, the better the performance of the algorithm is.

even higher effectiveness and efficiency if more powerful hashing approaches and MIML methods can be employed.

Acknowledgment. This work is partially supported by National Natural Science Foundation of China (61173068), Program for New Century Excellent Talents in University of the Ministry of Education, the Key Science Technology Project of Shandong Province (2014GGD01063), the Independent Innovation Foundation of Shandong Province (2014CG ZH1106) and the Shandong Provincial Natural Science Foundation (ZR2014FM020).

REFERENCES

[1] Z. Zhou and M. Zhang, Multi-instance multi-label learning with application to scene classification, *Proc. of the 19th Annual Conference on Neural Information Processing Systems*, pp.1609-1616, 2006.
[2] M. Zhang and Z. Zhou, M3MIML: A maximum margin method for multi-instance multi-label learning, *Proc. of the 8th IEEE International Conference on Data Mining*, pp.688-697, 2008.
[3] C. Nguyen, D. Zhan and Z. Zhou, Multi-modal image annotation with multi-instance multi-label LDA, *Proc. of the 23rd International Joint Conference on Artificial Intelligence*, pp.1558-1564, 2013.
[4] Z. Zhou, M. Zhang, S. Huang and Y. Li, Multi-instance multi-label learning, *Journal of Artificial Intelligence*, vol.176, no.1, pp.2291-2320, 2012.
[5] T. G. Dietterich, R. H. Lathrop and T. Lozano-Pérez, Solving the multiple instance problem with axis-parallel rectangles, *Journal of Artificial Intelligence*, vol.89, no.1-2, pp.31-71, 1997.
[6] A. McCallum, Multi-label text classification with a mixture model trained by EM, *Proc. of AAAI'99 Workshop on Text Learning*, pp.1-7, 1999.
[7] R. E. Schapire and Y. Singer, BoosTexter: A boosting-based system for text categorization, *IEE-Part D, Control Theory and Applications*, vol.39, no.2-3, pp.135-168, 2000.
[8] S. Andrews, I. Tsochantaridis and T. Hofmann, Support vector machines for multiple-instance learning, *Proc. of the 15th Neural Information Processing Systems*, pp.561-568, 2002.

[9] O. Maron and T. Lozano-Pérez, A framework for multiple-instance learning, *Proc. of the 11th Neural Information Processing Systems*, pp.570-576, 1998.
[10] J. Wang and J.-D. Zucker, Solving multiple-instance problem: A lazy learning approach, *Proc. of the 17th International Conference on Machine Learning*, pp.1119-1126, 2000.
[11] X. Xu and E. Frank, Logistic regression and boosting for labeled bags of instances, *Journal of Knowledge Discovery and Data Mining*, vol.3056, pp.272-281, 2004.
[12] Q. Zhang and S. A. Goldman, EM-DD: An improved multiple-instance learning technique, *Proc. of the 14th Neural Information Processing Systems*, pp.1073-1080, 2001.
[13] K. Brinker, J. Fürnkranz and E. Hüllermeier, A unified model for multilabel classification and ranking, *Proc. of the 17th European Conference on Artificial Intelligence*, Riva Del Garda, Italy, pp.489-493, 2006.
[14] A. Elisseeff and J. Weston, Kernel methods for multi-labelled classification and categorical regression problems, *Proc. of the 15th Annual Conference on Neural Information Processing Systems*, Vancouver, B.C., Canada, pp.681-687, 2002.
[15] H. Kazawa, T. Izumitani, H. Taira and E. Maeda, Maximal margin labeling for multi-topic text categorization, *Proc. of the 17th Annual Conference on Neural Information Processing Systems*, Vancouver, B.C., Canada, pp.649-656, 2004.
[16] S. Zhu, X. Ji, W. Xu and Y. Gong, Multi-labelled classification using maximum entropy method, *Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Salvador, Brazil, pp.274-281, 2005.
[17] E. K. Boukas, Z. Liu and P. Shi, MILES: Multiple-instance learning via embedded instance selection, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.28, no.12, pp.1931-1947, 2006.
[18] Y. Chen and J. Z. Wang, Image categorization by learning and reasoning with regions, *Machine Learning Research*, vol.5, pp.913-939, 2004.
[19] O. Maron and A. L. Ratan, Multiple-instance learning for natural scene classification, *Proc. of the 15th International Conference on Machine Learning*, Vancouver, B.C., Canada, pp.341-349, 1998.
[20] M. Zhang and Z. Zhou, Multi-instance clustering with applications to multi-instance prediction, *Applied Intelligence*, vol.31, no.1, pp.47-68, 2009.
[21] M. R. Boutell, J. Luo, X. Shen and C. M. Brown, Learning multi-label scene classification, *Pattern Recognition*, vol.37, no.9, pp.1757-1771, 2004.
[22] S. Gao, W. Wu, C. H. Lee and T. S. Chua, A MFoM learning approach to robust multiclass multi-label text categorization, *Proc. of the 21st International Conference on Machine Learning*, Banff, Canada, pp.681-687, 2004.
[23] N. Ueda and K. Saito, Parametric mixture models for multi-labeled text, *Proc. of the 15th Annual Conference on Neural Information Processing Systems*, Vancouver, B.C., Canada, pp.721-728, 2002.
[24] J. Foulds and E. Frank, A review of multi-instance learning assumptions, *Knowledge Engineering Review*, vol.25, no.1, pp.1-25, 2010.
[25] C. A. Tawiah and V. S. Sheng, A study on multi-label classification, *Proc. of the 13th Industrial Conference on Data Mining. Applications and Theoretical Aspects*, Dallas, Texas, USA, pp.137-150, 2013.
[26] S. Yang, H. Zha and B. Hu, Dirichlet-bernoulli alignment: A generative model for multi-class multi-label multi-instance corpora, *Proc. of the 22nd Neural Information Processing Systems*, Vancouver, B.C., Canada, pp.2143-2150, 2009.
[27] F. Briggs, X. Z. Fern and R. Raich, Rank-loss support instance machines for MIML instance annotation, *Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Beijing, China, pp.534-542, 2012.
[28] M. Datar, N. Immorlica, P. Indyk and V. S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, *Proc. of the 20th Annual Symposium on Computational Geometry*, New York, USA, pp.253-262, 2004.
[29] Y. Weiss, A. Torralba and R. Fergus, Spectral hashing, *Proc. of the 21st Neural Information Processing Systems*, Vancouver, B.C., Canada, pp.1753-1760, 2008.
[30] D. Zhang, J. Wang, D. Cai and J. Lu, Self-taught hashing for fast similarity search, *Proc. of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, Switzerland, pp.18-25, 2010.
[31] A. Joly and O. Buisson, A posteriori multi-probe locality sensitive hashing, *Proc. of the 16th ACM International Conference on Multimedia*, Vancouver, B.C., Canada, pp.209-218, 2008.

[32] A. Torralba, R. Fergus and Y. Weiss, Small codes and large image databases for recognition, *Proc. of the 21st IEEE Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, USA, pp.1-8, 2008.
[33] J. Yagnik, D. Strelow, D. A. Ross and R.-S. Lin, The power of comparative reasoning, *Proc. of the 13th IEEE International Conference on Computer Vision*, Barcelona, Spain, pp.2431-2438, 2011.
[34] F. Sebastiani, Machine learning in automated text categorization, *ACM Computing Surveys (CSUR)*, vol.34, pp.1-47, 2002.
[35] G. Salton, *Automatic Text Processing: The Transformation Analysis and Retrieval of Information by Computer*, Addison-Wesley, 1989.
[36] S. Dumais, J. Platt, D. Heckerman and M. Sahami, Inductive learning algorithms and representations for text categorization, *Proc. of the 7th International Conference on Information and Knowledge Management*, pp.148-155, 1998.
[37] Y. Yang and J. O. Pedersen, A comparative study on feature selection in text categorization, *Proc. of the 14th International Conference on Machine Learning*, pp.412-420, 1997.
[38] M. Zhang and Z. Zhou, Multi-label learning by instance differentiation, *Proc. of the 22nd AAAI Conference on Artificial Intelligence*, pp.669-674, 2007.