This is the published version:


Available from Deakin Research Online:

http://hdl.handle.net/10536/DRO/DU:30032345

Reproduced with the kind permission of the copyright owner.

Copyright: 2010, IEEE.
A Triple-Random Ensemble Classification Method for Mining Multi-label Data

Gulisong Nasierding, Abbas Z. Kouzani, Grigoris Tsoumakas

Dept. of Computer Science and Technology
Xinjiang Normal University
102 Xin Yi Rd, Urumqi, P.R. China
gulnas9@gmail.com

School of Engineering
Deakin University
Geelong, VIC 3217, Australia
kouzani@deakin.edu.au

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]. \( L = \{1, 2, \ldots, M\} \) is used to represent the finite set of labels, and \( D = \{(\mathbf{x}_i, Y_i), i = 1, \ldots, N\} \) is used to describe a set of multi-label training examples, where \( \mathbf{x}_i \) 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 subsampling 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
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 ranks 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 posteriori 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],

50
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.

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 \( \tilde{x} \) arrives for classification, each classifier \( h_i \) provides binary predictions \( h_i(\tilde{x}, l_j) \) for each label \( l_j \) in the corresponding training subset with the size of feature dimension \( S_j < F \) and label dimension \( S_j < L \). 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.

![Figure 1. Pseudo code of the TREMLC training process.](image)

![Figure 2. Pseudo code for the TREMLC testing process.](image)
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 color 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>
<thead>
<tr>
<th>Name</th>
<th>Domain</th>
<th>Instances</th>
<th>Attributes</th>
<th>Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>medical</td>
<td>text</td>
<td>978</td>
<td>1449 nom.</td>
<td>45</td>
</tr>
<tr>
<td>enron</td>
<td>text</td>
<td>1702</td>
<td>1001 nom.</td>
<td>53</td>
</tr>
<tr>
<td>yeast</td>
<td>biology</td>
<td>2417</td>
<td>103 num.</td>
<td>14</td>
</tr>
<tr>
<td>emotions</td>
<td>music</td>
<td>593</td>
<td>72 num.</td>
<td>6</td>
</tr>
<tr>
<td>scene</td>
<td>image</td>
<td>2407</td>
<td>294 num.</td>
<td>6</td>
</tr>
<tr>
<td>mediamill</td>
<td>video</td>
<td>43907</td>
<td>120 num.</td>
<td>101</td>
</tr>
</tbody>
</table>

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 \( (x_i, Y_i) \), where \( i = 1, 2, \ldots, N \), where \( Y_i \subseteq L \) denotes a set of true labels, \( L = \{l_j \mid j = 1, \ldots, M \} \) denotes the set of all the labels, and \( x_i \) denotes a new instance. Hence, Hamming-loss is:

\[
H\text{-}loss = \frac{1}{N} \sum_{i=1}^{N} \sum_{l=1}^{M} \frac{|Y_i \Delta Z_i|}{M}
\]

where \( Z_i = h(x_i) \) denotes a set of labels predicted by a multi-label classifier \( h \) for an example \( x_i \), and \( \Delta \) 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:

\[
F1\text{-}measure = \frac{2 \cdot tp}{2 \cdot tp + fp + fn}
\]

where \( tp, fp, tn, fn \) 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:

\[
B_{micro} = \frac{M}{\sum_{l=1}^{M} tp_l + \sum_{l=1}^{M} fp_l + \sum_{l=1}^{M} tn_l + \sum_{l=1}^{M} fn_l}
\]

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 \( l \in Y_i \):

\[
\text{avg. Pr ec.} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|Y_i|} \sum_{l \in Y_i} \left[ \frac{1}{r_i(l)} \leq r_i(l) \right]
\]  

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

C. Experimental Setting

In order to empirically 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 Parameters 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 \( S_b \), size of label subsets \( S_f \), 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 \( S_b \) 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 \( m, S_f, S_b \) 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 \( S_b=3 \); the performance of TREMLC is measured with the average precision using a different set of parameters, i.e. \( m=76 \), \( b=70 \), \( S_f=41 \), \( S_b=3 \). 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>
<thead>
<tr>
<th>( m )</th>
<th>( b )</th>
<th>( S_f )</th>
<th>( S_b )</th>
<th>Hamming-loss</th>
<th>Micro F1-m</th>
<th>Average Precision</th>
<th>Training -time</th>
</tr>
</thead>
<tbody>
<tr>
<td>76</td>
<td>70</td>
<td>51</td>
<td>3</td>
<td>0.079131</td>
<td>0.762233</td>
<td>0.878891</td>
<td>6.791927</td>
</tr>
<tr>
<td>76</td>
<td>70</td>
<td>41</td>
<td>3</td>
<td>0.085044</td>
<td>0.708779</td>
<td>0.891581</td>
<td>6.038281</td>
</tr>
</tbody>
</table>

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.

53
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 RAkEL 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 general 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


[37] Computational medical center:Medical NLP challenge. URL: http://bailando.sims.berkeley.edu/enron_email.html

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


[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

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>medical</th>
<th>enron</th>
<th>yeast</th>
<th>emotions</th>
<th>scene</th>
<th>mediamill</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREMLC</td>
<td>0.010319</td>
<td>0.045962</td>
<td>0.18783</td>
<td>0.180758</td>
<td>0.082821</td>
<td>0.02814</td>
</tr>
<tr>
<td>EPS</td>
<td>0.011886</td>
<td>0.054341</td>
<td>0.245494</td>
<td>0.233644</td>
<td>0.102275</td>
<td>0.030748</td>
</tr>
<tr>
<td>ML-KNN</td>
<td>0.015112</td>
<td>0.052457</td>
<td>0.193296</td>
<td>0.195122</td>
<td>0.085309</td>
<td>0.028189</td>
</tr>
<tr>
<td>BR</td>
<td>0.010344</td>
<td>0.051869</td>
<td>0.245432</td>
<td>0.247401</td>
<td>0.136762</td>
<td>0.03349</td>
</tr>
<tr>
<td>LP</td>
<td>0.013476</td>
<td>0.071492</td>
<td>0.277901</td>
<td>0.277750</td>
<td>0.143819</td>
<td>0.042314</td>
</tr>
<tr>
<td>RAKEL</td>
<td>0.010411</td>
<td>0.047744</td>
<td>0.219515</td>
<td>0.217538</td>
<td>0.098884</td>
<td>0.029003</td>
</tr>
<tr>
<td>CLR</td>
<td>0.010364</td>
<td>0.047457</td>
<td>0.220227</td>
<td>0.242302</td>
<td>0.138348</td>
<td>0.028317</td>
</tr>
<tr>
<td>HOMER</td>
<td>0.011229</td>
<td>0.065759</td>
<td>0.286063</td>
<td>0.278315</td>
<td>0.165357</td>
<td>0.04496</td>
</tr>
</tbody>
</table>

### TABLE IV. Predictive Performances of MLC Algorithms Measured with Micro F1-measure

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>medical</th>
<th>enron</th>
<th>yeast</th>
<th>emotions</th>
<th>scene</th>
<th>mediamill</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREMLC</td>
<td>0.803205</td>
<td>0.574332</td>
<td>0.654469</td>
<td>0.680128</td>
<td>0.730163</td>
<td>0.62180</td>
</tr>
<tr>
<td>EPS</td>
<td>0.78320</td>
<td>0.557836</td>
<td>0.640016</td>
<td>0.659547</td>
<td>0.703534</td>
<td>0.632455</td>
</tr>
<tr>
<td>ML-KNN</td>
<td>0.678398</td>
<td>0.470077</td>
<td>0.647126</td>
<td>0.659763</td>
<td>0.737853</td>
<td>0.597035</td>
</tr>
<tr>
<td>BR</td>
<td>0.809087</td>
<td>0.540546</td>
<td>0.585697</td>
<td>0.601974</td>
<td>0.619391</td>
<td>0.56484</td>
</tr>
<tr>
<td>LP</td>
<td>0.752437</td>
<td>0.429285</td>
<td>0.54057</td>
<td>0.548976</td>
<td>0.597837</td>
<td>0.50677</td>
</tr>
<tr>
<td>RAKEL</td>
<td>0.808453</td>
<td>0.576458</td>
<td>0.620809</td>
<td>0.638645</td>
<td>0.697095</td>
<td>0.610112</td>
</tr>
<tr>
<td>CLR</td>
<td>0.807684</td>
<td>0.567556</td>
<td>0.613756</td>
<td>0.627627</td>
<td>0.627572</td>
<td>0.596357</td>
</tr>
<tr>
<td>HOMER</td>
<td>0.798167</td>
<td>0.52762</td>
<td>0.589529</td>
<td>0.601781</td>
<td>0.574643</td>
<td>0.533611</td>
</tr>
</tbody>
</table>

### TABLE V. Predictive Performances of MLC Algorithms Measured with Average Precision

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>medical</th>
<th>enron</th>
<th>yeast</th>
<th>emotions</th>
<th>scene</th>
<th>mediamill</th>
</tr>
</thead>
<tbody>
<tr>
<td>TREMLC</td>
<td>0.871313</td>
<td>0.64653</td>
<td>0.771701</td>
<td>0.820078</td>
<td>0.880521</td>
<td>0.69915</td>
</tr>
<tr>
<td>EPS</td>
<td>0.839276</td>
<td>0.633101</td>
<td>0.733262</td>
<td>0.770904</td>
<td>0.832368</td>
<td>0.745029</td>
</tr>
<tr>
<td>ML-KNN</td>
<td>0.813356</td>
<td>0.629096</td>
<td>0.765812</td>
<td>0.796454</td>
<td>0.865763</td>
<td>0.755868</td>
</tr>
<tr>
<td>BR</td>
<td>0.834109</td>
<td>0.588909</td>
<td>0.621568</td>
<td>0.701352</td>
<td>0.710852</td>
<td>0.576282</td>
</tr>
<tr>
<td>LP</td>
<td>0.814071</td>
<td>0.509567</td>
<td>0.645407</td>
<td>0.683013</td>
<td>0.743422</td>
<td>0.57648</td>
</tr>
<tr>
<td>RAKEL</td>
<td>0.826389</td>
<td>0.618527</td>
<td>0.724137</td>
<td>0.783797</td>
<td>0.835592</td>
<td>0.691481</td>
</tr>
<tr>
<td>CLR</td>
<td>0.851976</td>
<td>0.627808</td>
<td>0.729328</td>
<td>0.759014</td>
<td>0.809449</td>
<td>0.699258</td>
</tr>
<tr>
<td>HOMER</td>
<td>0.801279</td>
<td>0.49263</td>
<td>0.584668</td>
<td>0.702491</td>
<td>0.716795</td>
<td>0.524566</td>
</tr>
</tbody>
</table>

### TABLE VI. Evaluation Times of the Examined MLC Algorithms

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