IEEE International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), August 01-02,2014, Dr. Virendra Swarup Group of Institutions, Unnao, India

A Novel Approach for Multi-label Classification using Probabilistic Classifiers

Gangadhara Rao Kommu
Asst.Professor,Department of IT,CBIT
Hyderabad, Telangana-5000075
Email: kgr@cbit.ac.in

M.Trupthi
Asst.Professor,Department of IT,CBIT
Hyderabad, Telangana-5000075
Email: trupthijan@gmail.com

Suresh Pabboju
Prof & Head,Department of IT,CBIT
Hyderabad, Telangana-5000075
Email: plpsuresh@gmail.com

Abstract*—This paper presents different approaches to solve multi-label classification. In recent study there exist many approaches to solve multi-label classification problems. Which are used in various applications such as protein function classification, music categorization, semantic scene classification, etc., It in-turn uses different evaluation metrics like hamming loss and subset loss for solving multi-label classification but which are deterministic in nature. In this work, we concentrate on probabilistic models and develop two new probabilistic approaches to solve multi-label classification. One of the two approaches is based on logistic regression and nearest neighbor classifiers. This approach is similar to binary relevance method. The training phase of this approach is same as the training phase of binary relevance method, but differs in testing phase. The second approach is based on the idea of grouping related labels. This method trains one classifier for each group and the corresponding label is called as group representative. Predict other labels based on the predicted labels of group representative. The relations between the labels are found using the concept of association rule mining.

***Keywords*—multi-label, hamming ,probablistic, logistic, regression**

I. Introduction

In supervised learning, most of the research concentrated on data where each example is labelled with only one label, known as single-label data. But in many real life applications, examples are associated with a subset of labels unlike single-label data. This setting, where examples are tagged with a set of labels, is known as multi-label data.Binary classification and multi-class classification are special cases of single-label classification. If the label set L contains only two elements $(i.e \mid L \mid= 2)$ then single-label classification becomes binary classification. Otherwise $(i.e., \mid L \mid> 2)$, it becomes multi-class classification.

There exist two types of procedures to solve the multi-label classification problem. They are multi-label classification (MLC) and multi-label ranking (MLR). The former procedure assigns a subset of label set L to each test example using a model which is learnt from training data. But the latter approach results in ranking among labels for each test example. Using some heuristics, we can classify test data with ranking approaches also.

In literature, there exist a lot of methods which deal with multi-label classification. Those methods are categorized into two types. They are 1) Problem Transformation (PT) methods, in which the original multi-label classification problem can be transformed into several single-label classification problems. These methods are algorithm independent. Examples of these kind of procedures are Binary Relevance(BR) method [5], Label Powerset(LP) method [5] etc. 2)Algorithm Adaptation (AA) methods. These methods extend single-label classification methods to deal with multi-label classification. Examples of these methods are extending kNN $(called\ MLkNN)$ [7] , extending AdaBoost [7] etc. Apart from these methods, there also exist ensemble methods for some of the MLC approaches. For example, Ensemble Classifier Chain (ECC) [5] is the ensemble approach of the classifier chain method.

There also exist several types in multi-label classification. Some of them are multi-label collective classification, hierarchical multi-label classification etc. In multi-label collective classification, related instances in a dataset are classified together unlike in normal multi-label classification. For example, consider two collaborated researchers. If we know one of them has interests in data mining, then there is a high probability that the other researcher also has interest in data mining. This tells correlation among instances. Similarly correlated labels are also predicted simultaneously. For example, one person may have interest in machine learning as well as artificial intelligence i.e., ML and AI are correlated. To know the correlation among instances and correlation among labels, we need to provide extra input to algorithms which solve multi-label collective classification.

The rest of the paper is organised as follows. In section 2, we will look at some of the existing methods of MLC. In section 3, we will briefly look at logistic regression method for binary classification. In section 4, we will see how we extended LR to multi-class classification. In section 5, we discuss various multi-label datasets and its statistics. In section 6, we look at some of the evaluation metrics for MLC. In section 7, we will look at our multi-label classification approaches. In section 8, we will see experimental results of proposed methods. In section 9, we conclude this paper.

II. Literature Review

As we have seen earlier, there exist many multi-label classification methods. We will look at some of the methods. The oldest approach is Binary Relevance (BR) method. This is one of the PT methods. This method trains one binary classifier for each label in the label set L, using modified data. Modified data consists of actual feature vector and one extra label which tells the absence or presence of the corresponding label. For

example, consider the dataset given in Table I
The data set consists of 4 examples and x_i is the feature vector

TABLE I. EXAMPLE DATASET

Sl No.	Feature Vector	Labels
1	x_1	l_1, l_2
2	x_2	l_3
3	x_3	l_1
4	x_4	l_2, l_3

of i^{th} example. Here, the label set L=$\{l_1, l_2, l_3\}$. Modified datasets for each label are given in Tables II - IV.

TABLE II. DATA-SETS FOR LABEL l_2

Sl No.	F.V	Labels
1	x_1	l_1
2	x_2	$\neg l_1$
3	x_3	l_1
4	x_4	$\neg l_1$

These are the modified datasets of the corresponding labels. To these datasets, apply binary classification problem to get different classifiers. In this way we get k binary classifiers where k is the cardinality of label set. In the above case, k=3. This method invokes all classifiers on a test example to assign the labels. This approach is simple. The disadvantage of this technique is that it doesn't take label correlation into account. We implemented this technique and we presented the results in Section 8.

The LabelPowerset (LP) method assumes every subset of the label set as one single different label. Using this rule the modified dataset is created. Using that modified dataset, a single-label classifier is learnt. This method invokes that classifier on a test example to assign labels. The modified dataset of Table I for LP method is given in Table V.

This method considers label correlation but it has a disadvantage of having higher complexity. Another disadvantage is that some of the subsets of label set are associated with very few examples which makes training process difficult.

In CC model, classifiers are linked along a chain where each classifier deals with the binary problem associated with label $l_i \in L$. The feature space of each link in the chain is extended with the 0/1 label associations of all previous links. In this way, CC method considers label correlation, unlike BR.

Each classifier C_i in the chain is responsible for learning and predicting the association of label l_i, given all prior binary relevance predictions in the chain $l_1, l_2, ..., l_{i-1}$. Consider the dataset given in Table I. Modified datasets for each label are given in Table V - VIII. In each of the modified datasets, the last column tells the presence or absence of label and all the remaining columns are treated as features. In the above example, classifier C_1 is learnt using the modified dataset shown in Table VI. Similarly C_2 and C_3 are learnt using datasets shown in Table VII and Table VIII respectively. To classify a test example, CC first invoke C_1 to get association of label l_1 and later, it invokes other classifiers on feature vector and associations of all previous labels. One disadvantage of CC is that it depends on the order of chain.

TABLE III. DATA-SET FOR l_1

Sl No.	F.V	Labels
1	x_1	l_2
2	x_2	$\neg l_2$
3	x_3	$\neg l_2$
4	x_4	l_2

TABLE IV. l_3

Sl No.	F.V	Labels
1	x_1	$\neg l_3$
2	x_2	l_3
3	x_3	$\neg l_3$
4	x_4	l_3

TABLE V. MODIFIED DATASET FOR LP METHOD

Sl No.	Feature Vector	Labels
1	x_1	l_{12}
2	x_2	l_3
3	x_3	l_1
4	x_4	l_{23}

TABLE VI. MODIFIED DATASET FOR l_1

Sl No.	Feature Vector	l_1
1	x_1	1
2	x_2	0
3	x_3	1
4	x_4	0

TABLE VII. MODIFIED DATASET FOR l_2

Sl No.	Feature Vector	l_1	l_2
1	x_1	1	1
2	x_2	0	0
3	x_3	1	0
4	x_4	0	1

TABLE VIII. MODIFIED DATASET FOR l_3

Sl No.	Feature Vector	l_1	l_2	l_3
1	x_1	1	1	0
2	x_2	0	0	1
3	x_3	1	0	0
4	x_4	0	1	1

The Random k-labelset$(RAkEL)$ method [6] constructs an ensemble of LP classifiers in which each LP classifier is trained with a different small random subset of the set of labels. In this way, RAkEL avoids the problem of Label Powerset algorithm and also it manages to take label correlations into account. The training algorithm for RAkEL is shown in Algorithm 2 and testing is as follows. For each test instance, label ranking is produced by averaging the prediction of each LP classifier and use appropriate threshold value to get final classification.

As we have seen earlier, multi-label classification problem transformed into one or more single-label classification problems in all problem transformation methods. The classifiers for those single-label classification are called as base classifiers. In our approaches, we used logistic regression as base classifier. So let us look at the details of logistic regression for both binary classification and multi-class classification before seeing the details of our proposed approaches.

III. LR: BINARY CLASSIFICATION

Before seeing the details of logistic regression, we first look at the format of data. The dataset D is of the form (X_i, Y_i) where $X_i = (x_{i1}, x_{i2}, ..., x_{ik})$ is the feature vector of i^{th}

example and k is the dimensionality of dataset. Y_i =1 if label is assigned, 0 otherwise.

In logistic regression, the probabilities are defined as follows.

$$P\left(y=1/\omega,X_i\right)=P_i=\frac{1}{1+e^{-\omega^T*X_i}} \tag{1}$$

$$P\left(y=0/\omega,X_i\right)=1-P_i=\frac{e^{-\omega^T*X_i}}{1+e^{-\omega^T*X_i}} \tag{2}$$

From both the above equations, we can generalize the probability formula as:

$$P\left(y_i/\omega,X_i\right)=P_i^{y_i}*\left(1-P_i\right)^{1-y_i} \tag{3}$$

Then the likelihood is as follows:

$$P\left(D/\omega\right)=\Pi_{i=1}^{n}P\left(y_i/X_i,\omega\right) \tag{4}$$

Posterior is proportional to product of likelihood and prior. i.e.,

$$\text{P}(\omega/D)\ \ \alpha\ P\left(D/\omega\right)*P\left(\omega\right)$$

We need to maximize the posterior of given dataset D. Assume prior has normal distribution. Instead of maximizing posterior directly, we maximize log of the posterior, i.e.,

$$\text{max log}[P\left(D/\omega\right)*P\left(\omega\right)]$$

$$\Rightarrow \text{max log}(P\left(D/\omega\right))\text{+log}(P\left(\omega\right))$$

$$\Rightarrow \text{max} \sum_{i=1}^{n} log\left(P\left(y_i/X_i,\omega\right)\right)\text{-}\tfrac{\lambda}{2}\|\omega\|^2 \qquad (from\,(4))$$

$$\Rightarrow \text{max} \sum_{i=1}^{n} log\left(P_i^{y_i}*\left(1-P_i\right)^{1-y_i}\right)\text{-}\tfrac{\lambda}{2}\|\omega\|^2$$

$$\Rightarrow\text{max} \sum_{i=1}^{n}\left[y_i log\left(p_i\right)+\left(1-y_i\right)log\left(1-p_i\right)\right]-\tfrac{\lambda}{2}\parallel w\parallel^2$$

From the fact that maximizing a function f is equivalent to minimizing a function -f, the objective function becomes

Objective function:

$$\text{E} = min\ \tfrac{\lambda}{2}\parallel w\parallel^2-\sum_{i=1}^{n}\left[y_i log\left(P_i\right)+\left(1-y_i\right)log\left(1-P_i\right)\right]$$

Differentiation:

$$\nabla\text{E} = \lambda w-\Sigma_{i=1}^{n}\left(y_i-p_i\right)x_i$$

To solve the above optimization problem E, we used steepest descent method with backtracking line search. The parameters needed are c$_1$=0.5, δ=0.75 and α=0.05. We have chosen λ value appropriately. For this method, we need differentiation of objective function that we have found already. Our approach is explained in algorithm 1.

After the execution of algorithm 1 on a training data, we get a classifier. Invoke that classifier on all test set examples to get the probabilities. Assign a label to each test example based on that probability i.e., if probability is greater than 0.5, then the predicted label is 1, otherwise 0. Using these predicted labels and actual labels, find test set accuracy. The results of implementation of above algorithm are described in section 8.

Algorithm 1 Logistic Regression for Binary Classification

```
Input: Train_data,Train_labels,
    lambda.($Each\ example\ in\ data\ set\ is\ row-wise$)
Output: Classifier function
 1: E⟵ $Objectivefunction$
 2: F⟵ $\nabla E$
 3: $\lambda$ ⟵ lambda
 4: w⟵ Initialize
 5: itr⟵0
 6: while($i < 50$)
 7:     $f_1$ ⟵ compute E at w
 8:     g⟵ compute F at w
 9:     $c_1$ ⟵ 0.5
10:     $\alpha$ ⟵ 0.005
11:     $\phi_1 \longleftarrow f_1 + c_1 * \alpha * g^t * g$
12:     w$_{new}$ =w-$\alpha * g$
13:     $\phi$ ⟵ compute E at w$_{new}$
14:     while($\phi_1 < \phi$)
            $\alpha \longleftarrow \alpha * 0.75$
            w$_{new}$ =w-$\alpha * g$
            $\phi$ ⟵ compute E at w$_{new}$
            $\phi_1 \longleftarrow f_1 + c_1 * \alpha * g^t * g$
        end while
15:     w=w-$\alpha * g$
16:     itr=itr+1
17: end while
```

IV. LR: Multi-Class Classification

Before seeing the details of LR for multi-class classification, we look at the format of dataset D. Let the number of classes be K i.e $|L| = K$. Each example is of the form (X_i, y_i) where $X_i = (x_{i1}, x_{i2}, ..., x_{id})$ is the feature vector of i^{th} example. d is the dimensionality of dataset. $y_i \in \{1, 2, ..., \mid L \mid\}$ is the label of i^{th} example.

The probability of X_i^{th} example having y_i label is given in equations 5,6.
For $y_i \in \{1, 2, ..., K-1\}$,

$$P_{y_i}\left(X_i,w\right)=\frac{exp^{w_{y_i}^T*X_i}}{1+\sum_{k=1}^{K-1}exp^{w_k^T*X_i}} \tag{5}$$

$$P_K\left(X_i,w\right)=\frac{1}{1+\sum_{k=1}^{K-1}exp^{w_k^T*X_i}} \tag{6}$$

Posterior is proportional to product of likelihood and prior, i.e.,

$$\text{P}(\omega/D)\ \ \alpha\ P\left(D/\omega\right)*P\left(\omega\right)$$

Now will maximize log of the posterior of given data set D i.e.,

$$\text{max log}[P\left(D/\omega\right)*P\left(\omega\right)]$$

$$\Rightarrow \text{max log}(P\left(D/\omega\right))\text{+log}(P\left(\omega\right))$$

$$\Rightarrow \max \sum_{i=1}^{n} log\left(P\left(y_i/X_i,\omega\right)\right)\text{-}\frac{\lambda}{2}\left\|\omega\right\|^2 \qquad (from\,(4))$$

$$\Rightarrow \max \sum_{i=1}^{n} log\left(\frac{exp^{w_{y_i}^T * X_i}}{1+\sum_{k=1}^{K-1} exp^{w_k^T * X_i}}\right)\text{-}\frac{\lambda}{2}\left\|\omega\right\|^2$$

$$\Rightarrow \max \sum_{i=1}^{n}\left[w_{y_i}^T * X_i - log\left(\sum_{k=1}^{K} exp^{w_k^T * X_i}\right)\right] \text{-}\frac{\lambda}{2}\left\|\omega\right\|^2$$

Now the objective function becomes

Objective function:

$$\text{E= min } \frac{\lambda}{2}\left\|\omega\right\|^2\text{-}\sum_{i=1}^{n}\left[w_{y_i}^T * X_i - log\left(\sum_{k=1}^{K} exp^{w_k^T * X_i}\right)\right]$$

Differentiation:

$$\nabla \text{E}_{w_j} = \lambda w - \Sigma_{i=1}^{n}\left[\left(y_i == j\right) - \frac{exp^{w_j^T * X_i}}{1+\sum_{k=1}^{K-1} exp^{w_k^T * X_i}}\right] X_i$$

$$\Rightarrow \nabla \text{E}_{w_j} = \lambda w - \Sigma_{i=1}^{n}\left[\left(y_i == j\right) - P_{w_j}\left(X_i, w\right)\right] X_i$$

To solve the optimization problem E $(-f)$, we used exactly same procedure as applied to algorithm 1.To solve the differentiation of the objective function , we implemented algorithm 2.

Algorithm 2 Logistic Regression for Multi-class Classification

Input: Train_data,Train_labels, lambda

Output: w with size d*K-1.

```
 1: E⟵ $Objective function$
 2: F⟵ $\nabla E$
 3: $\lambda$ ⟵ lambda
 4: w⟵ Initialize
 5: itr⟵0
 6: while($i < 50$)
 7:     $f_1$ ⟵ compute E at w
 8:     g⟵ compute F at w
 9:     $c_1$ ⟵ 0.5
10:     $\alpha$ ⟵ 0.005
11:     $\phi_1 \longleftarrow f_1 + c_1 * \alpha * g^t * g$
12:     $w_{new}$ =w-$\alpha * g$
13:     $\phi$ ⟵ compute E at $w_{new}$
14:     while($\phi_1 < \phi$)
            $\alpha \longleftarrow \alpha * 0.75$
            $w_{new}$ =w-$\alpha * g$
            $\phi$ ⟵ compute E at $w_{new}$
            $\phi_1 \longleftarrow f_1 + c_1 * \alpha * g^t * g$
        end while
15:     w=w-$\alpha * g$
16:     itr=itr+1
17: end while
```

After the execution of algorithm 2, we get classifiers for all classes. Invoke all these classifiers on test example. Now we have probabilities of all the classes for test example. Assign a class label which has high probability among all. Similarly apply this on all examples of test data. Now we have predicted labels for all test set examples. Our results are presented in section 8.

V. Datasets and Statistics

The details of multi-label datasets include the number of training examples, the number of test examples, the number of features, the number of labels, label density and label cardinality. Those details of four multi-label datasets are shown in Table IX.

In some applications, the number of labels of each example is small compared to total number of labels q, while in some other applications it is large. This could be a parameter that influence the performance of different multi-label classifiers. Now, we define label cardinality and label density.
Label cardinality of a dataset D is the average number of labels of the examples in D:

$$\text{Label Cardinality}=\frac{1}{m}\sum_{i=1}^{m}\left|Y_i\right|$$

where m is total number of examples in dataset D.

Label density of dataset D is the average number of labels of the examples in D divided by q:

$$\text{Label Density }=\frac{1}{m}\sum_{i=1}^{m}\frac{\left|Y_i\right|}{q}$$

where m is total number of examples and q is number of labels in dataset D.

Label cardinality is independent of the number of labels q in the classification problem. Label density takes into consideration the number of labels in the domain. Two datasets with the same label cardinality but with different number of labels may exhibit different behaviour to the multi-label classification methods.

TABLE IX. Datasets and its statistics.

Dataset	Train Examples	Test Examples	Attributes	Labels	Label Density	Label cardinality
Scene	1211	1196	294	6	0.18	1.08
Emotions	391	202	72	6	0.311	1.869
Yeast	1500	917	103	14	0.30	4.25
Tmc2007	21519	7077	30438	22	0.098	2.158
MediaMill	30993	12914	120	101	0.043	4.376
Bibtex	4880	2515	1836	159	0.015	2.402
Corel5k	4500	500	499	374	0.009	3.522
Delicious	12920	3185	500	983	0.019	19.02

VI. Evaluation Metrics

Before seeing the details of our proposed approaches for solving MLC, we first look at evaluation metrics in multi-label setting. Unlike single-label classification, multi-label setting has several evaluation metrics. Here we take some of the evaluation metrics [3] to evaluate performance of MLC methods. Assume we have multi-label training data D_r containing n examples which are of the form (X_i, Y_i) where X_i is the feature vector, $Y_i \in \{0,1\}^q$is the label vector for i^{th} class and q is the number of classes. Let $h\left(X_i\right)$ denote a multi-label classifier predicted label set for X_i and test data D_e contains m examples which are of the same form as train data D_r.

1) **Hamming Loss:** Hamming loss is one of the well used evaluation metric and it evaluates the number of labels which are wrongly predicted.

$$\text{HammingLoss}(h, D_e) = \frac{1}{q}\sum_{i=1}^{m} \parallel h\left(X_i\right) \oplus Y_i \parallel_1$$

where $\oplus$ is XOR operation.

2) **Subset Loss:** It evaluates strictly whether a classifier's predicted label set is exactly correct.

$$\text{SubsetLoss}(h, D_e) = \frac{1}{m}\sum_{i=1}^{m} I\left(h\left(X_i\right) \neq Y_i\right)$$

where $I\left(.\right)$ denotes the indicator function. i.e $I\left(l\right)$=1 if l holds otherwise 0.

3) **Micro-F1** [3]: It evaluates a classifier's label set prediction performance, which considers both micro average of prediction and recall on different classes.

$$\text{Micro-F1}(h, D_e) = \frac{2 * \sum_{i=1}^{m} \|h(X_i) \cap Y_i\|_1}{\sum_{i=1}^{m} \|h(X_i)\|_1 + \sum_{i=1}^{m} \|Y_i\|_1}$$

4) **Macro-F1** [3]: It considers average the F1 measure on the predictions of different labels.

$$\text{Macro-F1}(h, D_e) = \frac{1}{q} \sum_{k=1}^{q} \frac{2 * \sum_{i=1}^{m} h^k(X_i) Y_i^k}{\sum_{i=1}^{m} h^k(X_i) + \sum_{i=1}^{m} Y_i^k}$$

whereY_i^k is the k^{th} entry of Y_i and similarly $h^k(X_i)$ is k^{th} entry of $h(X_i)$.

There also exist several other metrics. The first two metrics out of above four describe loss of prediction function. So we can say that the lesser the value, the better the performance. But the other two describe accuracy. So we can say that the larger the value, the better the performance.

VII. MULTI-LABEL CLASSIFICATION

In multi-label setting, examples are associated with several classes. Formally we define the format of dataset D. It consist of n examples, which are of the form (X_i, Y_i) for i=1 to n where X_i is a feature vector of i^{th} example and $Y_i \in \{0,1\}^q$ and q is number of total classes in a setting.

Many applications such as protein function classification, music categorization, semantic scene classification etc., require multi-label classification methods. There exist a lot of methods and evaluation metics for solving multi-label classification problem such as hamming loss, subset loss. Most of the existing methods for solving multi-label classification are deterministic in nature. We are concentrating on probabilistic models.

We develop two new probabilistic approaches to solve MLC. One of the two approaches is based on logistic regression. This approach is similar to Binary Relevance method. The training phase of this approach is same as the training phase of binary relevance method, but testing phase is different. The other approach is based on the idea of grouping related labels. This method trains one classifier for each group and the corresponding label is called as group representative. Predict other labels based on the predicted labels of group representative. The relations between the labels are found using the concept of association rule mining.

A. Method using Partial Information (MPI)

While solving multi-label classification problem, one may face two main problems. They are, considering label correlation and how many number of labels to be assigned for each test example. We are focusing on the second problem by considering independence among labels. To choose how many number of labels to be assigned for test example, we propose a method which can be implemented using multi-class logistic regression and binary logistic regression.

We start with our proposed method, which is based on multi-class logistic regression. The training process of this method is same as the training process of binary relevance method i.e., it learns binary classifier for each class. The training process of our method is shown in algorithm 5.

Algorithm 3 Training process of method 1

```
Input: Train_data,Train_labels
Output: Classifiers for all classes.
1: For each class $i$
       $W_i \longleftarrow$ Train a binary classifier using data matrix and $i^{th}$ label vector.
2: end for
```

There exist three phases in the testing process of this method. The objective of the first phase is to find out the number of labels to be assigned for test examples. The objective of second phase is to apply binary logistic regression on multi-label data to get probabilities of all classes for each test example. The objective of third phase is to assign set of labels to each test example which is based on the results of first two phases.

Algorithm 4 Testing process of method 1.

```
Input: All trained classifiers W, Test_data
Output: Predicted labels of Test_data
1: T $\longleftarrow$ design a vector of length equal to number of train examples and T$_i$ tells how many number of labels assigned with $i^{th}$ train example
2: Train a multi-class classifier on data matrix of train set and vector T using multi-class logistic regression
3: Z $\longleftarrow$ multi-class classifier obtained in above step.
4: V $\longleftarrow$ vector of length equal to number of test examples and V$_i$ tells predicted number of labels assigned with $i^{th}$ test example by invoking Z on $i^{th}$ test example
5: for each test example $t_k$
       i) Invoke all trained classifiers W and get probabilities for all classes
       ii)Arrange all classes in ascending order of probabilities
       iii)$d \leftarrow V_k$
       iv)Assign top $d$ classes to test example $t_k$
```

In first phase, the first step is to create a new vector of length equal to number of training examples in which i^{th} component tells how many number of labels are assigned to i^{th} training example. Train a multi-class classifier on train data and vector created in first step using multi-class logistic regression. It results in classifiers for all classes. Invoke these classifiers on test data to get predicted labels. For example, predicted label of j^{th} test example is 3, which means that j^{th} test example should be tagged with three labels.

In second phase, apply binary logistic regression on actual training data to get probabilities of all class labels for each test example. In third phase, assign labels to each test example using probabilities obtained in second phase. Result of first phase tells that the how many of labels are to be assigned. For example, j^{th} test example is tagged with d in first phase, which means that j^{th} test example should be tagged with d labels. For each test example, arrange all classes in descending order of probabilities obtained in second phase and tag first d classes to it. Testing phase of this method is explained in algorithm 4.

In the above method, we used classification twice. One classification is for getting number of labels to be assigned for test examples. Other classification is used to get the probability values for each test example. Since this method involves classification twice, time taken by these methods is higher than existing approaches like BR method. These methods are well suitable for small datasets. The results of above implementation are shown in section 8.

B. Method using Association Rules (MAR)

In binary relevance and many other methods, binary classifiers are learnt for each class. But in multi-label setting, many of the labels are either positively related or negatively related. This method is based on the correlation information between the labels. The basic idea in this method is group the correlated labels together. From each group, select one label which is called as group representative. Train only those group representatives using train data. Predict all other classes using the predicted values of corresponding group representative.

Before going to see the details of our method, let us discuss basic definitions in association rule theory. Let $I = \{i_1, i_2, ..., i_l\}$ be the set of all items in a data and $T = \{t_1, t_2, ..., t_K\}$ be the set all transactions. Each transaction contains a subset of items chosen from I. Collection of one or more items is called as *itemset*. The number of transactions that contain a particular itemset is called as support count of that itemset. Mathematically, the support count of itemset X, $\sigma(X)$, can be expressed as follows.

$$\sigma(X) = \mid \{t_i | X \subseteq t_i, t_i \in T\} \mid \quad (7)$$

Association Rule: An association rule is an implication expression of the form $X \rightarrow Y$, where X and Y are disjoint item sets, while confidence determines how frequently items in Y appear in transactions that contain X. The support and confidence of rule $X \rightarrow Y$ is shown as $s(X \rightarrow Y)$ and $c(X \rightarrow Y)$ respectively. Mathematical definitions are as follows.

$$s(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{K} \quad (8)$$

$$c(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)} \quad (9)$$

We used this theory in our method by adding some tolerance to confidence values i.e., if the confidence value of association rule involving two classes is in certain range then we merge those two classes in one group. In our method, if the particular class is present in atleast 90% of training examples then we directly predict that class for all test instances as one. Similarly, if atmost 10% of training examples have particular class then we predict that class for all test instances as zero.

Algorithm 5 Training Process of New Approach

```
Input: Train_data D, label set L.
Output: A group of binary classifiers.
 1: $ncl$ ⟵ number of classes in ML setting.
 2: ConfMat ⟵ Create a matrix of $ncl \times ncl$ dimension and fill the matrix with the following rule, ConfMat$(i, j) = c(y_i \rightarrow y_j)$
 3: IConfMat ⟵ Inverse of matrix ConfMat.
 4: $SumMat$ ⟵ ConfMat+IConfMat
 5: Direct0 ⟵ zero vector of $ncl$ length
 6: Direct1 ⟵ zero vector of $ncl$ length
 7: for each class $i$
 8:         if atleast 90% of examples contains $i$ then
 9:             Direct1$(i)$=1
10:         if atmost 10% of examples contains $i$ then
11:             Direct0$(i)$=1
12: end for
13: $SumTable$ ⟵ Initialize a four column matrix
14: for each pair$(i, j)$
15:         $val \leftarrow SumMat(i, j)$
16:         if $val$ >1.6 OR $val$ <0.4 then
17:             Add $[i\ j\ val\ \ (2 - val)]$ to SumTable
18:         end if
19: end for.
20: $SST$ ⟵ Sort SumTable in decreasing order of fourth column value
21: for each row in $SST$
22:     if $(SumMat(i, j) > 1.6)$ then
23:         i and j are positively correlated and group them
24:     else
25:         i and j are negatively correlated and group them
26:     end if
27: end for
28: $G$ ⟵ Set of all group representatives
29: for i=1 to ncl
30:     if$(i \in G)$
31:         Learn classifier for label i using LR
32:     end if
33: end for
```

The first step in this method is to create a matrix of dimension $ncl \times ncl$, where ncl is the number of classes in multi-label setting. Name the matrix as ConfMat. In ConfMat (i, j)th entry tells the value of $c(y_i \rightarrow y_j)$. We get all these values from label matrix of training set. Now, create another matrix with same dimension which is transpose of ConfMat, named as IConfMat. In IConfMat, (i, j)th entry tells the value of $c(y_j \rightarrow y_i)$. Now, sum those matrices and store the result in SumMat. Now, observe the value in each entry of this matrix. Let x be the value in (i, j)th entry in SumMat.

1) If the value of x is greater than 1.6 i.e., $c(y_i \rightarrow y_j) + c(y_j \rightarrow y_i)$ is $>$ 1.6 then we place y_i *and* y_j in one group as they are positively correlated.
2) If the value of x is lesser than 0.4 i.e., $c(y_i \rightarrow y_j) + c(y_j \rightarrow y_i)$ is $<$ 0.4 then we place y_i *and* y_j in one group as they are negatively correlated.
3) If the value of x is greater than 0.4 or lesser than 1.6 then there is no significant relation between y_i *and* y_j.

Now, we have label groups. In each group, the labels are either positively correlated or negatively correlated. Now, select one label from each group and call it as group representative. Select the group representative by the following rule. Choose the label which has maximum number of correlation with the other group members. Now create set G, which is set all group representatives. Train classifiers for only those group representatives. The complete training algorithm is shown in algorithm 5.

We used logistic regression to learn classifier. The parameters used in logistic regression are same as those explained in LR-Binary classification section. In this method, classifiers are learnt only for group representatives. Now, we discuss about testing phase of our approach.

Given a test instance $test_i$, invoke all classifiers learn in training phase on it and predict corresponding labels. Now we have predicted values of some classes which are group representatives. Now we need to predict the labels of other classes also. For that we need to find its group representative and the relationship between class and its group representative.

Algorithm 6 Testing Process Of New Approach

```
Input: Test_data $T$, All trained classifiers.
Output: Predicted Label set.
1: for j=1 to $| L |$
2:     if$(Direct0\,(j) == 1)$
3:         Predict $j$th label as zero vector.
4:     else if$(Direct1\,(j) == 1)$
5:         Predict $j$th label as one vector.
6:     else if $(j \in G)$
7:         Predict the label vector using corresponding trained classifier.
8:     end if
9: end for
10: Predict all other labels using predicted labels of corresponding group representative.
```

If they are positively correlated then we predict the label same as predicted value of its group representative. If they are negatively correlated then we predict the label as reverse of predicted value of its group representative. The complete testing phase of our method is explained in algorithm 6.

VIII. Experimental Results

We used eight different datasets to compare the performance of our proposed approaches with the binary relevance method. The names of those datasets and statistics of those datasets are shown in Table IX. All those datasets are arranged in increasing order of number of labels in it. We also used four different performance metrics to compare the results of our method with BR method.

Now, let us look at the implementation details of BR method. In BR implementation, we used logistic regression as base classifier. The parameters used in logistic regression are already explained in section 3. The table X shows the results obtained with different methods. The third column in that table gives the result of BR method for several datasets.

Now, let us discuss the implementation details of MPI method. We have already seen that the training phase of this approach is same as the training phase of BR method. In this method also, we used logistic regression as base classifier. We also used logistic regression for multi-class classification in the testing phase of this approach. The parameters used in this LR are shown in section 4. Fourth column of table X shows the result of this method.

Now, let us look at the implementation details of MAR method. In this method also, we used logistic regression as base classifier. The threshold confidence values for merging two labels are shown in algorithm 5. Fifth column of table X represents the result of MAR method for several datasets.

Now, let us compare the results of all the three methods. The best result for each metric is shown in bold. If we analyse the datasets from their statistics, we can observe that MPR method performs better for small datasets such as scene ($2407\ examples\ with\ 294\ dimension$) and emotions ($593\ examples\ with\ 72\ dimension$). Also we can observe that MAR performs well in many datasets where the number of labels is high like corel5k ($374\ labels$) and delicious ($983\ labels$).

Let us define a term called *maximum label cardinality* of dataset. It is defined as the maximum number of labels assigned to any train example in a dataset. In MPI, apart from base classifier we also used LR for multi-class classification. In this muti-class logistic regression setting, the number of classes is equal to *maximum label cardinality* of dataset. The MPI method should train these many number of classes apart from classifier for each label. The total number of classifiers learnt in all three methods are shown in table XI.

TABLE X. Results of BR, MPI and MAR methods.

Dataset	Metric	Binary Relevance	MPI $(method1)$	MAR $(method2)$
Scene	Hamming Loss	0.1219	**0.0992**	0.1219
	Subset Loss	0.5602	**0.3311**	0.5602
	Micro-F1 score	0.6160	**0.7152**	0.6160
	Macro-F1 score	0.6137	**0.7221**	0.6137
Emotions	Hamming Loss	0.2987	0.3176	**0.2987**
	Subset Loss	0.9158	**0.8713**	0.9158
	Micro-F1 score	0.3170	**0.6050**	0.3170
	Macro-F1 score	0.2440	**0.5755**	0.2440
Yeast	Hamming Loss	0.4284	**0.3065**	0.3335
	Subset Loss	0.9346	**0.9138**	0.9389
	Micro-F1 score	0.4695	0.4760	**0.5138**
	Macro-F1 score	**0.4286**	0.3972	0.4205
Tmc2007	Hamming Loss	0.0783	0.0849	**0.0645**
	Subset Loss	0.8679	0.8467	**0.7975**
	Micro-F1 score	0.4328	0.5104	**0.5650**
	Macro-F1 score	0.0974	0.1707	**0.1918**
Mediamill	Hamming Loss	0.0337	0.0407	**0.0305**
	Subset Loss	0.9431	0.9888	**0.9365**
	Micro-F1 score	0.4837	**0.5423**	0.5001
	Macro-F1 score	0.0272	**0.0555**	0.0332
Bibtex	Hamming Loss	0.0139	0.0147	**0.0128**
	Subset Loss	0.9233	**0.8060**	0.8561
	Micro-F1 score	0.1944	**0.4415**	0.3797
	Macro-F1 score	0.0304	**0.2884**	0.1758
Corel5k	Hamming Loss	0.0096	0.012	**0.0095**
	Subset Loss	0.9980	0.9980	**0.9980**
	Micro-F1 score	0.0180	0.0546	**0.0732**
	Macro-F1 score	0.0006	0.0007	**0.0021**
Delicious	Hamming Loss	0.0216	0.0451	**0.0201**
	Subset Loss	0.9987	**0.9541**	0.9800
	Micro-F1 score	0.2264	0.2154	**0.2275**
	Macro-F1 score	0.0503	**0.0725**	0.0148

Now, let us discuss the advantages of MAR approach over binary relevance method and MPI method. In binary

TABLE XI. NUMBER OF CLASSIFIERS LEARNT IN DIFFERENT METHODS.

Dataset	**Binary Relevance**	**MPI** $(method1)$	**MAR** $(method2)$
Scene	6	6+3	**6**
Emotions	6	6+3	**6**
Yeast	14	14+11	**9**
tmc2007	22	22+10	**7**
MediaMill	101	101+18	**8**
Bibtex	159	159+28	**96**
Corel5k	374	374+5	**9**
delicious	983	983+25	**37**

relevance method, one classifier is learned for each class. For example, consider mediamill dataset. It has 101 classes. Binary relevance method trains 101 classifiers for mediamill dataset. But the MAR method learns less or equal number of classifiers as number of classes in dataset, based on relation between the classes. If more classes are related then MAR learns very less number of classifiers as compare to binary relevance method. For example, mediamill dataset has correlated$(positively\ or\ negatively)$ labels. It has total 101 classes. MAR method learns only eight classifiers as there are eight different groups. At the same time MPI method trains 119 classifiers. So, we can say that time taken by MAR method is lesser or equal than MPI method and BR method and also we can say that the time taken by MPI method is greater than BR method. This is the disadvantage of MPI method and advantage of MAR method.

IX. CONCLUSION AND FUTURE WORK

This paper discussed about some of the existing approaches for multi-label classification. We also performed experiments on logistic regression for both binary and multi-class classification. We proposed two new approaches to solve multi-label classification. They are probabilistic in nature. One method uses partial information about labels, which is explained earlier. The other method uses mutual information between the labels. The results of both the methods presented in previous section.

In this paper, we used pairwise information between the labels. Here, there is a scope of future work. In the future, we intend to use group wise information among labels instead of pairwise relation. We also intend to perform more experiments with more datasets and methods.

REFERENCES

[1] Yi Zhang and Jeff Schneider. A Composite Likelihood View for Multi-Label Classification. *In proceedings of the 15th international conference on AISTATS 2012, La Palma, Canary Islands.*

[2] Wei Bian, Bo Xie and Dacheng Tao. CorrLog:Correlated Logistic Models for Joint Prediction Of Multiple Labels. *Appearing in proceedings of the 15th International conference on AISTATS 2012, La Palma, Canary Islands.*

[3] Jesse Read, Bernhard Pfahringer, Geoff Holmes and Eibe Frank. Classifier Chains for Multi-label Classification.

[4] Xiangnan Kong, Xiaoxiao Shi and Philip S.Yu. Multi-label Collective Classification. *Xiangnan Kong, Xiaoxiao Shi and Philip S.Yu.*

[5] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Mining multi-label data. *In Data Mining and Knowledge Discovery Handbook, pages 667685. 2010.*

[6] G. Tsoumakas and I. Vlahavas. Random k-labelsets:an ensemble method for multilabel classification. *In ECML 07: Proceedings of the 18th European conference on Machine Learning, pages 406417, Berlin,Heidelberg, 2007.*

[7] Grigorious Tsoumakas and Ioannis Katakis. Multi-label Classification: An Overview.

[8] Zhang, M.L., Zhou, Z.H.: Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition 40 (2007) 20382048.*

[9] Akinori Fujino and Hideki Isozaki. Multi-label classification using logistic regression models for NTCIR-7 patent mining task. *Proceedings of NITCIR-7 workshop meeting, 2008, Tokyo, Japan.*

[10] Jianwen Xie, Jianhua Wu and Qingquan Qian. Feature Selection Algorithm Based on Association Rules Mining Method. *2009 Eigth IEEE International Conference on Computer and Information Science.*

[11] Francisco Charte, Antonio Rivera, Maria Jose del Jesus and Francisco Herrera. Improving Multi-Label Classifiers via Label Reduction with Association Rules. *HAIS 2012, Part II, LNCS 7209, pp.188-199, 2012.*

[12] Grigorios Tsoumakas, Ioannis Katakis and Ioannis Vlahavas. Effective and Efficient Multilabel Classification in Domains with Large Number of Labels.

[13] Grigorios Tsoumakas, M.L. Zhang, Z.H. Zhou. Introduction to the special issue on learning from multi-label data. *Mach Learn(2012) 88:1-4*

[14] Pang-Ning Tan, Vipin Kumar, Michael Steinbach. *Introduction to Data Mining.*