A Triple-Random Ensemble Classification Method for Mining Multi-label Data

Gulisong Nasierding [1,2]
[1] Dept. of Computer Science and Technology
Xinjiang Normal University
102 Xin Yi Rd, Urumqi, P.R. China
gulnas9@gmail.com

Abbas Z. Kouzani
[2] School of Engineering
Deakin University
Geelong, VIC 3217, Australia
kouzani@deakin.edu.au

Grigorios Tsoumakas
Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki 54124, Greece
greg@csd.auth.gr

***Abstract*—This paper presents a triple-random ensemble learning method for handling multi-label classification problems. The proposed method integrates and develops the concepts of random subspace, bagging and random k-labelsets ensemble learning methods to form an approach to classify multi-label data. It applies the random subspace method to feature space, label space as well as instance space. The devised subsets selection procedure is executed iteratively. Each multi-label classifier is trained using the randomly selected subsets. At the end of the iteration, optimal parameters are selected and the ensemble MLC classifiers are constructed. The proposed method is implemented and its performance compared against that of popular multi-label classification methods. The experimental results reveal that the proposed method outperforms the examined counterparts in most occasions when tested on six small to larger multi-label datasets from different domains. This demonstrates that the developed method possesses general applicability for various multi-label classification problems.**

Keywords-triple-random ensemble; multi-label classification; subspace method; RAkEL; bagging

I. INTRODUCTION

In multi-label classification (MLC), each example can be associated with multiple labels, whereas in single-label classification, each example can be associated with a single label [1-5]. $L = \{l_j : j = 1...M\}$ is used to represents the finite set of labels, and $D = \{(\vec{x}_i, Y_i), i = 1,..., N\}$ is used to describe a set of multi-label training examples, where $\vec{x}_i$ denotes a feature vector, and $Y_i \subseteq L$ denotes a set of labels of the *i*-th example in *D*. Multi-label classification problems can be found in various domains including text document classification [6-9], bioinformatics data classification [10, 11], music categorization [12], scene image classification [1, 3, 13], image and video annotations [14-17].

Multi-label classification methods can be categorized into two groups [4, 5]: (i) problem transformation (PT) and (ii) algorithm adaptation (AA). The PT based MLC methods are algorithm independent. They transform a multi-label problem into one or more single-label classification or regression problems, then use single-label classifiers to tackle the learning problem. Various PT methods are introduced and summarized in [4, 5, 8, 9]. The AA based MLC methods extend a specific learning algorithm to adapt multi-label learning for handling multi-label problem directly [4, 18]. This paper introduces a novel PT based triple-random ensemble multi-label classification (TREMLC) framework to handle multi-label problems from various domains. The proposed TREMLC algorithm integrates the concepts of random subspace [19-21], bagging [22-25] and random k-label sets ensemble learning methods [9, 18] to form an approach to classify multi-label data. It applies the random subspace method to feature space, label space as well as instance space.

The TREMLC method inspires from the ensemble learning approaches [2, 3, 7, 9, 18-29]. Firstly, a number of variant random subspace methods [19-21] build ensemble classifiers by using a pseudo-random selection of a small number of feature dimensions. Secondly, the bootstrap sub-sampling bagging method [22] generates many sub training sets by replicating the original training set, which enables building ensemble of classifiers for a better classification accuracy [22, 23]. Thirdly, RAkEL randomly selects k-label subsets to build an ensemble learning method whose performance has been promising comparing to some popular counterparts [18]. Furthermore, the combination and extension of random subspace and bagging strategies can bring robustness to ensemble learning, such as attributes bagging [20], random feature subset selection [21], bootstrap-inspired techniques [23], random forests and its extension [24, 25], especially the methods developed for tackling multi-label classification problems [2, 3, 9, 18, 27-29]. Hence, the TREMLC method is formed by constructing an ensemble of multi-label classifiers based on the feature subsets, label subsets and instance sets produced by applying the random subspace method to the instance space, feature space and label space of the multi-label data.

II. RELATED WORK

A. Multi-label Learning Algorithms

Focusing on the PT based MLC methods, the binary relevance (BR) [5, 11] learns *M* binary classifiers, one for each different label in L. For the classification of a new instance, it outputs the union of the labels that are positively predicted by the classifiers. The label power set (LP) method [1, 4] considers each unique set of labels that exists

978-0-7695-4257-7/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDMW.2010.139

in a multi-label training set, as one of the classes in a new single-label classification task. Given a new instance, the single-label classifier outputs the most probable class. Due to the large number of classes produced by LP, the classes correspond to a few examples cause difficulties.

The random k-label sets (RAkEL) method [18] builds an ensemble of LP classifiers. Each classifier is trained using a different small random subset of the set of labels. In such a way, RAkEL is able to take label correlations into account. A ranking of the labels is produced and a threshold is used in forming the decision for classification of a new instance. The calibrated label ranking (CLR) method [13] learns a mapping from instances to rank over a finite number of predefined set of class labels. It separates the relevant labels from the irrelevant labels in each example.

The hierarchy of multi-label classifiers (HOMER) method [30] constructs a tree-shaped hierarchy of simple multi-label classifier, where each classifier handles a smaller set of labels. The divide-and-conquer strategies are adopted for a balanced example distribution in HOMER. Different approaches for distribution of labels into subsets are used for HOMER [30]. The pruned problem transformation (PPT) [8] and ensemble of pruned set (EPS) methods are reported for text document and other type of multi-label data classifications. EPS is an ensemble of the pruned set (PS) [9]. In order to avoid unnecessary and detrimental complexity and to ensure minimal information loss, PS prunes away infrequently occurring label sets. Then, the pruned sets are broken up to more frequently occurring subsets, and pruned instances are reintroduced into the data. EPS is the improvement of PS by applying the ensemble learning strategy to PS [8, 9].

As AA based methods, the multi-label k-nearest neighbour (ML-KNN) [31] and variants [32] extend the k-Nearest Neighbours (kNN) lazy learning algorithm using maximum a posterioris principle to determine the label set. Back-propagation for multi-label learning (BP-MLL) [33] is an adaptation of the back-propagation algorithm to multi-label learning problems by introducing a new error function. Multi-instance, multi-label boosting based ensemble learning framework was proposed by developing the MIMLBOOST and MIMLSVM algorithms [3]. BoosTexter [2] is a robust ensemble learning method for multi-label text categorization. In which, Adaboost.MH and Adaboost.MR algorithms are used by applying AdaBoost [26] on weak classifiers. Although ML-KNN is considered as an AA based MLC approach, it actually employs one of the PT schemes, then uses the kNN algorithm for each label independently finding the k nearest examples for the test instance [31].

Several binary classifiers for single-label classification can be employed as the baseline algorithm for multi-label classification [5, 18]. These baseline algorithms include the decision trees [2, 3, 7, 34], support vector machines (SVM) [1, 10], neural networks [33], random field and probabilistic methods [35], and k nearest neighbour lazy learning [11, 31, 32].

B. Ensemble Learning

Bagging [22, 23], boosting [2, 26] and random forests [24, 25] are popular ensemble learning methods for classification problems. The performances of the ensemble learning methods are appealing compared to single classifiers [2, 3, 18-29]. The bagging method obtains bootstrap random sub-samples iteratively to train weak learning algorithm, by which ensemble classifiers are built at the end of iteration [22, 23]. The random subspace method [19-21] applies the base-level decision tree algorithm [36] on randomly selected features subset at each step of the tree construction. The attribute bagging method [20] was proposed for improving accuracy of ensembles by applying the bagging method to feature space. Furthermore, random feature subset selection strategies [21] and bootstrap-inspired techniques [23] also made great contribution to the improvement of ensemble learning performances. Breiman [24] combined the bagging and random subspaces to form random forests, and Panov et al. [25] developed a variant of random forests for achieving better ensemble classification performances. However, these methods are suitable for single-label classification problems.

The ensemble learning methods can bring robustness to multi-label classification. For instance, in order to reduce the information redundancy during the multi-label learning, a model-shared subspace boosting algorithm was developed [27], which automatically finds shared subspace models, where each model is learned from the random feature subspace and bootstrap data, and combines a number of base models through multiple labels. A method for exploiting correlation information that contained in different labels, and extracting shared common subspace among multiple labels is explored for multi-label classification [28]. EPS was developed by using pruned set ensemble learning strategy [8]. As random subspace methods optimize the performance of the classification in terms of general accuracy [19-21, 27, 28], and RAkEL algorithm achieves better performance compares to LP [18] by building an ensemble of LP classifiers, thus, a dual-random ensemble multi-label classification algorithm [29] was structured by taking advantage of the best part of the random subspace method and the RAkEL. Moreover, these methods are yet to satisfy the requirements for effective and accurate classification of multi-label data from various domains.

III. TRIPLE-RANDOM ENSEMBLE MULTI-LABEL CLASSIFICATION

The proposed triple-random ensemble multi-label classification (TREMLC) algorithm randomly selects feature subsets, label subsets and instance subsets to build ensemble of multi-label classifiers for dealing with multi-label problem effectively. It combines and extends the concepts of random subspace method (RSM) [19-21],

bagging [22, 23] and random k-labelset sampling ensemble learning methods [18], where RSM applies the random subspace selection strategy to feature space, RAkEL applies the random subset selection scheme to label space, and bagging applies the random sub-sampling method to instance space. Furthermore, since the random forest [24] and its variants [25] construct ensemble classifiers by applying the random subset selection mechanism to both feature space and instance space, TREMLC can be viewed as the extension of the ideas of the RSM, bagging and RAkEL, or integrating the concepts of random forests and RAkEL. That is, TREMLC applies the random subspace method to feature space, label space, as well as instance space. The pseudo-code of the TREMLC training process is described in Fig. 1.

Input: Set of training data *D* of size *N*, set of attributes *A* of size *F*, set of labels *L* with size *M*, size of feature subset S_f < F, size of label subset S_l < M, bag percentage *b,* number of models *m*
Output: Subsets R_i from projection of D_i within feature dimension F, label dimension G;
Constructed ensemble of LP classifiers h_i, (i=1…m) trained on R_i

```
FS ← {}
LS ← {}
for i =1 to m
 {
    Di ← random selection of N*b% instances from D;
    do {
        Fi ← random selection of Sf features from A
       } while (Fi not in FS);
    FS← FS union {F}
    do{
        Gi ← random selection of Sl labels from L
       } while (Gi not in LS);
    LS ← LS union {G}
     Ri ← projection of Di to the attribute and label
          dimensions F and G.
  Train an LP classifier hi based on Ri;
 }
```

Figure 1. Pseudo code of the TREMLC training process.

In Fig. 1, *FS* denotes feature subsets and S_f denotes the size of feature subset, *LS* denotes label subsets, and S_l denotes the size of label subsets. In each iteration, i.e. when the number of models *m* incrementally changing from 1 to a specified value, randomly select a certain percentage of instances from *D,* within a selected dataset, randomly select a feature subset with size S_f and a label subset with size S_l, These random subset selections are without replacement. By the end of the iteration, a set of ensemble multi-label classifiers are constructed by training the label power set classifier on the randomly selected subsets. As a result, sufficient trained multi-label ensemble classifiers are obtained.

Next, a set of optimal parameters can be determined based on the best training performance of TREMLC. The optimal parameters include the best number of models, best sizes of feature subset, label subset, instance set, and threshold. Then, the selected parameters can be used for testing of TREMLC. The LP algorithm is employed as the base learner and the decision tree [36] is recommended as the base level classifier for LP in TREMLC. TREMLC is a three layer structured triple random subsets selection ensemble learning algorithm for mining multi-label data. It employs the binary classifier decision tree in the bottom layer, builds ensembles of LP classifiers in the second layer, and constructs the TREMLC framework in the third layer by combining the ensemble of multi-label LP classifiers.

When a new instance $\vec{x}$ arrives for classification, each classifier h_i provides binary predictions h_i ($\vec{x}$, l_j) for each label l_j in the corresponding training subset with the size of feature dimension S_f <*F* and label dimension S_l < *L*. By default, the size of label subset is set to be 3, the size of feature subset is set to be 70% of the original feature set, the size of instance subset is set to be 70% of entire instances, and the threshold is set to be 0.5. The threshold is used at final decision making on labels predictions when applying majority vote, and determining the confidence of the predictions. Note that one can find most suitable threshold by fine tuning it during the training of TREMLC along with finding optimal parameters for a specific multi-label classification task. The pseudo-code for the classification of new instances using TREMLC is given in Fig. 2.

Input: Set of labels L with size M, number of models m, k-sized label subsets G_i, attribute subsets F_i with specified size, and built LP classifiers h_i, new instance $\vec{x}$
Output: Multi-label classification vector Result

```
for ( i = 1 to m)
 {
    x' ← projection of x in dimensions of Fi and Gi;
    p = hi(x');
   for( j = 1 to L)
    {
     SumVote_label index of j = SumVote_label index of j + Vote(p);
     LengthVote_label index of j++;
    }
 }
for (int j = 1 to M)
 {
   Confj ← SumVotej / LengthVotej;
    if (Confj > threshold)
      {
       Resultj ←1;
      }
  else Resultj ← 0;
 }
```

Figure 2. Pseudo code for the TREMLC testing process.

IV. EXPERIMENTAL SETUP

A. Datasets

The TREMLC algorithm and several counterparts are tested on six multi-label datasets including diagnostic text report dataset *medical* [37], multiple topic related email messages dataset *enron* [38], biological dataset *yeast* [10], music categorical dataset *emotions* [12], image dataset *scene* [1], and multimedia video dataset *mediamill* [39].

The *medical* dataset was constructed from the available data in Computational Medicine Center's 2007 Medical Natural Language Processing Challenge [37]. This dataset contains 978 clinical free text reports, and each diagnostic report is related to one or more disease code from the 45 classes [9, 18, 37].

The *enron* dataset is a subset of the Enron email Corpus which contains 1702 email messages that are associated with a set of 53 topics, such as humor, company strategy and legal advice. The Enron dataset is developed by the UC Berkeley Enron Email Analysis Project [9, 18, 38].

The *yeast* dataset contains 2417 gene examples, and each of which is related to a set of 14 functional gene classes from the comprehensive Yeast Genome Database of the Munich Information Center for protein Sequences. Each gene is expressed with 103 numeric features [9, 10, 18].

The *emotions* dataset contains a set of 593 songs with 6 clusters of music emotions, which is constructed based on the Tellegen-Watson-Clark model [12, 32].

The *scene* image dataset contains 2407 images annotated with up to 6 concepts such as beach, mountain and field. Each image is described with 294 visual numeric features and these features are represented with spatial colour moments in Luv colour space. Each instance in the train and test datasets is labelled with possible 6 object classes as mentioned above [1, 13].

The *mediamill* dataset is based on the mediamill challenge data set [16, 18, 39]. It contains pre-computed low-level multimedia features from 85 hours of international broadcast news video of the TRECVID 2005/2006. This dataset contains Arabic, Chinese, and US news broadcasts that were recorded during November 2004, and the contents are annotated with multiple labels. The component used for the evaluation of MLC algorithms are based on still image data from the video shot key frames extracted. The annotation of the *mediamill* data was extended to current 101 concepts from a manual annotation of 39 labels.

The described datasets are widely used as benchmark datasets for evaluation of MLC algorithms. They can be obtained from the knowledge discovery and machine learning website [4]. Table I shows general characteristics of these datasets, including name, number of instances, number of attributes or features, and number of labels for each dataset, types of attributes, and the domains these datasets belonging to. Note that, 'num' in the table refers to the numerical attribute of dataset, and 'nom' refers to the nominal attribute dataset.

TABLE I. CHARACTERISTICS OF THE DATASETS USED

Name	Domain	Instances	Attributes	Labels
medical	text	978	1449 nom	45
enron	text	1702	1001nom.	53
yeast	biology	2417	103 num	14
emotions	music	593	72 num	6
scene	image	2407	294 num.	6
mediamill	video	43907	120 num.	101

B. MLC Evaluation Methodology

The evaluation measures for multi-label classifiers are different from those of single-label classifiers [1, 5, 31]. They can be divided into example based, label-based measures, and ranking based measures [5].

- *Example-based evaluation measures* are based on the average differences of the actual and predicted sets of labels over all examples of the evaluation dataset. The Hamming-loss refers to the average binary classification error. Suppose the multi-label evaluation dataset D contains multi-label examples (x_i, Y_i), i=1, 2,…,N, where $Y_i \subseteq L$ denotes a set of true labels, L= $\{l_j$: j=1…M$\}$ denotes the set of all the labels, and x_i denotes a new instance. Hence, Hamming-loss is:

$$\text{H-loss}=\frac{1}{N}\sum_{i=1}^{N}\frac{|Y_i \Delta Z_i|}{M} \quad (1)$$

where $Z_i = h(x_i)$ denotes a set of labels predicted by a multi-label classifier h for an example $x_{i.}$, and Δ stands for the symmetric difference of two sets [5]. The smaller value of Hamming-loss is indicative of better performances of the classification.

- *Label-Based Evaluation Measures*: A label based F1-measure refers to the harmonic mean between precision and recall, where the recall refers to the percentage of relevant labels that are predicted and precision refers to the percentage of predicted labels that are relevant. F1-measure is widely used for single-label classification evaluation, which is applicable for evaluating multi-label classification by using Micro-averaging and Macro averaging. F1 can be defined as:

$$\text{F1-measure}=\frac{2*tp}{2*tp+fp+fn} \quad (2)$$

where $tp_l, fp_l.tn_l, fn_l$ denote the number of true positives, false positives, true negatives and false negatives measures for a label l prediction [5]. The micro-averaged version of binary evaluation measure (B), which used in this paper, can be calculated as:

$$B_{micro}=B\left(\sum_{l=1}^{M}tp_l,\sum_{l=1}^{M}fp_l,\sum_{l=1}^{|M}tn_l,\sum_{l=1}^{M}fn_l\right) \quad (3)$$

The larger value of the micro F1-measure is indicative of better performances of the classification.

- *Ranking-based Evaluation Measures* perform ranking of label predictions. The most relevant label is ranked to receive the highest score, while the most irrelevant one is ranked to receive the lowest score. There are four ranking-based metrics for measuring the label ranking, i.e. one-error, coverage, ranking-loss and average precision [5]. The average precision evaluates the average fraction of labels ranked above a particular label $l \in Y_i$:

$$avg.\Pr ec. = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|Y_i|}\sum_{l\in Y_i}\frac{|l' \in Y_i : r_i(l') \le r_i(l)\}|}{r_i(l)} \quad (4)$$

The larger value of the average precision is indicative of the better performance of the classification.

C. Experimental Setting

In order to empirical study the TREML method, several popular MLC algorithms are chosen from the open source MULAN library [4], which is built on top of the open source Weka library [40]. The default parameters are set for the examined MLC algorithms as indicated in the literature. Such as, ML-kNN [31] is run with 10 nearest neighbours and a smoothing factor equal to 1. RAKEL uses label power set [1, 4] as multi-label learner base, and the size of label subset is set to k=3, number of models (number of iterations) is twice the size of the label set of the multi-label dataset. The threshold is set to be 0.5 for making the final decision on the prediction [18]. HOMER distributes the labels evenly and randomly into 3 subsets, and CLR is chosen to be the learner base for the HOMER. Furthermore, the decision tree C4.5 [36] is used as the base classifier for the selected PT based MLC methods in this paper. LP [1, 4] is used as multi-label base learner for TREMLC as well. The rest of parameters for TREMLC are as follow: each subset of feature set and instance set covers 70% of its original set; the number of models *m* set to be twice the size of the label set, i.e. m=2M, and label subset size *k* is 3 because m=2M and k=3 give better performances for RAkEL. Additionally, the minimum size of models is 200 if m=2M < 200 since adequate models are assumed to assist for gaining better performance for TREMLC. Threshold is chosen to be 0.5.

The predictive performances of the examined MLC algorithms are evaluated using the 10-fold cross-validation. Multi-label classification evaluation measures including the example-based Hamming-loss, label-based micro averaging F1-measure and ranking-based average precision are employed to present the evaluation results of the examined MLC algorithms. Additionally, the records of the evaluation time for the algorithms are also calculated in order to estimate the computational complexity of the algorithms. The experiments have been performed on VPAC super computer platform providing adequate memory and speed.

V. Results and Discussion

A. Training Performance with Parametesr Selection

The experiment reported in this section examines the impact of parameter selections on the behavior of TREMLC. It is conducted on the training set of the *scene* dataset. The training set is obtained using the standard train/test splits of the dataset [4]. Selection of the best set of main parameters, i.e. number of models *m*, size of feature subsets S_f, size of label subsets S_l, and size of instance set b (or bag size percentage), are important for achieving optimal performance of TREMLC. These parameters can be obtained based on iterative training of TREMLC. Firstly, the maximum number of models set to be m=2L, and varied from 1 to 2L with an increment, additionally, set the minimum number of models to 200 if m<200; then, set the maximum lengths of the S_f and S_l to be 70% of their original sets; next, vary these parameters from 1 to the specified maximum sizes with increments. Besides, the bag percentage set to vary from 20 to 100, accordingly, a set of optimal parameters m, S_f, S_l, and b can be chosen for TREMLC at the end of the training.

The change of the size of randomly selected feature subset, label subset and instance subset can bring different benefits to the TREMLC algorithm, which can be observed in Tables II. The table shows that the best training performance of TREMLC in terms of Hamming-loss (0.079131) and micro F1-measure (0.762233) is achieved when the number of models m=76, bag percentage b=70, size of feature subset S_f =51, size of the label subset S_l =3; the performance of TREMLC is measured with the average precision using a different set of parameters, i.e. the m=76, b=70, S_f =41, S_l=3. The difference between the two set of parameters is due to the different size of feature subset, i.e. S_f =51 for the best Hamming-loss and micro F1-measure, but S_f =41 for the micro averaged F-measure. The training time varies when different sets of parameters are used. Since the *scene* dataset is relatively small, especially its label set size is quite small, i.e. 6, the training of TREMLC on this dataset is not costly.

TABLE II. Optimal Parameter Selection for TREMLC on Scene Training Set

m	b	S_f	S_l	Hamm. -loss	Micro F1-m	Average Precision	Training -time
76	**70**	**51**	**3**	**0.079131**	**0.762233**	0.878891	6.791927
76	70	41	3	0.085044	0.708779	**0.891581**	6.038281

Selected optimal parameters may vary for different type of datasets, which can be observed from TABLE II. Once the best parameters are selected, they can be used for testing the predictive performance of TREMLC on the chosen dataset.

B. Predictive Performance

Predictive performances of TREMLC vs. existing MLC counterparts are given in TABLES III-VI using example based Hamming-loss, label-based micro F1-measure and ranking-based average precision. In order to fairly compare TREMLC with its counterparts, default parameters are used for all the examined algorithms (see Section IV).

As can be seen from Table III, TREMLC performed the best in terms of Hamming-loss when tested on all the selected multi-label datasets, i.e. *medical, enron*, *yeast, emotions, scene* and *mediamill,* since the smaller value of Hamming-loss, the better performance of the MLC algorithms. In the second place, ML-KNN performed nicely on *yeast*, *scene* and *mediamill,* while RAEKL performed well on *emotions, medical* and *enron*, and it climbed to the third best place on *scene*, *mediamill* and *yeast*. Furthermore, CLR also achieved reasonably good results on the *medical*, *enron* and *mediamill*.

According to TABLE IV, under micro averaging F1-measure, TREMLC is evaluated as the best performing algorithm on *yeast,* and *emotions,* and achieved the second best performance on the rest of the selected datasets. Note that, TREMLC got only a minor difference with the top performing algorithms on *medical*, *enron*, *scene* and *mediamill*. ML-KNN achieved the best performance on *scene* and reached the second place on the *yeast*, while EPS jumped to the highest performance level on mediamill, BR climbed to the top on *medical,* and RAKEL reached the best on *enron* and the second best on *mediamill, emotions* and *medical*. In the next level of ranking, CLR performed well on almost all the selected datasets.

Using average precision measure, the TREMLC algorithm is ranked the best among the counterparts on almost all the selected evaluation datasets, except for the *mediamill*, which can be observed from TABLE V. ML-KNN reached the best performance level on *mediamill* and approached to the second best level on *yeast, emotions* and *scene*, while EPS reached to the second best level on *enron* and *mediamill*, CLR achieved the second best on *medical,* and gain the above averaging performance on the rest of datasets. Note that, RAKEL showed the next performance level on almost all the selected datasets.

Overall, TREMLC achieved the top performance on all six evaluation datasets under Hamming-loss; it reached to the top performance on two out of six evaluation datasets under micro-averaged F1-measure, while EPS, ML-KNN, BR and RAkEL achieved the best performance individually on one dataset only under the F1; TREMLC showed excellence on five out of six datasets when measured with ranking based average precision, while ML-KNN gain the best performance on *mediamill* dataset. It can be also concluded that TREMLC possesses better geral applicability, i.e. it not only works well on smaller sized datasets with different type of attributes, e.g. nominal and numerical, but also is effective on large sized datasets with both large label set size (e. g. mediamill) and large feature set size (e.g. medical). Hence, TREMLC can be recommended for application to learning and mining multi-label data in various domains.

C. Evaluation Time

TABLE VI shows that the ML-KNN is the most efficient algorithm among the examined algorithms when tested on all the selected datasets and BR is the second efficient method. The most time consuming MLC algorithms on larger sized dataset *mediamill* is LP, followed by RAKEL, CLR and TREMLC. For the remaining relatively smaller datasets, TREMLC is identified as most time consuming algorithm. This may be due to the fact that TREMLC constructs ensemble classifiers iteratively with triple times randomly selected subsets, which takes time. As another randomized ensemble MLC algorithm, RAkEL also consumed great amount of time to build MLC classifiers on those larger sized datasets. That is, the sizes of the datasets and dimensionality of the data in feature space and label space greatly influence the efficiencies of the MLC algorithms. Improvement of the computational efficiency of the TREMLC algorithm is a critical task for the next step of this research work.

VI. Conclusion

The paper presented a new ensemble learning method, named triple-random ensemble classification for dealing with multi-label learning problems. The TREMLC method is a combination of the three randomization methods, i.e. using random subspace method for feature subset selection, bagging for examples sampling and random k-labelsets ensemble learning RAkEL for label subsets sampling.

The experiments were carried out on six small to large datasets. Popular MLC evaluation measurements are chosen from three major types, i.e. example-based Hamming-loss, label-based micro averaging F1-measure and ranking-based average precision, to present the experimental evaluation results. Additionally, the influence of parameter selections on the training performance of TREMLC was described. Then, the default parameters were applied to all the examined MLC algorithms in order to compare the predictive performances of the algorithms fairly.

The empirical investigation results show that TREMLC performs better than its examined counterparts when evaluated on the selected six evaluation datasets. Hence, the proposed TREMLC method is suggested for applying to a wide range of multi-label learning problems thanks to its general applicability. However, further optimization of the computational complexity of the TREMLC method is a critical task for the future development.

REFERENCES

[1] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, "Learning multi-label scene classification", Pattern Recognition, vol.37, no.9, 2004, pp.1757 – 1771.

[2] R. E. Schapire and Yoram Singer, "BoosTexter: A boosting-based system for text categorization", Machine Learning, vol.39, no.2/3, 2000, pp.135-168.

[3] Z. H. Zhou, M. L. Zhang, "Multi-instance multi-label learning with application to scene classification", Scho"lkopf. B., Platt. J. C., Hoffman. T., eds. NIPS 2006, MIT Press, pp. 1609 – 1616.

[4] G. Tsoumakas, I. Katakis, "Multi label classification: An overview", International J. of Data Warehousing and Mining, David Taniar Ed., Idea Group Publishing, vol.3, no.3, 2007, pp. 1-13.

[5] G. Tsoumakas, I. Katakis, I. Vlahavas, "Mining multi-label data", Data Mining and Knowledge Discovery Handbook, O. Maimon, L. Rokach (Ed.), Springer, 2nd edition, 2010.

[6] W. Chen, J. Yan, B. Ahang, Z. Chen, and Q. Yang, "Document transformation for multi-label feature selection in text categorization", Proc. of Seventh IEEE International Conf. on Data Mining, 2007, pp. 451- 456.

[7] A. Esuli, T. Fagni, F. Sebastiani, "Boosting multi-label hierarchical text categorization", J. of Information Retrieval, vol.11, 2008, pp. 287–313.

[8] J. Read, "A pruned problem transformation method for multi-label classification", Proc. of the 2008 New Zealand Computer Science Research Student Conference (NZCSRS 2008), pp 143-150.

[9] J. Read, B. Pfahringer, G. Holmes, "Multi-label classification using ensembles of pruned sets", 2008 Eighth IEEE International Conf. on Data Mining (ICDM '08), Dec. 2008, pp.995 – 1000.

[10] A. Elisseeff and J. Weston, "A Kernel method for multi-labeled classification", T. G. Dietterich, S. Becker, Z. Ghahramani Eds. Advances in Neural Information Processing Systems 14, MIT Press, Cambridge, MA, 2002, pp.681-687.

[11] K. Brinker, E. Hullermeier, "Case-based multi-label ranking", Proc. of the 20th International Conf. on Artificial Intelligence (IJCAI '07), Hyderabad, India, 2007, pp. 702–707.

[12] K. Trohidis, G. Tsoumakas, G. Kalliris, I. Vlahavas. "Multilabel Classification of Music into Emotions", Proc. of 9th International Conf. on Music Information Retrieval (ISMIR 2008), pp. 325-330, Philadelphia, PA, USA, 2008.

[13] J. Fu"rnkranz, E. H"ullermeier, E. L. Mencia and K. Brinker, "Multilabel classification via calibrated label ranking", J. of Machine Learning. vol. 73, 2008, pp. 133-153.

[14] F. Kang, R. Jin and R. Sukthankar, "Correlated label propagation with application to multi-label learning", Proc. of the 2006 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'06), pp.1719 -1726.

[15] M. Wang, X. Zhou and T.-S. Chua, "Automatic image annotation via local multi-label classification", Proc. of the 2008 International Conf. on Content-based Image and Video Retrieval, Canada, pp.75-84.

[16] G. Nasierding, G. Tsoumakas and A. Z. Kouzani, "Clustering based multi-label classification for image annotation and retrieval", Proc. of 2009 IEEE International Conf. on Systems, Man, and Cybernetics (SMC 09), Texas, USA, October 2009, pp. 4627-4632.

[17] A. Dimou, G. Tsoumakas, V. Mezaris, I. Kompatsiaris, I. Vlahavas, "An empirical study of multi-label learning methods for video annotation", 7th International Workshop on Content-Based Multimedia Indexing, IEEE, Chania, Crete, 2009.

[18] G. Tsoumakas, I. Katakis, I. Vlahavas, "Random k-Labelsets for multi-label classification", IEEE Transactions on Knowledge and Data Engineering, IEEE, 2010.

[19] T. K. Ho, "The random subspace method for constructing decision forests", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no.8, 1998, pp.832-844.

[20] R. Bryll, R. Gutierrez-Osuna, F. Quek, "Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets", Pattern Recognition, vol.36, no.6, 2003, pp.1291- 1302.

[21] J. DePasquale , R. Polikar¨Random feature subset selection for ensemble based classification of data with missing features", Lecture Notes in Computer Science, vol. 4472, M. Haindl and F. Roli, Eds. Berlin: Springer-Verlag, 2007, pp.251-260.

[22] L. Breiman, (1996a), "Bagging predictors", Machine Learning, vol. 24, no. 2, pp. 123-140.

[23] R. Polikar, "Bootstrap-inspired techniques in computational Intelligence", Signal Processing Magazine, IEEE, July 2007, vol. 24, Issue 4, pp. 59-72.

[24] L. Breiman, "Random Forests", Machine Learning, vol.45, no.1, 2001, pp. 5–32.

[25] P. Panov and S. Džeroski, "Combining bagging and random subspaces to create better ensembles", Lecture Notes in Computer Science, vol.4723, 2007, Advances in Intelligent Data Analysis VII, Springer Berlin, pp. 118-129.

[26] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm", Proc. of the Thirteenth International Conf. on Machine Learning, 1996, pp. 148-156.

[27] R. Yan, J. Tesic, and J. R. Smith, "Model-shared subspace boosting for multi-label classification", Proc. of Knowledge Discovery and Data Mining , 2007 (KDD'07), California, USA, pp. 834-843.

[28] S. Ji, Li. Tang, S. Yu, and J. Ye, "Extracting shared subspace for multi-label classification", Proc. of the 14th SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, 2008, Las Vegas, USA.

[29] G. Nasierding, B. Duc, S. L. A. Lee, A. Z. Kouzani, "Dual-random ensemble method for multi-label classification of biological data", Proc. of IEEE International Symposium on Bioelectronics and Bioinformatics, Dec. 2009, Melbourne, Australia, pp. 49-52.

[30] G. Tsoumakas, I. Katakis, I. Vlahavas, "Effective and efficient multi-label classification in domains with large number of labels", Proc. of ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08), Antwerp, Belgium, 2008.

[31] M. L. Zhang, Z. H. Zhou, "ML – KNN: A lazy learning approach to multi-label learning", Pattern Recognition, vol.40, no.7, 2007, pp.2038–2048.

[32] E. Spyromitros, G. Tsoumakas, I. Vlahavas, "An empirical study of lazy multilabel classification algorithms", Proc. of 5th Hellenic Conf. on Artificial Intelligence, Syros, Greece, 2008, Springer, pp. 401-406.

[33] M. L. Zhang, Z. H. Zhou, "Multi-label neural networks with applications to functional genomics and text categorization", IEEE Transaction on Knowledge and Data Engineering, vol.18, no.10, 2006, pp.1338–1351.

[34] F. D. Comite, R. Gileron, M. Tommasi, "Learning multi-label alternating decision tree from texts and data", Proc. of the 3rd International Conf. on Machine Learning and Data Mining in Pattern Recognition (MLDM 2003), Leipzig, Germany, July 2003, pp. 35–49.

[35] N. Ghamrawi and A. McCallum, "Collective multi-label classification", Proc. of the 14th ACM International Conf. on Information and Knowledge Management, 2005, pp. 195-200.

[36] J. R. Quinlan. C4.5: Programs for Machine Learning. San Mateo, CA; Morgan Kaufmann, 1993.

[37] Computational medical center:Medical NLP challenge. URL: http://www.computationalmedicine.org/chllenge/index.php.

[38] UC Berkeley enron email analysis project. URL: http://bailando.sims.berkeley.edu/enron_email.html.

[39] C.G.M. Snoek, M. Worring, J.C. van Gemert, J.-M. Geusebroek, and A.W.M. Smeulders, "The challenge problem for automated detection of 101 semantic concepts in multimedia", Proc. of ACM Multimedia, Santa Barbara, USA, October 2006, pp. 421-430.

[40] I. H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2005.

TABLE III. PREDICTIVE PERFORMANCES OF MLC ALGORITHMS MEASURED WITH *HAMMING-LOSS*

Algorithm	medical	enron	yeast	emotions	scene	mediamill
TREMLC	**0.010319**	**0.045962**	**0.18783**	**0.180758**	**0.082821**	**0.02814**
EPS	0.011886	0.054341	0.245494	0.233644	0.102275	0.030748
ML-KNN	0.015112	0.052457	0.193296	0.195122	0.085309	0.028189
BR	0.010344	0.051869	0.245432	0.247401	0.136762	0.03349
LP	0.013476	0.071492	0.277901	0.27775	0.143819	0.042314
RAKEL	0.010411	0.047744	0.219515	0.217538	0.098884	0.029003
CLR	0.010364	0.047457	0.220227	0.242302	0.138348	0.028317
HOMER	0.011229	0.065759	0.286063	0.278315	0.165357	0.04496

TABLE IV. PREDICTIVE PERFORMANCES OF MLC ALGORITHMS MEASURED WITH *MICRO F1-MEASURE.*

Algorithm	medical	enron	yeast	emotions	scene	mediamill
TREMLC	0.803205	0.574332	**0.654469**	**0.680128**	0.730163	0.62180
EPS	0.78320	0.557836	0.640016	0.659547	0.703534	**0.632455**
ML-KNN	0.678398	0.470077	0.647126	0.659763	**0.737853**	0.597035
BR	**0.809087**	0.540546	0.585697	0.601974	0.619391	0.56484
LP	0.752437	0.429285	0.54057	0.548976	0.597837	0.50677
RAKEL	0.808453	**0.576458**	0.620809	0.638645	0.697095	0.610112
CLR	0.807684	0.567556	0.615765	0.627627	0.627572	0.596357
HOMER	0.798167	0.52762	0.589529	0.601781	0.574643	0.533611

TABLE V. PREDICTIVE PERFORMANCES OF MLC ALGORITHMS MEASURED WITH *AVERAGE PRECISION.*

Algorithm	medical	enron	yeast	emotions	*scene*	mediamill
TREMLC	**0.871313**	**0.64653**	**0.771701**	**0.820078**	**0.880521**	0.69915
EPS	0.839276	0.633101	0.733262	0.770904	0.832368	0.745029
ML-KNN	0.813356	0.629096	0.765812	0.796454	0.865763	**0.755868**
BR	0.834109	0.588909	0.621568	0.701352	0.710852	0.576282
LP	0.814071	0.509567	0.645407	0.683013	0.739422	0.57648
RAKEL	0.826389	0.618527	0.724137	0.783797	0.835592	0.691481
CLR	0.851976	0.627808	0.729328	0.759014	0.809449	0.699258
HOMER	0.801279	0.49263	0.64668	0.702491	0.71679	0.524566

TABLE VI. *EVALUATION TIMES* OF THE EXAMINED MLC ALGORITHMS.

Algorithm	medical	enron	yeast	emotions	scene	mediamill
TREMLC	51.2055	357.275	117.0322	8.769833	172.8003	2020.85
EPS	3.5475	25.163	14.294167	0.647	12.758667	1849.847
ML-KNN	**0.1185**	**1.180833**	**2.947167**	**0.1195**	**2.757667**	**339.457**
BR	3.496833	49.22017	3.330667	0.153833	3.281833	727.2203
LP	0.7685	3.7425	5.336	0.140167	2.487	3207.094
RAKEL	21.08367	280.7615	26.50683	0.770333	16.16567	3081.987
CLR	6.285	106.0373	9.6385	0.285	5.075333	2577.75
HOMER	3.688833	32.22417	4.744833	0.225667	4.356	533.3867