



# Incorporation of Neighborhood Concept in Enhancing SOM Based Multi-label Classification

Naveen Saini<sup>(✉)</sup>, Sriparna Saha, and Pushpak Bhattacharyya

Indian Institute of Technology Patna, Bihar, India  
{naveen.pcs16,sriparna,pb}@iitp.ac.in

**Abstract.** The self-organizing map (SOM), which is a type of neural network, helps in the exploratory phase of data mining by projecting the input data into a lower-dimensional map consisting of a grid of neurons. In recent years, SOM has also been applied for classification of data points. The prominent utility of SOM based classification is evident from the use of no labeled data during training. In this paper, a self-organizing map based algorithm is proposed to solve the multi-label classification problem, named as ML-SOM. SOM follows an unsupervised training process to learn the topological structure of the training points. At testing-phase, a testing instance can be mapped to a specific neuron in the network and it's label can be determined using the training instances mapped to that specific neuron and nearby neurons. Thus in this paper, we have considered the neighborhood information of SOM to determine the label vector of testing instances. Experiments were performed on five multi-labeled datasets and performance of the proposed system is compared with various state-of-the-art methods showing competitive performance. Results are also validated using statistical significance t-test.

**Keywords:** Self-organizing Map · Multi-label classification · Unsupervised methods · Neighborhood function · Gaussian function

## 1 Introduction

In general, classifying an object or instance,  $x_i$ , in machine learning refers to assigning a single label out of a set of disjoint labels,  $L$ . This type of task is called as single label classification problem. But, in real-life, we can encounter with some classification problems where each of the data instances,  $x_i$ , may belong to more than one class and the task is to predict multiple labels of that instance and thus it can be referred to as multi-label classification (MLC) [5]. Now-a-days, MLC has become a hot research area and its applications can be found in various real-life domains like bioinformatics [7], image classification, etc. In document classification problem (DCP), one document may belong to more than one category, for example, biology and computer science. Therefore, DCP can also be considered as a MLC problem.

In the literature, several techniques are developed to deal with multi-label classification problems. As discussed in ref. [2], two types of existing approaches are there: algorithm dependent and algorithm independent. In algorithm independent approach, traditional classifiers are used which transform the multi-label classification problem into a set of single-label classification problems. While, this is not the case in algorithm dependent approach for multi-label task. In algorithm independent approach, each classifier is associated to some class and used to solve binary classification problem. This developed method is called as Binary-Relevance (BR) [3]. But, BR method has some drawbacks: it considers the classes independent of each other, which is not always true. Some of the examples of multi-label classifiers belonging to BR category are: Support Vector Machine-BR, J48 Decision tree-BR and k-Nearest Neighbor-BR. Later, some improvements over BR method were proposed [1]. Another version of algorithm-independent approach is LP (Label-Powerset) transformation method. All the classes allocated to a particular instance are combined into a unique and new class by considering correlations among-st classes. But this method increases the number of classes. An example of algorithm dependent approach is ML-kNN [8] in which for each instance, k-nearest classes are determined. Then principle of maximum posteriori is used to find the classes of a new instance.

Most of the proposed algorithms for multi-label classification are supervised in nature which require some labeled data for training the models. Therefore, there is a need to develop an unsupervised/semi-supervised framework to deal with multi-label classification which can achieve comparable results or can outperform the supervised methods. In the current paper, unsupervised neural network, called as self-organizing map [4], is used for proposing a multi-label classification framework which does not require any labeled data at the time of training. Self-organizing Map (SOM) is a neural network model consisting of two layers: input and output. Output layer is a grid of neurons arranged in low-dimensional manner. The principle of SOM states that the input patterns which are similar to each other in the input space appear next to each other in the neuron space. This is due to cooperation and adaptation process of SOM. Thus, it can be used for classification purpose. Usually, low dimensional space consists of 2-d grid of neurons. Let  $T = \{x_1, x_2, \dots, x_H\}$  be a set of  $H$  training samples in  $n$ -dimensional space, then each neuron (or map unit)  $u \in D$  (number of neurons) has: (a) a predefined position in the output space:  $z^u = (z_1^u, z_2^u)$ ; (b) a weight vector  $w^u = [w_1^u, w_2^u, \dots, w_n^u]$ . It is important to note that dimension of weight vector of a neuron should be equal to vector dimension of input vector to perform mapping.

Label information of the testing instances is used while checking the performance of the system. However, there are many previous works on supervised SOM that make use of labeled data during training. We can use the supervised SOM to increase the performance of multi-label classification task, but, generating labeled data is a time consuming and cost sensitive process. Therefore, in this paper, we adopted the unsupervised SOM.

Recently, [2] have proposed a SOM based method for multi-label classification and used the traditional SOM training algorithm. But, it suffers from

following drawbacks: (a) during testing an instance, it considers only the winning (mapping) neuron to decide its label vector. Label vector can be obtained by first averaging the label vectors of training instances mapped to that winning neuron and then some threshold value can be utilized to decide the class labels. The neighborhood information captured by SOM, one of its key-characteristics, was not utilized at the time of generating the label vector of any test-instance. In the current study we incorporated the use of neighborhood information during the testing phase of SOM based multi-label classification framework. In our proposed algorithm, we have also varied the number of neighboring neurons. (b) Authors have not given any information regarding the parameters like SOM training parameters, threshold value, used in their algorithm, therefore, sensitivity analysis is performed in our framework to determine the best values of the parameters used.

Experiments were performed on five multi-labeled datasets and results are compared with various existing supervised and unsupervised methods for MLC. Results illustrate that our system is superior to previously existing SOM based multi-label classifier and some other existing methods. Rest of the sections are organized as below: Sect. 2 discusses the proposed framework. Section 3 and Sect. 4 describe the experimental setup and discussion of results, respectively. Finally, Sect. 5 concludes the paper.

## 2 Proposed Methodology for Multi-label Classification Using Self Organizing Map

In this section, we first present the self-organizing map and its procedure of mapping instances to neurons. Then we will discuss the proposed classification algorithm (ML-SOM) for multi-label instances.

**1. Representation of Label Vector:** In multi-labeled data, an instance  $x_i$  may belong to more than one class. Therefore, label vector ( $LV_i$ ) of instance  $x_i$  will be represented by a binary vector and the size of the binary vector will be equal to the number of classes. The  $k$ th position of the label vector will be 1 if that particular instance belongs to class 'k', otherwise, it will be 0. For example, if there are 10 classes and an instance belongs to first, third, fifth, seventh and ninth classes, then binary vector will be represented as  $[1, 0, 1, 0, 1, 0, 1, 0, 1, 0]$ .

**2. SOM Training Algorithm:** Training of SOM starts after initializing the weight vectors of neurons denoted as  $w = \{w_1, w_2, \dots, w_D\}$ , where  $D$  is the number of neurons. These weight vectors are chosen randomly from the available training samples. In this paper, we have used the sequential learning algorithm for training of SOM. However, batch algorithm can also be used for the same task. Basic steps of the learning approach of SOM based multi-label classification can be found in the paper [2] and the brief overview of the algorithm is presented below. Let  $Maxiter$  be the maximum number of SOM training iterations,  $\eta_0$  and  $\sigma_0$  be the initial learning rate and neighborhood radius, respectively, which decrease continuously as the number of iterations increases. For each data sample presented to the network, first its winning neuron is

determined using the shortest Euclidean distance criterion. Then, neighboring neurons around the winning neuron are determined using the position vectors of the neurons. The best choice of selecting neighborhood is Gaussian function which is represented as  $h_{uj} = \exp(-\frac{d_{uj}^2}{2\sigma^2})$ , where,  $u$  is a winning neuron index and  $j$  is the neighboring neuron index,  $d_{uj}$  is the Euclidean distance between neuron  $u$  and neuron  $j$  using position vectors,  $\sigma$  is the neighborhood radius and calculated as  $\sigma = \sigma_0 * (1 - \frac{t}{Maxiter})$ , where,  $t$  is the current iteration number. Finally, weights of the winning and neighboring neurons are updated as  $w^u = w^u + \eta \times h_{u',j} \times (x - w^u)$ , where,  $\eta = \eta_0 * (1 - \frac{t}{Maxiter})$ . It was done so that they come close to the input samples presented to the network and form a cluster of similar neurons around the winning neuron. Note that for a particular neuron, initially all the remaining neurons are neighbors (represented by  $\sigma$ ) and this will keep on decreasing as training iteration continues.

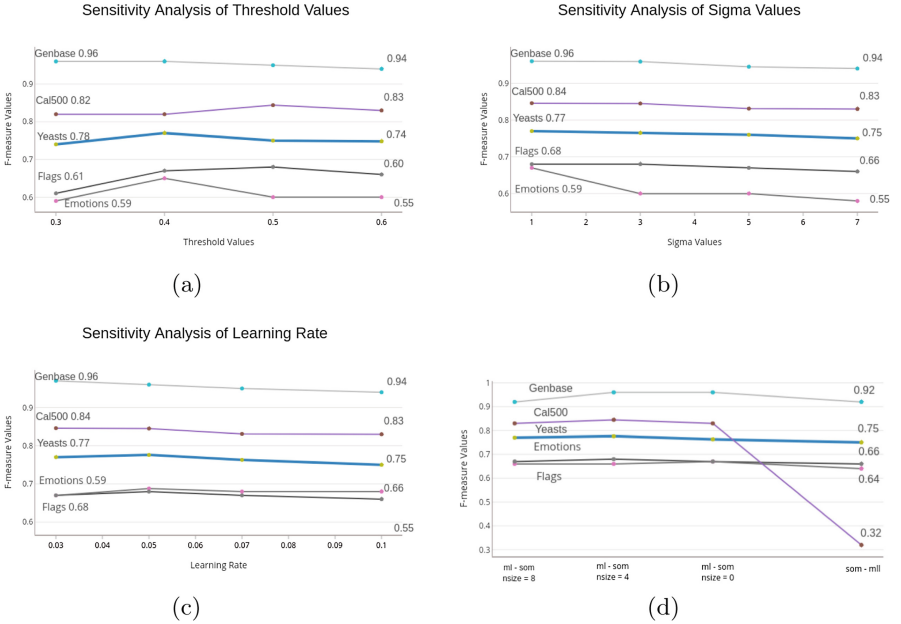
**3. Multi-label Data Classification Procedure:** When a test instance  $x_i$  is presented to the trained SOM network, then firstly mapping is performed *i.e.*, its closest neuron ‘b’ is determined. Now, the label vector of the test instance  $x_i$  can be determined using the following steps:

- (a) Perform averaging of label vectors of the training instances mapped to the closest neuron ‘b’ and nearby (adjacent neighbors) neurons ‘N’. The obtained vector is called prototype vector ( $PV_i$ ) and the  $k^{th}$  value of the prototype vector indicates the probability of test instance belonging to class ‘k’. The  $k^{th}$  value is represented as  $PV_{i,k} = \frac{|H_{w,k}| + |H_{N,k}|}{|H_w| + |H_N|}$ , where,  $|H_{w,k}|$  and  $|H_{N,k}|$  are the set of training instances mapped to closest and nearby neurons, respectively, belonging to class ‘k’,  $|H_w|$  and  $|H_N|$  are the total number of training instances mapped to closest and nearby neurons, respectively.
- (b) If the probability of a class in the obtained prototype vector is greater than or equal to some threshold, then probability value will be replaced by 1, otherwise it will be replaced by 0. The obtained binary vector will be the label vector of the testing instance.

**Extracting Neighboring Neurons Around Winning Neuron:** To decide the neighbors around the winning neuron, Euclidean distances between winning neuron and other neurons are calculated using position vectors. For example: suppose there are 9 neurons (indices = 0, 1, 2, . . . , 8) arranged in  $3 \times 3$  grid having position vectors  $\{(0, 0), (0, 1), (0, 2), (1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2)\}$ . If index = 3 is the winning neuron, then indices of neighboring neurons will be  $\{0, 1, 4, 6, 7\}$  as they are adjacent to the winning neuron. In general, if other neurons have distances less than or equal to 1.414, then those neurons will be adjacent neighboring neurons.

### 3 Experimental Setup

For our experimentation, we have chosen various multi-labeled data sets having varied number of labels. These data sets are publicly<sup>1</sup> available and related to different domains like audio, biology, images. Brief description of the datasets used in our experiment can be found in the paper [2]. These datasets are divided into 70% training data and 30% testing data. For the purpose of comparison, seven supervised algorithms and one recently proposed unsupervised algorithm are used. Supervised algorithms include: J48 Decision tree, Support Vector Machine (SVM), k-Nearest Neighbors (kNN), Multi-label k-Nearest Neighbor (MLkNN) [8], Back-propagation Multi-label Learning (BPMLL) [6] which is



**Fig. 1.** Sensitivity analysis on (a) threshold value; (b)  $\sigma_0$ ; (c)  $\eta_0$  for different datasets; (d) F-measure values obtained using ML-SOM method by varying the neighborhood sizes in comparison with SOM-MLL

**Table 1.** Precision values obtained by different multi-label classification techniques

| Dataset  | ML-SOM      | SOM-MLL     | SVM-BR      | J48-BR      | KNN-BR      | SVM-LP      | J48-LP      | KNN-LP      | BPMLL       | MLkNN       |
|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| cal500   | 0.84 ± 0.03 | 0.60 ± 0.02 | 0.62 ± 0.07 | 0.45 ± 0.08 | 0.35 ± 0.04 | 0.34 ± 0.05 | 0.34 ± 0.04 | 0.35 ± 0.04 | 0.35 ± 0.04 | 0.60 ± 0.06 |
| emotions | 0.68 ± 0.06 | 0.63 ± 0.07 | 0.68 ± 0.10 | 0.59 ± 0.13 | 0.63 ± 0.11 | 0.68 ± 0.16 | 0.58 ± 0.15 | 0.63 ± 0.11 | 0.64 ± 0.12 | 0.70 ± 0.16 |
| flags    | 0.75 ± 0.05 | 0.68 ± 0.05 | 0.72 ± 0.06 | 0.69 ± 0.14 | 0.68 ± 0.16 | 0.69 ± 0.12 | 0.66 ± 0.14 | 0.68 ± 0.16 | 0.69 ± 0.08 | 0.72 ± 0.10 |
| genbase  | 0.96 ± 0.03 | 0.93 ± 0.03 | 0.99 ± 0.02 | 0.99 ± 0.03 | 0.99 ± 0.02 | 0.99 ± 0.02 | 0.99 ± 0.03 | 0.99 ± 0.02 | 0.04 ± 0.04 | 0.98 ± 0.05 |
| yeast    | 0.80 ± 0.03 | 0.71 ± 0.01 | 0.72 ± 0.06 | 0.60 ± 0.06 | 0.60 ± 0.07 | 0.66 ± 0.05 | 0.54 ± 0.06 | 0.60 ± 0.07 | 0.62 ± 0.05 | 0.72 ± 0.04 |
| Average  | 0.80        | 0.71        | 0.75        | 0.66        | 0.65        | 0.67        | 0.62        | 0.65        | 0.47        | 0.74        |

<sup>1</sup> <http://mulan.sourceforge.net/datasets-mlc.html>.

a Neural network based dependent method. Unsupervised algorithm includes SOM-MLL [2] which is based on self-organizing map. For evaluation, three well known measures namely, Precision, Recall, and, F1-measure [7] are utilized in our approach. Their descriptions and mathematical formulations can be found in the paper [7].

**Table 2.** Recall values obtained by different multi-label classification techniques

| Dataset  | ML-SOM          | SOM-MLL         | SVM-BR          | J48-BR          | KNN-BR          | SVM-LP          | J48-LP          | KNN-LP          | BPMLL           | MLkNN           |
|----------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| cal500   | $0.86 \pm 0.03$ | $0.23 \pm 0.01$ | $0.23 \pm 0.04$ | $0.29 \pm 0.07$ | $0.35 \pm 0.06$ | $0.35 \pm 0.06$ | $0.34 \pm 0.05$ | $0.35 \pm 0.06$ | $0.72 \pm 0.05$ | $0.22 \pm 0.05$ |
| emotions | $0.67 \pm 0.03$ | $0.60 \pm 0.05$ | $0.66 \pm 0.11$ | $0.57 \pm 0.10$ | $0.63 \pm 0.08$ | $0.71 \pm 0.09$ | $0.58 \pm 0.17$ | $0.63 \pm 0.08$ | $0.73 \pm 0.11$ | $0.63 \pm 0.18$ |
| flags    | $0.67 \pm 0.04$ | $0.65 \pm 0.06$ | $0.76 \pm 0.16$ | $0.74 \pm 0.12$ | $0.65 \pm 0.14$ | $0.68 \pm 0.10$ | $0.66 \pm 0.15$ | $0.65 \pm 0.18$ | $0.76 \pm 0.12$ | $0.76 \pm 0.17$ |
| genbase  | $0.96 \pm 0.03$ | $0.92 \pm 0.03$ | $0.99 \pm 0.02$ | $0.99 \pm 0.02$ | $0.99 \pm 0.02$ | $0.99 \pm 0.03$ | $0.98 \pm 0.04$ | $0.99 \pm 0.02$ | $0.66 \pm 0.03$ | $0.95 \pm 0.05$ |
| yeast    | $0.77 \pm 0.03$ | $0.54 \pm 0.01$ | $0.58 \pm 0.03$ | $0.58 \pm 0.07$ | $0.60 \pm 0.06$ | $0.62 \pm 0.04$ | $0.54 \pm 0.07$ | $0.60 \pm 0.06$ | $0.69 \pm 0.05$ | $0.59 \pm 0.07$ |
| Average  | 0.79            | 0.59            | 0.64            | 0.63            | 0.64            | 0.67            | 0.62            | 0.64            | 0.71            | 0.63            |

**Table 3.** F-measure values obtained by different multi-label classification techniques

| Dataset  | ML-SOM          | SOM-MLL         | SVM-BR          | J48-BR          | KNN-BR          | SVM-LP          | J48-LP          | KNN-LP          | BPMLL           | MLkNN           |
|----------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| cal500   | $0.84 \pm 0.04$ | $0.32 \pm 0.01$ | $0.34 \pm 0.07$ | $0.34 \pm 0.07$ | $0.34 \pm 0.05$ | $0.34 \pm 0.05$ | $0.33 \pm 0.05$ | $0.34 \pm 0.05$ | $0.45 \pm 0.03$ | $0.32 \pm 0.06$ |
| emotions | $0.65 \pm 0.03$ | $0.60 \pm 0.06$ | $0.60 \pm 0.11$ | $0.55 \pm 0.08$ | $0.60 \pm 0.06$ | $0.67 \pm 0.12$ | $0.55 \pm 0.14$ | $0.60 \pm 0.08$ | $0.66 \pm 0.10$ | $0.63 \pm 0.16$ |
| flags    | $0.66 \pm 0.04$ | $0.64 \pm 0.05$ | $0.73 \pm 0.11$ | $0.70 \pm 0.13$ | $0.65 \pm 0.16$ | $0.67 \pm 0.09$ | $0.66 \pm 0.15$ | $0.65 \pm 0.15$ | $0.70 \pm 0.10$ | $0.73 \pm 0.11$ |
| genbase  | $0.96 \pm 0.03$ | $0.92 \pm 0.04$ | $0.99 \pm 0.02$ | $0.99 \pm 0.03$ | $0.99 \pm 0.02$ | $0.99 \pm 0.03$ | $0.99 \pm 0.04$ | $0.99 \pm 0.02$ | $0.06 \pm 0.06$ | $0.96 \pm 0.05$ |
| yeast    | $0.77 \pm 0.03$ | $0.59 \pm 0.01$ | $0.61 \pm 0.03$ | $0.56 \pm 0.06$ | $0.57 \pm 0.07$ | $0.62 \pm 0.04$ | $0.51 \pm 0.06$ | $0.57 \pm 0.07$ | $0.63 \pm 0.06$ | $0.62 \pm 0.05$ |
| Average  | 0.78            | 0.61            | 0.65            | 0.63            | 0.63            | 0.66            | 0.61            | 0.63            | 0.50            | 0.65            |

**Sensitivity Analysis on Parameters Used:** In SOM, there are 5 parameters used, which are: grid topology (hexagonal or rectangular), initial learning rate ( $\eta_0$ ), initial neighborhood radius ( $\sigma_0$ ), neighborhood function and number of neurons. In our work, rectangular topology is considered, while the number of neurons is kept as  $5 \times 5 = 25$ . During generation of label vector for the test instance, some *threshold* value is used. For each data set, to select the best values of  $\eta_0$ ,  $\sigma_0$  and threshold, we have performed sensitivity analysis on these parameters. For this analysis, we have varied the values of one parameter while keeping others as fixed. For example, to determine the best value of  $\eta_0$ , we have executed the experiment with varied value of  $\eta_0$ , while keeping  $\sigma_0$  and *threshold* values fixed. The value of  $\eta_0$  at which we got the best value of F-measure is considered as the best value of  $\eta_0$ . Now to determine the best value of  $\sigma_0$ , we have fixed  $\eta_0$  (equals to the best value obtained) and *threshold* values. The value of  $\sigma_0$  at which we attained the best value of F-measure is considered as the best value of  $\sigma_0$ . Similar experiments are executed to determine the best value of *threshold*. Thus, the F-measure values obtained by varying parameters, *threshold*,  $\sigma_0$  and  $\eta_0$ , are shown in Fig. 1(a), (b) and (c), respectively. Following are values obtained for different datasets: (a) flags: threshold = 0.4,  $\sigma_0 = 1$  and  $\eta_0 = 0.04$ ; (b) emotions: threshold = 0.4,  $\sigma_0 = 1$  and  $\eta_0 = 0.05$ ; (c) cal500: threshold = 0.5,  $\sigma_0 = 1$  and  $\eta_0 = 0.04$ ; (d) yeast: threshold = 0.4,  $\sigma_0 = 1$  and  $\eta_0 = 0.05$ ; (e) genbase: threshold = 0.3,  $\sigma_0 = 1$  and  $\eta_0 = 0.04$ .

In paper [2], grid topology was taken as *hexagonal*, while in our approach, it is taken as *rectangular* to see the performance improvement, but, the number of neurons are kept fixed in the grid *i.e.*, 25. The results reported in this paper by ML-SOM methods are the average values over 10 runs and framework is implemented on a Intel Core i7 CPU 3.60 GHz with 4 GB of RAM on Ubuntu.

## 4 Discussion of Results

Results obtained by our proposed methods (ML-SOM) for Precision, Recall and F-measure on five datasets are shown in Tables 1, 2 and 3, respectively. In these tables, algorithm-independent approaches are represented by Binary-relevance (BR) or Label-Powerset (LP). We have performed the experiment by varying the size of neighboring neurons (NS) around the winning neuron, *i.e.*, number of adjacent neurons to winning neuron as: 8, 4, and 0, using the best values of the parameters obtained after sensitivity analysis for different datasets. Here, NS = 8 means, neighboring neurons will be at distance less than or equal to 1.414, NS = 4 means neurons lying within distance of 1, NS = 0 means, no neighboring neuron is considered. In the tables, results shown are corresponding to NS = 4 (as we achieve good results using this size, see Fig. 1(d)). It is surprising to see that in SOM-MLL method, label vector of a testing instance is calculated using label vectors of training instances mapped to winning neuron of the testing instance or we can say they have taken NS = 0. This proves that incorporation of neighborhood information in determining the label vector of testing instance helps in getting better results.

Considering the Precision value in Table 1, ML-SOM method gives best results for *cal500*, *flags* and *yeast* data sets in comparison to supervised and unsupervised methods. While for remaining datasets, MLkNN performs the best. Regarding recall value shown in Table 2, for *cal500* and *yeast* data sets, our method performs the best. While for *emotions*, *flags* and *genbase* data sets, BPMLL, SVM-BR and SVM-LP perform better in comparison to other methods. After observing the F-measure table (see Table 3), we can conclude that our system performs best for *cal500* and *yeast* data sets. But, if we consider the average precision, recall and F-measure values of our proposed method, ML-SOM, then those are better than all remaining methods' average precision, recall and F-measure values which proves the overall effectiveness of the proposed method.

Figure 1(d) shows the F-measure value obtained by our proposed approach vs. different neighborhood sizes (NS) in comparison with SOM-MLL which considers NS = 0. This figure clearly indicates that when NS = 4 is considered, the best value of F-measure is obtained by our ML-SOM method and this is better than F-measure value of SOM-MLL method.

In general, experimental results show that our unsupervised method achieve competitive (or in some cases, best) performance in comparison to supervised algorithms. Our proposed methods obtained best results in comparison to SOM-MLL method, which is an unsupervised method for multi-label classification. To check the superiority of our proposed ML-SOM method, statistical hypothesis

test<sup>2</sup> is conducted at the 5% significance level. It checks whether the improvements obtained by the proposed approach are happened by chance or those are statistically significant. This t-test provides p-value. Smallest p-value indicates that the proposed approach is better than others. This test is conducted using the F-measure values obtained by the proposed method reported in Table 3 over other unsupervised methods, namely, SOM-MLL. The p-values obtained for *cal500*, *emotions*, *flags*, *genbase*, and *yeast* data sets, are 0.00001, 0.000522, 0.065792, 0.001394 and 0.00001, respectively, out of which the p-values 0.00001, 0.000522 and 0.00001 for *cal500*, *emotions*, *yeast* data sets evidently support our results. The higher p-values for remaining datasets are due to competitive F-measure by our method over SOM-MLL as can be seen in Table 3.

## 5 Conclusion

In the current paper, a self-organizing map based algorithm is proposed to solve the multi-label classification problem (ML-SOM). The principle of SOM is utilized in our framework which states that similar instances will map to nearby neurons in the grid. During training of SOM, no label information was used. It is used only for checking the performance of the system. For classification of a test instance, first it is mapped to closest neuron and then neighboring neurons are detected around the closest neuron using position vectors. Finally, its label vector is determined using the closest and neighboring neurons.

The proposed method was tested on five multi-labeled datasets related to different domains and results are compared with various supervised and unsupervised methods. Obtained experimental results proved that the incorporation of neighboring neurons in finding the label vector of a test instance enhances the system performance. Our system suffers from the problem of fixed number of neurons in the neuron grid. This should be determined adaptively/dynamically as per the training data. In future, we would like to apply this approach for solving different real-life problems of NLP domain.

## References

1. Alvares-Cherman, E., Metz, J., Monard, M.C.: Incorporating label dependency into the binary relevance framework for multi-label classification. *Expert Syst. Appl.* **39**(2), 1647–1655 (2012)
2. Colombini, G.G., de Abreu, I.B.M., Cerri, R.: A self-organizing map-based method for multi-label classification. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 4291–4298. IEEE (2017)
3. Read, J., Pfahringer, B., Holmes, G., Frank, E.: Classifier chains for multi-label classification. *Mach. Learn.* **85**(3), 333 (2011)
4. Simon, H.: *Neural networks and learning machines: a comprehensive foundation* (2008)

<sup>2</sup> <https://www.socscistatistics.com/tests/studentttest/default.aspx>.



5. Tsoumakas, G., Katakis, I.: Multi-label classification: an overview. *Int. J. Data Warehous. Min. (IJDWM)* **3**(3), 1–13 (2007)
6. Tsoumakas, G., Spyromitros-Xioufis, E., Vilcek, J., Vlahavas, I.: MULAN: a Java library for multi-label learning. *J. Mach. Learn. Res.* **12**(Jul), 2411–2414 (2011)
7. Venkatesan, R., Er, M.J.: Multi-label classification method based on extreme learning machines. In: 2014 13th International Conference on Control Automation Robotics and Vision (ICARCV), pp. 619–624. IEEE (2014)
8. Zhang, M.L., Zhou, Z.H.: A k-nearest neighbor based algorithm for multi-label classification. *GrC* **5**, 718–721 (2005)