

## RESEARCH ARTICLE

# Efficient label ordering for improving multi-label classifier chain accuracy

Tariq Ali<sup>1\*</sup> and Sohail Asghar<sup>2</sup>

<sup>1</sup> Department of Computer Science, Abasyn University, Islamabad, Pakistan.

<sup>2</sup> Department of Computer Science, COMSAT University Islamabad (CU), Islamabad, Pakistan.

Submitted: 06 November 2017; Revised: 11 June 2018; Accepted: 17 August 2018

**Abstract:** Classifier chain (CC) algorithms have been introduced for multi label classification predictions in recent years. The accuracy of these algorithms is considered better than the other state-of-the-art algorithms in this domain. In addition to accuracy, an effort is made to improve the complexity of the algorithms in order to predict an optimal order in which the binary classifiers are executed. Existing label ordering algorithms are executed twice, once for the generation of label ordering and another time for improving classifier chain accuracy with predicted order. In this paper, we discuss the current chain classifier algorithms and their comparison in terms of both accuracy and execution time. Moreover, we have introduced a new Label Ordering for Classifier Chain (LOCC), which exploits the semantic relationships among the labels of a dataset. The predicted label's order is computed without the execution of the classification algorithm. The semantic relations among the labels are analysed and an order is generated, which is fed to a classifier chain algorithm. The proposed algorithm is better in terms of accuracy and computational time than the available classifier chain algorithms.

**Keywords:** Classifier chain algorithms, data mining, digital libraries, multi-label classification, text classification.

## INTRODUCTION

In a single label classification, only a single label is predicted, while in multi label classification multiple labels are predicted. Both classification schemes are shown in Figure 1 and the focus of this paper is on multi label classification. Multi label classification can be modelled as shown in equation (1). Furthermore,

the binary relevance (BR) method (Luaces *et al.*, 2012) is the simplest way for multi label classification in which the problem with  $L$  labels is decomposed in  $|L|$  binary classifiers problem. For each classifier, a function  $h(x)$  is learnt for the input instance features as modelled in equations (2) to (4), for the example given in Figure 1. The multi label output is generated by combining the prediction of all the binary classifiers as a predicted list. BR expands the dataset into  $|L|$  problems with linear complexity, but the major problem is that it ignores the label dependences.

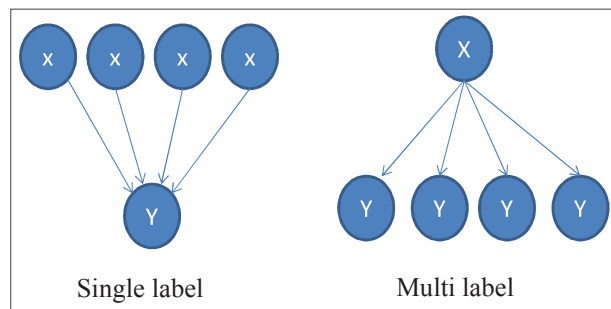



Figure 1: Single and multi label classification

$$\hat{y}_j = h_j(x) = \arg \max p(y_j | x) \quad \text{for index } j = 1, \dots, L \quad \dots(01)$$

\* Corresponding author (tariq.ali@uaar.edu.pk ;  <https://orcid.org/0000-0002-4974-1569>)



In equation (1),  $\hat{y}_j$  is the predicted  $J^{\text{th}}$  label for input features set  $x$ .  $h(x)$  is the binary classifier for which maximum a posteriori estimation is considered. For the above mentioned example in Figure 1 (having four possible labels), equation (1) can be modelled as given in equations (2) and (3). Each label is either assigned to an instance or not (represented as 0 or 1), the output is the amalgamation of all the label predictions. The BR method for the above mentioned example can generically be modelled as given in equation (4), in which a single binary classifier is trained for each label.

$$\hat{y} = h(x) = \left[ \hat{y}_1, \dots, \hat{y}_4 \right] \quad \dots(02)$$

$$= \left[ \arg \max_{y_1 \in \{0,1\}} p(y_1 | x), \dots, \arg \max_{y_4 \in \{0,1\}} p(y_4 | x) \right] \quad \dots(03)$$

$$= [f_1(x), \dots, f_L(x)] = f(W^T x) \quad \dots(04)$$

The assumption of label dependence, which is modelled as in equation (5) cannot be ignored for all the times. For example in the film genre classification, a romance movie does not imply that it cannot be a horror movie. The preliminary challenge in the multi label classification is to identify the label correlations, which exist among the labels in a dataset. Although identification of these dependencies has improved the classification accuracy, such as introducing the classifier chain (CC) model (Read *et al.*, 2009). The author introduced a chain of classifiers in which the output of the earlier binary classifier is fed along with the feature attribute ( $x$ ) to the successive binary classifiers as modelled in equation (6). To predict the label  $y_j$ , the previously predicted outputs ( $y_1, \dots, y_{j-1}$ ) are used with the input features. The label dependencies, thus improves the prediction of the successive binary classifiers (Read *et al.*, 2004; Senge *et al.*, 2014).

$$p(y | x) \propto p(x) \prod_{j=1}^L p(y_j | x) \quad \dots(05)$$

$$p(y | x) = \prod_{j=1}^L p(y_j | x, y_1, \dots, y_{j-1}) \quad \dots(06)$$

And the predicted label will be:

$$\hat{y} = \arg \max_{y \in \{0,1\}^L} p(y | x) \quad \dots(07)$$

The input of a preceding classifier is used by the successive classifier during multi label classification. The concept is demonstrated with an example in Figure 2 having three instances with input feature ( $x^1, x^2$  and  $x^3$ ) and modelled mathematically using equations (8) to (11). Likewise, CC uses the basic concept of BR. Nonetheless, it includes the predictions as a feature vector of the input. The binary classifier can be used for the prediction of individual label after transformation to binary problems from the multi label problem to some extent.

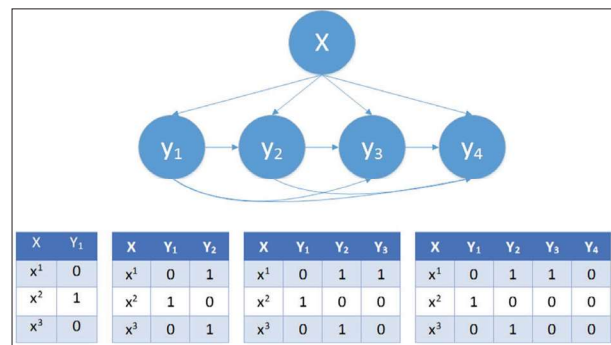


Figure 2: Classifier chain (CC)

In the above example, labels are predicted in a random order. In the example presented in Figure 2, first, the label  $Y_1$  will be predicted as defined in equation (8). The output of label  $Y_1$  will be used with input feature vector  $x$  for label  $Y_2$  prediction as formulated in equation (9). Label  $Y_3$  and  $Y_4$  will also be predicted in the same way as constructed in equations (10) to (11). The final output  $\hat{y} = [\hat{y}_1, \dots, \hat{y}_L]$  will be returned to the user.

$$\hat{y}_1 = h_1(\tilde{x}) \quad \dots(08)$$

$$\hat{y}_2 = h_2(\tilde{x}, \hat{y}_1) \quad \dots(09)$$

$$\hat{y}_3 = h_3(\tilde{x}, \hat{y}_1, \hat{y}_2) \quad \dots(10)$$

$$\hat{y}_4 = h_4(\tilde{x}, \hat{y}_1, \hat{y}_2, \hat{y}_3) \quad \dots(11)$$

In CC there is no strategy in the order of labels in which they are evaluated, i.e., which label to predict first and which one on the second level. Usually, the greedy strategy for CC is adopted, where label ordering is ignored. In this strategy, the error propagates in the chain, i.e., mis-classification at initial level propagates in the next level (Senge *et al.*, 2013; 2014; Montañes, *et al.*, 2014). The working out of an optimised label sequence is computationally inept due to higher search space of possible  $q!$  permutations, where  $q$  is the total number of labels available. Literature shows that there are different ways in which possible combinations of labels are evaluated. These techniques are used to minimise the error propagation in the chain. The simplest strategy is the greedy search, which selects the maximum probability at each stage. In this strategy, the error in early stages propagates in the chain (Mena *et al.*, 2016). The alternate way is the exhaustive search (Cheng *et al.*, 2010) in which all possible outputs are considered at later stages. The problem with this strategy is expensive in terms of computational resources as it checks all the possible paths down the tree. The technique is demonstrated equally in Figure 2 with arrows demonstrating the order. There is  $\epsilon$  approximate algorithm (Dembczyński *et al.*, 2012), beam search (Kumar *et al.*, 2012; 2013) and Monte Carlo sampling (Read *et al.*, 2004), which eliminates a few paths down the tree. In  $\epsilon$  approximation methods, those children of a node are selected whose joint conditional probability is more than the threshold value and rest are discarded. In beam search algorithms more than one path in the probabilistic tree is explored. This method considers a parameter  $b$ , which is the width of the beam for considering the paths in the tree at each level. Monte Carlo is a technique based on repeating random sampling. A comprehensive discussion and comparison based on maximum accuracy and computational efficiency is available in literature (Cheng *et al.*, 2010; Dembczyński *et al.*, 2012; Kumar *et al.*, 2012; 2013; Mena *et al.*, 2016; 2017).

To improve the two major drawbacks of CC, i.e., label ordering and error propagation, a technique called ordered classifier chains (OCC) based on classifier accuracy is proposed (Keikha & Hashemi, 2016). In this technique the multi label problem is transformed into a binary classifier problem. In addition, sequential minimal optimization (SMO) is used as a base classifier in this technique. After that, the classifier is executed and per label classifier accuracy is arranged in descending order. The algorithm of OCC is given as Algorithm 1 (Keikha & Hashemi., 2016). The generated order is fed to CC classifier, which improves the accuracy of the CC algorithm as compared to random order given to CC. The major problem with this technique is the execution of the algorithm, which takes double processing time.

This technique is tested on 10 benchmark datasets and is concluded that it is better than CC in terms of accuracy, and better than ensemble of classifier chain (ECC) in terms of computational complexity (Keikha & Hashemi, 2016).

---

**Algorithm 1:** OCC's finding chain phase for training set D and label set L (Keikha & Hashemi, 2016)

---

```

D = {(Xk, Lk), k = 1, ..., K}
L = {lj: j = 1, ..., M}
for j=1, ..., M do
  D' = {}
  D' ← D' ∪ (Y, lj)
  hj: D' → lj ∈ {0, 1}
  lj∧ ← hj(X∧)
  > A is an array for accuracies
  A(l, j) = Accuracy (hj)
end for
Rank (A) descendingly

```

---

The technique label priority chain using classifier chain (LPC-CC) (Soonsiripanichkul *et al.*, 2016) is proposed, which works similar to OCC with naive Bayes as a base classifier. The LPC-CC also used BR for label sequencing priority as given in Algorithm 2 (Soonsiripanichkul *et al.*, 2016) and CC classifier with naive Bayes is used for finding label domination. The results are tested on three datasets, i.e., yeast, emotions and collection of car sales records from an automotive company in Thailand.

---

**Algorithm 2:** Label ordering (LP)  
(Soonsiripanichkul *et al.*, 2016)

---

```

x # attribute vector
y # set of labels
q # range of y
N # number of instances
Input: Dataset M composed of feature x and set of labels y. There are I instances
Output: New dataset that order of labels is recorded
1: For j=1 to q Do
2: binaryTransformation(x,y)
3: ŷj ← BR(x,yj)
4: Sort yj from Acc (x,ŷ)
5: Return newM = {(xp, y1-), ..., (xN, yN-)}

```

---

The genetic algorithm is also used for generating label order (Goncalves *et al.*, 2013). In this approach the training dataset is further divided into two parts; building and validating. Label order optimisation is done on the

building part of the training set and then fitness function is evaluated on the validation set. Final accuracy is obtained using the test set with obtaining label ordering. All labels are represented in a sequence, which formulates initial population for the genetic algorithm. Accuracy [Equation (12)], exact match equation (13) and Hamming Loss equation (14) measures are amalgamated to formulate the fitness function equation (15). The initial population is evaluated using the CC algorithm, in case a sequence does not give better results than cross over and mutation is applied. The final optimal order is evaluated with the test dataset. This approach has a high execution cost as each time every order is evaluated on a subset of the dataset. The approach is evaluated on a set of benchmark datasets with comparison of the results on the accuracy, exact match and hamming loss.

$$ACC = \frac{1}{n} \sum_{i=1}^n \frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|} \quad \dots(12)$$

$$EM = \frac{1}{n} \sum_{i=1}^n I(Y_i = Z_i) \quad \dots(13)$$

$$HL = \frac{1}{n} \sum_{i=1}^n \frac{|Y_i \Delta Z_i|}{q} \quad \dots(14)$$

$$\text{Fitness (i)} = \frac{EM + ACC + (1 - HL)}{3} \quad \dots(15)$$

In all of the above three approaches the order is generated after the execution of the classification algorithm. None of the approaches exploit the semantic relationship that resides inside the dataset among the labels. To improve the accuracy with optimum order and less execution time, we have devised an algorithm that explores the relationship between the labels of a dataset without classifier execution. We have compared the results on 9 benchmark datasets with the OCC and LPC-CC techniques.

## METHODOLOGY

The problem is to formulate an organisation for a dataset  $D$  having  $L$  labels, which gives better accuracy results. The existing techniques are costly in terms of computation time, in which all the possible ordering are measured. There is a need to semantically analyse the dataset and predict a chain without taxonomy and algorithm execution. Nonetheless, our algorithm exploits the chain order by exploring the semantic relationship between the labels in the dataset. Initially the frequency of labels and the co-occurrence that exist among the labels have been

measured. Frequency and co-occurrence among labels is computed as  $L*L$  matrix using equation (16).

$$M(L_i, L_j) = \begin{cases} |L_i|, & i = j \\ |L_i, L_j|, & i \neq j \end{cases} \quad \dots(16)$$

In the proposed LOCC algorithm (Algorithm 3),  $\zeta$  will hold the chain order, which is the predicted output. At initialisation phase, the category with a high number of cardinality using the matrix  $M$  as defined by equation (16) is selected. The label with maximum cardinality will be added to the selected chain order  $\zeta$  and the selected label is removed from the original label set. In the second phase, a loop is iterated for the remaining labels. The cardinality of the remaining labels will be computed each time with the label selected at the previous level using equation (16). In the same way, the maximum cardinality between the previously selected label and the label in the remaining label set  $L$  will be added to the label chain  $\zeta$  and will be removed from the original label set  $L$ . The loop will iterate until one label is left in the original label set; the last label left will be appended to the predicted label set.

---

### Algorithm 3: LOCC label ordering for classifier chain

---

*Input:*  $D$  (dataset),  $L$ (label set)

*Output:*  $C$  (ordered label set)

$\zeta \leftarrow \emptyset$

$Max = 0$

**For each** label  $\ell \in L$  **do**

*Compute*  $|\ell|$

**If**  $(|\ell| > max)$  **then**

$Max \leftarrow |\ell|$

$S \leftarrow \ell$

**End if**

$\zeta \leftarrow \zeta \cup [I]$

$L \leftarrow L - [I]$

**End for**

**While**  $(|L| \neq 1)$  **do**

$Max = 0$

**For each** label  $\ell \in L$  **do**

$c \leftarrow sim(S, \ell)$  using Eq.16

**If**  $(max < c)$  **do**

$Then$   $max \leftarrow c$

$S \leftarrow \ell$

**End if**

**End for**

$\zeta \leftarrow \zeta \cup [I]$

$L \leftarrow L - [S]$

**End while**

**Return**  $\zeta$

---

The working of the proposed algorithm is demonstrated on the Emotions dataset in Figure 3. The dataset contains 6 labels which are relaxing calm (RC), angry aggressive (AA), amazed surprised (AS), sad lonely (SL), happy pleased (HP) and quite still (QS). The figure contains co-occurrence matrix of all labels available in the dataset using equation (16). L set is initialised with all available labels in the dataset and C set as empty. In the first part of the algorithm RC is selected having maximum

cardinality, which is removed from L set and added to the C set. The algorithm then iterates until the elements in the L set are more than one. In the first iteration, QS label is added to the chain C as having the maximum co-occurrence with the previous selected label RC. In the second iteration, SL label is added to the chain as having maximum co-occurrence with QS. The process continues until all labels are removed from the L set and added to the C set. Finally the algorithm returns the C set as an

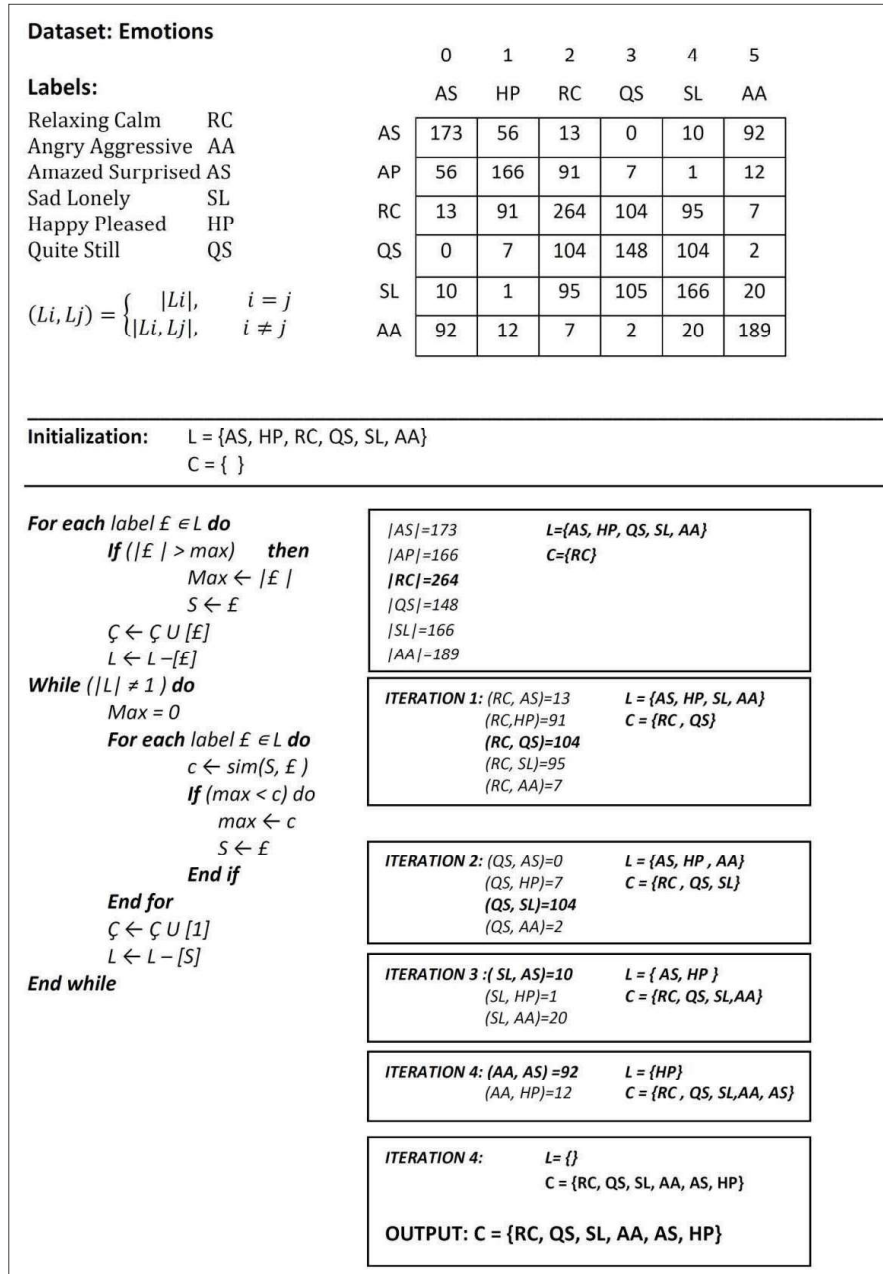


Figure 3: Stepwise execution of the LOCC algorithm over emotions dataset

output. The ordered chain of label  $C$  is given as input to the well-known CC algorithm. The results are compared with the original BR (Luaces *et al.*, 2012), CC (Read *et al.*, 2011), OCC (Keikha & Hashemi, 2016) and LPC-CC (Soonsiripanichkul *et al.*, 2016) classifiers.

## RESULTS AND DISCUSSION

In this section we have briefly discussed about the dataset used in the comparison of the proposed algorithm with existing algorithms. The evaluation criteria for the comparison and the achieved results along with their analysis is also discussed.

The experimental results are obtained on Intel core i3, with 1.7 GHz processor and 4 GB RAM. NetBeans IDE version 8.1 with Java Development Kit (JDK) version 1.8 is used. The algorithm was implemented in Java and the generated order was fed to the CC algorithm in MEKA version 1.9.0. For all the experiments default parameters of MEKA are used.

The performance of the techniques was evaluated in terms of accuracy, hamming loss, F1 micro averaged by the labels and running time. Nine supervised datasets were selected. Among these, 7 were benchmarked datasets that were taken from the Mulan Library (Tsoumakas *et al.*, 2011). Three datasets were from the scientific research paper taken from J.UCS (Afzal *et al.*, 2009) and ACM (Santos *et al.*, 2009). The J.UCS and ACM datasets were used for multi label scientific document classification in literature (Ali *et al.*, 2013; Sajid *et al.*, 2016).

In the experimental section, we have used nine standard datasets given in Table 1. These datasets were categorised on the basis of, number of labels, number of features, dataset size, domain, and cardinality.

Moreover, we have used four measures from multi label classification literature for comparison. The comparison parameters for multi label classification are different from the single label classification. Besides, the evaluation measures for multi label classification are categorised into label based evaluation and example based evaluation. We have used hamming loss [equation (17) and accuracy equation (18)] from example based evaluation and F1 macro averaged by label [equation (19) from label based evaluation measures. We have also compared the time taken by each technique for the test instances. A test set  $|D|=\{(X, L_i) \mid 1 \leq i \leq |D|\}$  was used in all three equations, where  $L_i$  is the set of true/actual labels, and  $E_i$  represents the predicted set of labels.  $L_i \Delta E_i$  represents the symmetric difference between the actual and predicted labels. Hamming loss encounters the prediction (incorrect label was assigned to an instance) and omission errors (correct label for an instance is not predicted). The symmetric difference is divided by  $|L_i|$  to get normalised values between 0 and 1. In computing the accuracy, the number of instances having the same predicted and actual labels was divided by the total number of instances. F1 macro is computed in terms of precision, recall and total number of labels  $|L_i|$ . Precision and recall are defined in equations (20) to (21) respectively.

$$\text{Hamming loss} = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{L_i \Delta E_i}{|L_i|} \quad \dots(17)$$

$$\text{Accuracy} = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|L_i \wedge E_i|}{|L_i \vee E_i|} \quad \dots(18)$$

$$\text{F-measure macro} = \frac{1}{|L|} \sum_{i=1}^{|D|} \frac{|2 * P_i * R_i|}{|P_i + R_i|} \quad \dots(19)$$

**Table 1:** Datasets description

Dataset	No. of labels	No. of features	Dataset size	Domain	Cardinality
Flags	7	19	194	Image	3.33
Yeast	14	103	2,417	Biology	4.24
GenBase	27	1,185	662	Biology	1.25
Emotions	6	72	593	Music	1.87
Scene	6	294	2,407	Image	1.07
Enron	53	1,001	1,702	Text	3.38
J.UCS	13	3,951	1,112	Papers	1.63
J.UCS 5Cat	5	3,576	974	Papers	1.23
ACM	11	16,255	31,403	Papers	0.75

$$P_i = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|L_i \Delta E_i|}{|L_i|} \dots(20)$$

$$R_i = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|L_i \Delta E_i|}{|L_i|} \dots(21)$$

These parameters are considered as standard parameters in the existing literature. The efficiency of the technique can be determined by the resulting higher values for accuracy and F measure and lower values for hamming loss and total time. By experimental results, we have achieved a low value for total time, which proves our claim of efficiency. The results obtained are the average run of 10 times.

We have compared our proposed technique with two standard techniques (BR and CC) and two proposed existing techniques with label ordering (OCC and LPC-CC). The results were obtained for BR, CC, existing (OCC) and proposed (LOCC) using SMO as the base classifier and BR, CC, another existing (LPC-CC) and proposed (LOCC) with Naive Bayes as the base classifier. One existing technique (OCC) is with base classifier SMO and the other existing technique (LPC-CC) is with base classifier Naive Bayes. The measure accuracy and F macro averaged by labels are better for higher values. The hamming loss and total time are better for smaller values. Our claim is pretty near to improved accuracy with less time. The results obtained were evaluated for statistically significant differences using the Friedman test (Friedman, 1937). Friedman test ranks the results

of all the techniques on every dataset. The proposed technique LOCC with base classifier SMO and Naive Bayes was selected as the control method to show its significance. {A} shows that technique A is better than B and C.

In Table 2, the rank results of accuracies show that LOCC considering SMO is better than the comparable counterparts such as for the datasets Flags, Yeast, Emotions, Enron, J.UCS 5Cat and ACM from all three classifiers BR, CC, and OCC. In case of Genbase all the techniques have the same results, while in a few cases, the accuracy of LOCC is close to the other techniques such as Scene and J.UCS datasets. The rank sum results of the Friedman test shows that LOCC has a better value than the other three techniques. Rank sum result for LOCC, OCC, CC and BR were 15,21,23 and 31, respectively. The results of Friedman test for the obtained value of p (0.03305) are significant at 0.05. The X<sup>2</sup>r statistics value of the results with SMO as a base classifier was 8.7333. The results show that LOCC is better than OCC, CC and BR. The technique OCC is better than CC and BR, and CC is better than BR technique only.

In Table 2, the rank result of LOCC considering Naive Bayes as a base classifier show that the accuracy is better on the Yeast, Enron, J. UCS and ACM datasets. On other datasets, its accuracy is close to the other three techniques. The rank sum results from the Friedman test show that the result for LOCC (16) is better than the existing techniques, LPC-CC which has a rank sum value of 21.5, CC with 19.5 and BR with 33. In this experiment

**Table 2:** Accuracy on 9 datasets by four techniques using SMO and Naive Bayes

Base classifier	SMO				Naïve Bayes			
	BR (Rank)	CC (Rank)	OCC (Rank)	LOCC (Rank)	BR (Rank)	CC (Rank)	LPC-CC (Rank)	LOCC (Rank)
Flags	0.597 (1.5)	0.596 (3)	0.595 (4)	0.597 (1.5)	0.473 (4)	0.540 (1)	0.537 (2)	0.517 (3)
Yeast	0.502 (3)	0.488 (4)	0.520 (2)	0.544 (1)	0.385 (4)	0.402 (2)	0.400 (3)	0.403 (1)
Genbase	0.989 (2.5)	0.989 (2.5)	0.989 ( 2.5)	0.989 ( 2.5)	0.433 (1)	0.273 (3.5)	0.273 (3.5)	0.277 ( 2)
Emotions	0.497 (4)	0.511 (2)	0.507 (3)	0.525 (1)	0.466 (4)	0.511 (1)	0.494 (3)	0.501 (2)
Scene	0.586 (4)	0.696 (2)	0.728 (1)	0.662 ( 3)	0.457 (4)	0.696 (1)	0.675 ( 2)	0.662 (3)
Dataset Enron	0.397 (4)	0.407 ( 2.5)	0.407 (2.5)	0.408 (1)	0.205 (4)	0.239 (3)	0.355 (2)	0.395 (1)
J.UCS	0.400 (4)	0.446 (2)	0.448 (1)	0.438 (3)	0.359 (4)	0.388 (3)	0.402 (1)	0.395 (2)
J.UCS 5Cat	0.550 (4)	0.570 (2.5)	0.570 (2.5)	0.580 (1)	0.523 (4)	0.555 (2.5)	0.555 (2.5)	0.583 (1)
ACM	0.900 (4)	0.902 (2.5)	0.902 (2.5)	0.920 (1)	0.826 (4)	0.840 (2.5)	0.840 (2.5)	0.860 (1)
Rank sums	31	23	21	15	33	19.5	21.5	16
Order	{LOCC}>{BR,CC, OCC}, {OCC}>{CC,BR}, {CC}>{BR}							
Order	{LOCC}>{BR,CC, OCC}, {CC}>{LPC-CC, BR}, {LPC-CC}>{BR}							

the CC results were even better than LPC-CC. CC sum rank results was better than both techniques LPC-CC and BR, although the result of LPC-CC were better than BR only. The results obtained through Friedman test in this experiment with base classifier Naive Bayes were significant with the p value 0.01266 at 0.05.

The lower hamming loss values show the efficacy of the proposed technique. In Table 3, the overall values of hamming loss for LOCC are less than the mentioned classifiers for most of the datasets. Rank sum results of all the techniques from the Friedman test with the base classifier SMO show that LOCC has a higher rank than

the other three techniques. The rank sum results for BR, CC, OCC and LOCC are 31, 21, 22, and 16 respectively. Over the nine datasets OCC has better ranks. Friedman test for the different runs of the four classifiers shows no significant difference. The p value for the base classifier SMO was 0.05033, which is not significant at 0.05. Similarly, with base classifier the rank sum results for BR, CC, LPC\_CC and LOCC are 22.5, 22, 26 and 19.5, respectively. LOCC has a better rank than the three comparative techniques. The results of the four techniques were not significant based on the Friedman test. Hamming loss values for the ten runs out of 18 were better for LOCC than the other three techniques.

**Table 3:** Hamming loss on all datasets by four classifiers using SMO and naive Bayes

Base classifier Classifier	SMO				Naïve Bayes			
	BR (Rank)	CC (Rank)	OCC (Rank)	LOCC (Rank)	BR (Rank)	CC (Rank)	LPC-CC (Rank)	LOCC (Rank)
Flags	0.271 (4)	0.264 (2)	0.266 (3)	0.262 (1)	0.335 (4)	0.305 (1)	0.312 (2)	0.320 (3)
Yeast	0.195 (2)	0.214 (4)	0.205 (3)	0.180 (1)	0.293 (1)	0.310 (2.5)	0.315 (4)	0.310 (2.5)
GenBase	0.001 (2.5)	0.001 (2.5)	0.001 (2.5)	0.001 (2.5)	0.033 (2)	0.035 (4)	0.034 (3)	0.032 (1)
Emotions	0.299 (4)	0.224 (2)	0.228 (3)	0.220 (1)	0.232 (2)	0.224 (1)	0.264 (4)	0.257 (3)
Scene	0.107 (3)	0.103 (2)	0.092 (1)	0.115 (4)	0.178 (3)	0.103 (1)	0.111 (2)	0.238 (4)
Dataset Enron	0.068 (5)	0.060 (2.5)	0.060 (2.5)	0.050 (1)	0.184 (4)	0.177 (2.5)	0.177 (2.5)	0.140 (1)
J.UCS	0.105 (3.5)	0.102 (1)	0.103 (2)	0.105 (3.5)	0.127 (1)	0.149 (4)	0.146 (2)	0.148 (3)
J.UCS 5Cat	0.178 (4)	0.171 (2.5)	0.171 (2.5)	0.160 (1)	0.181 (2)	0.186 (4)	0.185 (3)	0.171 (1)
ACM	0.075 (4)	0.065 (2.5)	0.065 (2.5)	0.053 (1)	0.098 (3.5)	0.097 (2)	0.098 (3.5)	0.090 (1)
Rank sum	31	21	22	16	22.5	22	26	19.5
Order	No significant differences according to Friedman test							

**Table 4:** F1 (macro averaged by label) on all datasets by four classifiers using SMO and naive Bayes

Base classifier Classifier	SMO				Naïve Bayes			
	BR	CC	OCC	LOCC	BR	CC	LPC-CC	LOCC
Flags	0.645 (3)	0.637 (4)	0.647 (2)	0.657 (1)	0.625 (4)	0.666 (1)	0.661 (2)	0.658 (3)
Yeast	0.325 (4)	0.362 (2)	0.353 (3)	0.380 (1)	0.423 (4)	0.442 (2)	0.433 (3)	0.448 (1)
GenBase	0.768 (2.5)	0.768 (2.5)	0.768 (2.5)	0.768 (2.5)	0.130 (1)	0.047 (3.5)	0.047 (3.5)	0.048 (2)
Emotions	0.611 (3)	0.603 (4)	0.639 (2)	0.642 (1)	0.593 (4)	0.603 (3)	0.609 (2)	0.618 (1)
Scene	0.688 (3)	0.718 (2)	0.750 (1)	0.683 (4)	0.595 (3)	0.718 (1)	0.693 (2)	0.580 (4)
Dataset Enron	0.218 (3)	0.219 (2)	0.217 (4)	0.227 (1)	0.144 (4)	0.178 (2.5)	0.178 (2.5)	0.357 (1)
J.UCS	0.318 (4)	0.332 (2)	0.334 (1)	0.319 (3)	0.336 (4)	0.35 (3)	0.353 (2)	0.357 (1)
J.UCS 5Cat	0.463 (4)	0.466 (2.5)	0.466 (2.5)	0.483 (1)	0.491 (4)	0.510 (2.5)	0.510 (2.5)	0.591 (1)
ACM	0.570 (4)	0.580 (2.5)	0.580 (2.5)	0.610 (1)	0.517 (4)	0.537 (2.5)	0.537 (2.5)	0.547 (1)
Rank sums	30.5	23.5	20.5	15.5	32	21	22	15
Order SMO	{LOCC}<{BR,CC, OCC}, {OCC}<{BR, CC}, {CC}<{BR}							
Order naive Bayes	{LOCC}<{BR,CC, OCC}, {CC}<{LPC-CC, BR}, {LPC-CC}<{BR}							



As mentioned earlier, the efficiency of an algorithm is achieved by higher values for F1 measure. The result of the F measure averaged by label on all the datasets using the four techniques given in Table 4 shows that LOCC has better accuracy on Flag, Yeast, Emotions, Enron, J. UCS and ACM datasets. Rank sum for BR technique results, CC, OCC and LOCC are 30.5, 23.5, 20.5 and 15.5, respectively. LOCC is better in order than BR, CC and OCC, whereas OCC is better than BR and CC. CC is better than only BR. The results are significant at 0.05 based on the Friedman test with a p value 0.04885.

The result of F measure with Naive Bayes as a base classifier for the techniques BR, CC, LPC-CC and LOCC are given in Table 5. The results are significant with a p of value 0.04885 based on the Friedman test. The rank sum values for the BR, CC, LPC-CC and LOCC are 32, 21, 22 and 15 respectively. The rank sum results show the outstanding order of LOCC over LPC-CC, CC and

BR. The result shows that CC has a better order than LPC-CC and BR. BR is placed lower in this order among the four techniques.

The lower value for total time parameter shows the supremacy of the proposed technique over the existing techniques OCC and LPC - CC. In Table 5, the comparison is made only with OCC and LPC - CC because in existing techniques only these two techniques make extra processing. The LOCC values are low for all datasets with base classifier SMO and existing technique OCC. Similarly, in Naive Bayes most of the values of LOCC are less for almost all of the datasets that show the basic supremacy of our proposed technique. The tradeoff between computational time and accuracy is much higher for the existing techniques (LOCC, LPC-CC) and the proposed algorithm. LOCC achieve better accuracy in a very short time.

**Table 5:** Total time (in seconds) on all datasets by 3 classifiers using SMO and naive Bayes

Base classifier		SMO		Naive Bayes	
		OCC (Rank)	LOCC (Rank)	LPC-CC (Rank)	LOCC (Rank)
Dataset	Flags	0.784 (2)	0.391 (1)	0.141 (1.5)	0.141 (1.5)
	Yeast	18.557 (2)	11.643 (1)	4.52 (2)	2.367 (1)
	GenBase	19.071 (2)	9.676 (1)	9.249 (2)	4.491 (1)
	Emotions	1.191 (2)	1.238 (1)	0.641 (2)	0.396 (1)
	Scene	16.535 (2)	8.2 (1)	14.09 (2)	2.486 (1)
	Enron	110.569 (2)	55.506 (1)	140.096 (2)	70.407 (1)
	J.UCS	18.842 (2)	12.651 (1)	73.16 (2)	36.787 (1)
	J.UCS 5Cat	6.494 (2)	4.65 (1)	22.787 (2)	13.116 (1)
	ACM	2168.655 (2)	988.332 (1)	2383.455 (2)	1305 (1)
	Rank sum	18	9	17.5	9.5
	Ordering	{LOCC}>{OCC, LPC-CC}			

## CONCLUSION

The label orders play an important role in classifier accuracy, especially when the number of labels is hefty. We have concluded from the experimental analysis that our algorithm achieves better accuracy than the previous algorithm in a short time. In future, more consideration can be given to exploit more semantic relations among the labels of a dataset. The similarity measure can be amalgamated with some other measures to improve the label ordering.

## REFERENCES

- Afzal M. T., Balke W. T., Maurer H., & Kulathuramaiyer N. (2009, July). Improving citation mining. *Proceedings of the First International Conference on Networked Digital Technologies (NDT'09)*, Ostrava, Czech Republic, pp. 116–121.
- Ali T., Sajid N.A., Asghar S. & Ahmed M. (2013). Classification of scientific publications using swarm intelligence. *Pakistan Academy of Sciences* **50**(1): 115–126.
- Cheng W., Hüllermeier E. & Dembczynski K.J. (2010). Bayes optimal multilabel classification via probabilistic classifier

- chains. *Proceedings of the 27<sup>th</sup> International Conference on Machine Learning (ICML-10)*, Haifa, Israel, pp. 279–286.
- Dembczyński K., Waegeman W., Cheng W. & Hüllermeier E. (2012). On label dependence and loss minimization in multi-label classification. *Machine Learning* **88**(1-2): 5–45.
- Friedman M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association* **32**(200): 675–701.
- Goncalves E.C., Plastino A. & Freitas A.A. (2013). A genetic algorithm for optimizing the label ordering in multi-label classifier chains. *Proceedings of the 25<sup>th</sup> International Conference on Tools with Artificial Intelligence (ICTAI)*, Herndon, USA, pp. 469–476.  
DOI: <https://doi.org/10.1109/ICTAI.2013.49>
- Keikha M. & Hashemi S. (2016). Ordered classifier chains for multi-label classification. *Journal of Machine Intelligence* **1**(1): 7–12.
- Kumar A., Vembu S., Menon A.K. & Elkan C. (2012). Learning and inference in probabilistic classifier chains with beam search. *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2012)*, Bristol, UK, pp. 665–680.  
DOI: [https://doi.org/10.1007/978-3-642-33460-3\\_48](https://doi.org/10.1007/978-3-642-33460-3_48)
- Kumar A., Vembu S., Menon A.K. & Elkan C. (2013). Beam search algorithms for multilabel learning. *Machine learning* **92**(1): 65–89.
- Luaces O., Díez J., Barranquero J., del Coz J.J. & Bahamonde A. (2012). Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence* **1**(4): 303–313.
- Mena D., Montañés E., Quevedo J.R. & Coz J.J. (2016). An overview of inference methods in probabilistic classifier chains for multilabel classification. *WIREs Data Mining and Knowledge Discovery* **6**(6): 215–230.  
DOI: <https://doi.org/10.1002/widm.1185>
- Mena D., Montañés E., Quevedo J.R. & del Coz J.J. (2017). A family of admissible heuristics for A\* to perform inference in probabilistic classifier chains. *Machine Learning* **106**(1): 143–169.
- Montañés E., Senge R., Barranquero J., Quevedo J.R., del Coz J.J. & Hüllermeier E. (2014). Dependent binary relevance models for multi-label classification. *Pattern Recognition* **47**(3): 1494–1508.
- Read J., Pfahringer B., Holmes G. & Frank E. (2009). Classifier chains for multi-label classification. *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2009)*, Bled, Slovenia, pp. 254–269.
- Read J., Pfahringer B., Holmes G. & Frank E. (2011). Classifier chains for multi-label classification. *Machine Learning* **85**(3): 333–359.
- Read J., Martino L. & Luengo D. (2014). Efficient Monte Carlo methods for multi-dimensional learning with classifier chains. *Pattern Recognition* **47**(3): 1535–1546.
- Rousu J., Saunders C., Szedmak S. & Taylor J.S. (2006). Kernel-based learning of hierarchical multilabel classification models. *Journal of Machine Learning Research* **7**: 1601–1626.
- Sajid N., Muhammad A. & Muhammad Q. (2016). Multi-label classification of computer science documents using fuzzy logic. *Journal of the National Science Foundation of Sri Lanka* **44**(2): 155–165.  
DOI: <http://doi.org/10.4038/jnsfsr.v44i2.7996>
- Santos A.P. & Rodrigues F. (2009). Multi-label hierarchical text classification using the acm taxonomy. *Proceedings of the 14th Portuguese Conference on Artificial Intelligence (EPIA)*, Aveiro, Portugal, pp. 553–564.
- Senge R., Coz Velasco J.J.D. & Hüllermeier E. (2013). Rectifying classifier chains for multi-label classification. *Space* **2**(8).
- Senge R., Del Coz J.J. & Hüllermeier E. (2014). On the problem of error propagation in classifier chains for multi-label classification. In: *Data Analysis, Machine Learning and Knowledge Discovery. Proceedings of the 36<sup>th</sup> Annual Conference of the German Classification Society -2012*, (eds. M. Spiliopoulou, L. Schmidt-Thieme, & R. Janning), Springer, Germany, pp. 163–170.  
DOI: [https://doi.org/10.1007/978-3-319-01595-8\\_18](https://doi.org/10.1007/978-3-319-01595-8_18)
- Soonsiripanichkul B. & Murata T. (2016). Domination dependency analysis of sales marketing based on multi-label classification using label ordering and cycle chain classification. *Proceedings of the 5<sup>th</sup> IIAI International Congress on Advanced Applied Informatics*, Kumamoto, Japan, pp. 1048–1053.
- Tsoumakas G., Spyromitros-Xiouis E., Vilcek J. & Vlahavas I. (2011). Mulan: A java library for multi-label learning. *Journal of Machine Learning Research* **12**: 2411–2414.