Discriminative Adaptive Sets for Multi-Label Classification

MUHAMMAD USMAN GHANI, MUHAMMAD RAFI, (Member, IEEE), AND MUHAMMAD ATIF TAHIR
School of Computer Science, National University of Computer and Emerging Sciences, Karachi Campus, Karachi 75030, Pakistan

Corresponding author: Muhammad Usman Ghani (musman.ghani@nu.edu.pk)

This work was supported in part by the Higher Education Commission (HEC) Pakistan, and in part by the Ministry of Planning Development and Reforms under the National Center in Big Data and Cloud Computing.

ABSTRACT Multi-label classification aims to associate multiple labels to a given data/object instance to better describe them. Multi-label data sets are common in a lot of emerging application areas like: Text/Multimedia classification, Bio-Informatics, Medical image annotations and Computer Vision to name a few. There is a growing interest in efficient and accurate multi-label classification. There are two major approaches to perform multi-label classification (i) problem transformation methods and (ii) algorithm adaptation methods. In algorithm adaptation, the traditional classification algorithms are modified to handle multi-label data sets. One classification algorithm which is often modified to do multi-label classification is k- nearest neighbor (kNN). k-nearest neighbor is popular due to its simplicity, easy to implement and seamlessly adaptability. Despite its merits it has several drawbacks like: sensitivity of noisy data, missing values and outliers; feature scaling and often becoming inaccurate for large overlapping solution space. In this paper, a modification to kNN method is suggested for multi-label classification with three improvement strategies (i) selection of local example w.r.t. unknown example – the motivation for this comes from the fact that local and relevant space is vital for the improvement in multi-label classification; (ii) Splitting the input space into multiple sub-spaces for optimal label estimation – the motivation is to estimate label accurately in the presence of noisy labels; And (iii) selection of labels using Mean Average Precision (MAP) estimates – here our motivation is to utilize the training data effectively to maximize the hidden distribution and optimal parameters for the method. The proposed method is implemented and compared with state-of-the-art approaches based on kNN or similar approaches that effectively select and optimize relevant spaces for multi-label classification. Evaluation based on multiple metrics like Hamming loss, Precision/Recall and F-measure are used for evaluation. The suggested approach performed much better than the state-of-the-art on the datasets with strong label cardinalities.

INDEX TERMS Lazy learning, multi-label classification, instance based learning, MAP estimation, nearest neighbors.

I. INTRODUCTION

A. BACKGROUND

Several researches have been conducted in ML for binary or multi-class classification in which a single label from a set of labels/classes \( \mathcal{L} \) is assigned to each instance in the dataset. Multi-label classification is an emerging field that refers to the set of instances which have more than one label assigned. It is very popular in categorization of text, classification of multimedia, bio-informatics, classification of protein functions and semantic scenes. Formally, Let \( \mathcal{X} \) be an an space, and \( \mathcal{L} = \{ \lambda_1, \lambda_2, \ldots, \lambda_k \} \) be a set of finite labels. An example \( x \in \mathcal{X} \), represented in terms of features vector \( x = (x_1, x_2, \ldots, x_m) \), is assigned with a subset of labels \( \mathcal{L} \in 2^\mathcal{L} \). See Table 1 for an example.

| Instance | Attr1 | Attr2 | Attr3 | Class  |
|----------|-------|-------|-------|--------|
| 1        | A     | 1.2   | 787   | \( \lambda_1, \lambda_2 \) |
| 2        | A     | 2.3   | 5415  | \( \lambda_1, \lambda_3, \lambda_4 \) |
| 3        | B     | 3.8   | 7507  | \( \lambda_4 \) |
| 4        | C     | 0.5   | 7560  | \( \lambda_2, \lambda_4 \) |

The associate editor coordinating the review of this manuscript and approving it for publication was Bohui Wang.
The types of multi-label classification include problem transformation methods, and algorithm adaptation methods [1]. In problem transformation, the task of classifying multi-label is transformed to multi-class classification or multi-label ranking tasks. Whereas in algorithm adaptation, the algorithm is modified to cater to the multi-label data.

In algorithm adaptation, the instance-based multi-label learning has gained importance in research since the $k$-Nearest Neighbors concept was introduced. This concept effectively overcome the challenges with multi-label datasets owing to its simplicity and easy to validate nature.

B. MOTIVATION

1) CHALLENGES WITH MULTI-LABEL DATASETS

There are still very few approaches for multi-label classification considering the effectiveness of single class instance collection. These methods focuses on the boundaries between multiple classes. These boundaries play a vital role in defining the class of the instance. Among the simplest classification problem in machine learning is the binary classification where there is only one class, and an instance may or may not belong to it. But in case of multi-class classification, there are more classes present but any given instance can belong to any one of it. The problem with multi-label data sets is that instances may potentially be associated to multiple classes. This blurs the boundaries due to the overlapping of different labels. Examples of multi-class and multi-label data sets are shown in Fig. 1 and Fig.2 respectively. Figures have been taken from [2].

![Figure 1. Examples of single-label data instances: binary problems are on left and multi-class problems are on right. Dashed lines represent the class boundaries.](image1)

2) WHY LAZY LEARNING?

Several machine learning algorithms are called as eager learning methods because they are generalized on training instances before any inference is made from the system. In these approaches, a model is trained using a collection of examples usually called as training sets and then this model is used to classify the unseen examples (test set). Eager learning methods have some limitations. If the data is not distributed evenly in the input space, these techniques will have a negative impact on the overall generalization. Intuitively, the model trained on training set may not be able to provide good results on test instances as it tries to extract features globally found in the whole training set and not the distinguishing features of the individual samples. The reason for this is that it tries to minimize the global error which may not work for some particular regions of the input space. Lazy learning methods try to grab only those patterns that are more appropriate for the learning tasks. They delays generalization of training data until the test instances are presented to them, thus constructing hypothesis directly from the training instances. The selection of labels is made by using some similarity measures. The primary motivation for employing and improving lazy learner is due to its adaptation by online applications. In online application, such as recommendation systems, the data is in continuously updated and any eager learner would be obsolete in a relatively short span of time. Lazy learners are very effective for large continuously updating datasets with a small number of attributes. These learners compute target functions locally for each query; Hence, these systems can solve multiple problems simultaneously in dynamically changing datasets. The different multi-label approaches proposed in the literature have unique voting mechanism. Typical examples include BR$k$_NN [3] and ML$k$_NN [4].

3) ADAPTIVE SETS OF NEIGHBORS

The efficient way of finding adaptive examples for classification in lazy learning could help in a number of applications. The classification of an unknown instance is heavily dependent on how good or similar the neighbors are. In this work, we have used mutual and non-mutual sets for identifying the relevant instances for an unknown example. These rules will help to identify the adaptive neighbors in the surrounding. The mutuality strategies have already produced good results in multi-class as well as in multi-label classification [5]. We have extended this work further to address some issues in the traditional multi-label classification algorithms.

4) REGION OF INTEREST FOR UNKNOWN EXAMPLES

One of the core issues in the ML$k$_NN [4] is that it only considers the global differences of instances and fails to
extract benefit from the local regions. There are many real world datasets that contain more information when local regions are considered. Wang in LAMLKNN [6] conducted an experiment on multiple MULAN datasets and analyzed the local differences of samples. In the first step, they counted the number of neighbors of each instance in the dataset and consider only the ones with a single label. Table 3 shows a sample multi-label dataset and algorithm adaptation methods [1]

### II. RELATED WORK

In this section, we briefly reviewed problem transformation and algorithm adaptation methods [1]

#### A. PROBLEM TRANSFORMATION METHODS

A number of transformation methods have been proposed in the past that transform the MLD so that the existing multi-class classification algorithm can be applied to it [7].

Copy transformation method replaces each example \((x_i, Y_i)\) with \(|Y_i|\) examples \((x_i, \lambda_j)\), for each \(\lambda_j \in Y_i\). Dubbed copy-weight method is an extension of copy transformation method. It uses \(\frac{1}{|Y_i|}\) weight to each new example. Some other methods that replaces \(Y_i\) by one of its members were also proposed which basically extract the labels with least and most frequent? Ignored transformation is another algorithm of this family that ignore all the examples in the multi-label instances in the dataset and consider only the ones with a single label. Table 3 shows a sample multi-label dataset and Table 3 shows a transformed multi-label data produced by these approaches.

### TABLE 3. Example multi-label dataset.

| Instance | Labelset         |
|----------|------------------|
| 1        | \{\lambda_2, \lambda_3\} |
| 2        | \{\lambda_1\}   |
| 3        | \{\lambda_1, \lambda_2, \lambda_3\} |
| 4        | \{\lambda_2, \lambda_4\} |

In above cases, actual data distribution may get lost and that negatively impacts and decreases the predictive accuracy of the model.

Label Powerset (LP) prepares unique label sets as single class so that this will become a multi-class classification task. At inference time, this method extracts the class with the highest probability in the data that actually represents a set of label. Table 5 shows the transformation of a dataset with Label Powerset method. One of the problem, with the LP method is that there can be a large number of examples associated with smaller number of labels that will lead to a huge class imbalance problem.

### TABLE 2. Each cell of the above table represents the average \(C_x(l)\) of the instances with \(l\).

| Label | beach | sunset | fall | field | mountain | urban |
|-------|-------|--------|------|-------|----------|-------|
| \(\bar{C}\) Cluster 1 | 1.432 | 0.926  | 1.470 | 2.000 | 1.785 | 5.532 |
| Cluster 2 | 1.600 | 1.500  | 2.047 | 6.188 | 2.256 | 1.000 |
| Cluster 3 | 1.250 | 5.265  | 6.055 | 1.571 | 0.500 | 1.090 |
| Cluster 4 | 1.044 | 1.607  | 1.214 | 1.936 | 4.761 | 2.000 |
| Cluster 5 | 4.863 | 1.333  | 1.000 | 2.333 | 1.444 | 1.956 |

In above cases, actual data distribution may get lost and that negatively impacts and decreases the predictive accuracy of the model.

Label Powerset (LP) prepares unique label sets as single class so that this will become a multi-class classification task. At inference time, this method extracts the class with the highest probability in the data that actually represents a set of label. Table 5 shows the transformation of a dataset with Label Powerset method. One of the problem, with the LP method is that there can be a large number of examples associated with smaller number of labels that will lead to a huge class imbalance problem.

### TABLE 2. Each cell of the above table represents the average \(C_x(l)\) of the instances with \(l\).

| Label | beach | sunset | fall | field | mountain | urban |
|-------|-------|--------|------|-------|----------|-------|
| \(\bar{C}\) Cluster 1 | 1.432 | 0.926  | 1.470 | 2.000 | 1.785 | 5.532 |
| Cluster 2 | 1.600 | 1.500  | 2.047 | 6.188 | 2.256 | 1.000 |
| Cluster 3 | 1.250 | 5.265  | 6.055 | 1.571 | 0.500 | 1.090 |
| Cluster 4 | 1.044 | 1.607  | 1.214 | 1.936 | 4.761 | 2.000 |
| Cluster 5 | 4.863 | 1.333  | 1.000 | 2.333 | 1.444 | 1.956 |

In summary, the following are the main contributions of the paper:

- Splitting of input space to multiple sub-spaces for optimal label estimation
- Selection of local examples using mutual/non-mutual examples w.r.t to an unknown instance
- Selection of labels using MAP estimate

The paper is organized as follows. Section II provides the literature review. Section III provides the details about the learning methods used in this study. Section IV discusses the experimental study, proposed approach, datasets and Evaluation Measures and results. Conclusion and future work are presented in Section V.
TABLE 4. Transformed multi-label dataset.

| Example | Label | Example | Label | Weight |
|---------|-------|---------|-------|--------|
| 1a      | λ₂    | 1a      | λ₂    | 0.5    |
| 1b      | λ₃    | 1b      | λ₃    | 0.5    |
| 2       | λ₁    | 2       | λ₁    | 1      |
| 3a      | λ₁    | 3a      | λ₁    | 0.33   |
| 3b      | λ₂    | 3b      | λ₂    | 0.33   |
| 3c      | λ₃    | 3c      | λ₃    | 0.33   |
| 4a      | λ₄    | 4a      | λ₄    | 0.5    |
| 4b      | λ₄    | 4b      | λ₄    | 0.5    |

(a) copy transformation    (b) doubled copy-weight transformation

TABLE 5. After LP method transformation.

| Instance | Labelset |
|----------|----------|
| 1        | λ₂₂,₃   |
| 2        | λ₂₁     |
| 3        | λ₂₁,₂₂,₃|
| 4        | λ₂₂,₄   |

RPC (Ranking by Pairwise Comparison) create chunks of dataset into \( \binom{k}{2} \) binary datasets that represents one dataset for each pair of labels, \((λ_i, λ_j)\), 1 ≤ i < j ≤ k. Each of these datasets will have the examples from the original dataset that belong to at least one of the corresponding labels (Table 7). It will be useful to train a binary classifier on it. In this way, we can also have the ranking of the labels based on the votes against labels through each classifier.

The authors in [8] considers that the ranking should have a natural 0-point for identifying the sets of appropriate classes based on the ranking of the labels. A calibrated label ranking which is an extended version of RPC is then proposed in which an extra label to the dataset is added and interpreted it as a calibration label, \(λ₀\) or neutral breaking point. Another interpretation of this can be a neutral point that differentiates the irrelevant labels from the relevant labels.

TABLE 6. Transformed dataset produced by BR method

| Example | Label  | Example | Label  | Example | Label  |
|---------|--------|---------|--------|---------|--------|
| 1       | −λ₁    | 1       | λ₂     | 1       | λ₃     |
| 2       | λ₁     | 2       | −λ₂    | 2       | −λ₃    |
| 3       | λ₂     | 3       | λ₃     | 3       | λ₀     |
| 4       | −λ₁    | 4       | λ₄     | 4       | −λ₄    |

(a) select-max    (b) select-min    (c) select-random

TABLE 7. Results of transformation after RPC method.

| Example | Label  | Example | Label  |
|---------|--------|---------|--------|
| (i) 1   | λ₁₋₁,₂| (ii) 1  | λ₁₋₁,₃|
| 2       | λ₁₋₁,₂| 2       | λ₁₋₁,₃|
| 4       | λ₁₋₁,₂| 4       | λ₁₋₁,₃|

| Example | Label  | Example | Label  |
|---------|--------|---------|--------|
| (iii) 2 | λ₁₋₁,₄| (iv) 4  | λ₂₋₃   |
| 3       | λ₁₋₁,₄| 3       | λ₃₋₄   |
| 4       | λ₋₁,₄ | 4       | λ₋₃,₄  |

B. ALGORITHM ADAPTATION METHODS

Several popular boosting techniques are available for multi-label classification [9] which are extended versions of AdaBoost for multi-label classification. AdaBoost.MH is intended to minimize hamming loss and the other is for ensuring the optimal ranking. In AdaBoost.MH, the labels of the instances are paired. The weight of all wrongly predicted label pairs is increased in every iteration. In contrast, AdaBoost.MR works on the same label pairs but it is focused on improving the label pairs that were ordered incorrectly.

Multi-class Multi-label Associative Classification (MMAC) [10] is an associative rule learning algorithm. It iteratively learns a rule by looking at the examples in the multi-label dataset and removes them so that in the next iteration, new rules can be identified. The labels will be ranked as per the support retrieved from the rules. A label that receives support from a larger number of rules will be given a higher order. [11] uses the same idea and incorporates it with lazy learning.

Some other techniques that shows promising results in this domain are neural networks and multi-layer perceptron (MLP) for multi-label datasets. The most widely used back-propagation in neural networks was modified to manage the multi-label data.

In MLP for multi-label data, every label has its own output node. Some Multi-class Multi-layer Perceptron (MMP) have been proposed in [12]. They worked on weight tuning in such a way that order ranking of labels was possible.

Several lazy learning based approaches have been proposed in the literature for both categories (i.e: problem transformation or algorithm adaptation) with kNN being the most popular for lazy learning. The differences between these approaches is the way of aggregation of label sets for the given examples. [3] is proposed by Spyromitros et. al which is basically equivalent to BR followed by kNN. The complexity of this technique is \(|L|\) times the cost of computing kNN. An empty set will be returned if algorithm does not get the support of the labels from atleast half of the neighbors. To overcome this issue, the same author proposes two extended version of BRkNN that helped to calculate the confidence of the labels properly.

\[
\text{Confidence}(λ) = \frac{1}{k} \sum_{j=1}^{k} I_{S_j}(λ)
\]
where, \( N \) is the set of labels for \( k \) nearest neighbors and,

\[
I_{N_j}(\lambda) = \begin{cases} 
1, & \text{if } \lambda \in N_j \\
0, & \text{otherwise} 
\end{cases} 
\tag{4}
\]

In this way, only the label with highest confidence will be returned. In their second extension, the average size of the label sets of \( \text{kNN} \) (average-size, \( s = \text{Confidence}(\lambda) = \frac{1}{k} \sum_{j=1}^{k} |N_j| \)) is estimated and \( s \) highest confident labels will be returned.

ML-kNN [4] is yet another most popular approach which is also based on BR. It uses MAP estimate for extracting the label set for a test instance. The method will first calculate the prior probability of the labels from the given dataset and for inference, it first fetches the nearest neighbors and then posterior probabilities will be calculated to get the final MAP estimate.

\[
Z_t^k = \arg \max_{b \in [0,1]} P(H_b^k|E_{C_t(\lambda)}^k), \quad l \in \mathcal{L} 
\tag{5}
\]

where, \( Z_t^k \) = label set predicted for instance \( t \), \( H_b^k(t) \) represents the occurrences that \( t \) possess label \( \lambda \) and does not possess label \( \lambda \), respectively. \( E_{C_t(\lambda)}^k \) is the occurrence which shows that exactly \( j \) instances of \( k \text{nn} \) test instance posses label \( \lambda \) and, \( C_t(\lambda) \) represents membership vector that accounts for all the neighbors of \( t \) associated to class \( \lambda \). This can be re-written as:

\[
Z_t^k = \arg \max_{b \in [0,1]} P(H_b^k)P(E_{C_t(\lambda)}^k|H_b^k), \quad l \in \mathcal{L} 
\tag{6}
\]

There may be errors in the input vector or output labels which may lead to failure of correctly predicting of multi-label. Sawsan Kanj and Fahed Abdalla in [13] address this issue by editing the existing training dataset and adapting the updated dataset with different multi-label classification methods. If the mean Hamming loss is less than a predefined threshold, then all instances with low hamming loss will be discarded from the original dataset.

The distance metric plays a vital role in computing how far away any two instances are. For this reason, LMO-KNN was introduced in [14] that determines an appropriate distance metric for dealing with multi-output tasks. This large margin metric learning project embeds input and output into the same space. Output vector is first compressed to lower dimensional space. Then both input and compressed output were projected into the same embedding space for finding a distance metric.

In CML-kNN [15], authors have studied the internal relationship between categorical labels. They have proposed two similarity measures Intra-Coupling Label Similarity (IaCLS) and Inter-Coupling Label Similarity (IeCLS). IaCLS will measure the similarity between two different labels in the space. On the other hand, IeCLS will capture the interaction of labels based on the co-occurrence of some value in the feature space.

To improve the prototype selection strategy in nearest neighbors, J. Calvo-Zaragoza et al [16] fetched the reduced set from the training data once in pre-processing stage. This reduced set is then used as fast recommending system which will propose some of the possible labels. The prototypes of these proposed labels will then be recovered for the final decision thereby speeding up the nearest neighbor classification process.

In [17], authors have enhanced the importance of local information of neighbors by splitting the dataset into five clusters and for every cluster with each label, they calculated the average count of the instances with and without label \( l \). By this, they proved that the distribution has a strong impact on the local information. In this way, the examples having equal confidences/votes among \( k \) nearest neighbors may possess different probabilities for having a label coming from different regions.

Local positive and negative correlation-based k-labelsets for multi-label classification [18] is an improved version of RAkEL. It randomly group the output labelset. These labels may be independent or weakly correlated. This behaviour may lead to a poor prediction of sub-classifiers. Label correlation is used to form \( k \)-labelset combinations. Modeling of local positive and negative label correlations will be done in conjunction with LP.

In [19], authors proposed a method which combines features specific to labels and their correlation. They achieved this by transforming the original feature space to low dimensional space that contains label specific features only. In this way, every label will get its own representation and then local correlation between each pair of labels is identified by nearest neighbor techniques. The label-specific features are gathered by compiling the related data from other label-specific features. The classification is done by binary classifier that built upon the label-specific features.

In [20], every example in the dataset has been assigned a relevance score with the labels it belongs to. This score will be used in the nearest neighbor classification with a voting-margin ratio. For producing a soft relevance score, modified fuzzy c-means is used. For classification, modified kNN for MLC is proposed. The FCM were modified in such way that it treats each class as a cluster. The Minkowski distance is used as described below:

\[
d(\bar{x}_i, \bar{w}_j) = \left( \sum_{j=1}^{d} |x_{ij} - w_{kj}|^f \right)^{\frac{1}{f}} 
\tag{7}
\]

where \( f \) is an integer and \( f \geq 1 \); \( \bar{x}_{ij} \) and \( \bar{w}_{kj} \) are the j-th components of \( \bar{x}_i \) and \( \bar{w}_j \) respectively. Changing \( f \) in above formula will cater to different shape of clusters.

Instance based logistic regression [21] is another technique that used the labels of neighbors as features for the test query. Similar techniques had produced great results in relational learning and collective classification. This approach helped in classification of labels considering the correlation between labels. For a particular value of \( k \), the confidence of each label within that region is calculated and appended into the dataset for classification.
Universal law of gravitation has been used in machine learning to solve various problems. Cheng in [22] presents an MLC lazy learner based on the idea of data gravitational model. MLGDC takes every example as an atomic data particle. The distance between two particles will be calculated by using:

\[ d_f(i, j) = \sqrt{\sum_{f \in F} \delta(x_{ij}, x_{jf})^2} \]

(8)

where \( x_{ij} \) and \( x_{jf} \) represent the value of the \( f \)th feature for particles \( i \) and \( j \), respectively with \( F \) as the feature space. The \( d_f \) is the HEOM (Heterogeneous Euclidean Overlap Metric) distance. A key feature in this approach is Neighborhood gravitational coefficient that is responsible to strengthen or weakness the gravitational force that a particle has on any test instance. The formula for NGC is described below:

\[ g_i = d_i^{w_i} \]

(9)

where \( d_i \) and \( w_i \) are the neighborhood-density that is the distribution of the particle in the neighborhood and neighborhood-weight is the probability of having dissimilar labels in the neighborhood of the particle \( i \).

After all the conversion of all instances into particles, the score of gravitational force between the test instance \( i \) and the \( j \)th particle will be computed as:

\[ f_{i,j} = \frac{g_{i}}{d_f(i,j)^2} \]

(10)

In [23], authors have proposed the idea of Shelly Nearest Neighbors (SNN). Let \( L = \{l_1, l_2, \ldots, q\} \) is the set of class labels, and \( D = (X_i, Y_j) | i = 1, \ldots, n \) is the multi-label data set. \( X_i \) is a vector with \( p \) features and \( Y_j \) is the labelset of \( j \)th instance. For a particular attribute, the left neighbors of an instance will be the ones whose \( j \)-th attribute is the multi-label data set. It works mostly for the datasets that have label hierarchy in them. An example of such datasets is expressed in 3 for the below sample label set:

For \( Y = \{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \lambda_6\} \)

In [6], authors conducted an experiment on local difference in neighborhood space and found that MLkNN has one short coming, i.e: MLkNN assigns the same probability to labels that belong to different regions in the input space. They further provided some statistics on MULAN repository and proved that the density of a label is different around the input space. Thus, they presented a technique called LAMLkNN (A Locally Adaptive Multi-Label \( k \)-Nearest Neighbor Algorithm) in which the original input space is divided into multiple clusters and then local differences are taken into account when calculating prior and posterior probabilities.

Local sets for multi-label instance selection [2], is primarily motivated by the concept of local sets given by Brighton and Mellish [25]. They proposed two techniques on the adaptation of local sets for multi-label data. One of them is aimed to clean the dataset and other is to reduce the dataset size to improve the overall predictions. They have provided comparative studies with other local set selection methods. Their strategies are improvements of LSSm and LSBo.

Random walk graph with KNN (MLRWKNN) [26] is yet another technique in this domain which works on the principal of graph theory by producing the set of vertices using a random walk for the nearest neighbor instances. It also generates the set of edges of label correlation. This method uses discrete and continuous features to learn the relationships between instances.

In MLG-kNN [27], authors have addressed the issue of high computation cost required by LM-kNN. The reason for high computation cost is due to the use of square hinge loss. They first transformed the original instance into the label space by least square regression and then ran the algorithm to learn the metric matrix in the new label space.

Another recent method in this category is [28] where authors have use linear ensemble of labels from instance and feature based nearest neighbors. The main novelty in their work is the use of an inverted index to compute cosine similarity on sparse datasets.

III. DISCRIMINATIVE ADAPTIVE SETS

A. KNN

\( k \)-Nearest Neighbors is a non-parametric lazy learning based classification algorithm that worked well in practice. It has been implemented in a variety of domains such as vision, protein, computational geometry, graphs and many more. An interesting application of this algorithm is online learning with a supervised setting. This algorithm by nature does not make underlying assumptions about the distribution, i.e it does not use training data for generalization; there will be no or a minimal training phase. By nature, the algorithm is widely applicable in many scenarios. Because it is non-parametric, it works by measuring proximity between query and examples in the dataset. It selects a particular number of neighbors from surrounding represented with \( k \). The classification is then carried out by voting techniques.
clusters in our dataset. These clusters can be explained by the following parameters:

1) A mean $\mu$ that defines its centre.
2) The width can be defined using co-variance $\Sigma$.
3) A probability known as mixing probability expresses the size of the Gaussian function.

The Gaussian density function is given by:

$$
N = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)
$$

where $x$ represents instances present in the distribution, $D$ is the number of dimensions of each instance, $\mu$ and $\Sigma$ represent the mean and co-variance, respectively.

GMM is an alternate for splitting the input space into multiple regions. However, there are a couple of advantages of using Gaussian mixture models over k-means. One of the limitations of $k - \text{means}$ algorithm that has overcome by GMM is that $k - \text{means}$ does not take variance into consideration. Variance in 2-dimensional space expresses the shape of the distribution. $k - \text{means}$ creates a circle around its mean whose radius can be identified by the member at the most distant point. As far as the data distributions that do not take circular/spherical shape are concerned, GMM comes into play. The clusters in GMM can take any oblong shape.

**B. K-MEANS**

K-means is a clustering algorithm that splits the input space into $k$ well separated and non-overlapping regions where each instance can be a member of any one group. The primary objective of this algorithm is to make data points similar within clusters while keeping the clusters as far as possible. It associates the instances to a cluster in a way that the sum of the squared distance between instances and the cluster’s centroid is minimum. In this way, the less variation in one cluster will produce more homogeneous data points inside a cluster.

The way k-means algorithm works is as follows:

1) Select number of clusters $k$.
2) Initialize centroids by selecting the $k$ data points randomly.
3) Continue iterating until centroids don’t alter.
4) Measure the sum of the square distance between all centroids and data points.
5) Assign each data point to the closest cluster.
6) Calculate the centroids for the clusters by taking all the data points belonging to each cluster on average.
7) Assigning the instances in a dataset to the closest cluster can be considered as E-step. The process of computing the centroid of each cluster will be the M-step.

**C. GAUSSIAN MIXTURE MODEL**

Gaussian mixture model (GMM) involves the mixture of multiple Gaussian distributions. Each distribution in GMM is identified by $k \in 1, \ldots, K$, where $K$ is the number of clusters in our dataset. These clusters can be explained by the following parameters:

1) A mean $\mu$ that defines its centre.
2) The width can be defined using co-variance $\Sigma$.
3) A probability known as mixing probability expresses the size of the Gaussian function.

The Gaussian density function is given by:

$$
N = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp \left( -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)
$$

where $x$ represents instances present in the distribution, $D$ is the number of dimensions of each instance, $\mu$ and $\Sigma$ represent the mean and co-variance, respectively.

GMM is an alternate for splitting the input space into multiple regions. However, there are a couple of advantages of using Gaussian mixture models over k-means. One of the limitations of $k - \text{means}$ algorithm that has overcome by GMM is that $k - \text{means}$ does not take variance into consideration. Variance in 2-dimensional space expresses the shape of the distribution. $k - \text{means}$ creates a circle around its mean whose radius can be identified by the member at the most distant point. As far as the data distributions that do not take circular/spherical shape are concerned, GMM comes into play. The clusters in GMM can take any oblong shape. GMM besides hard classification, also supports soft classification in the form of probabilities that define the membership of data points to each of $K$ clusters.

**D. MUTUALITY STRATEGIES**

1) MUTUAL NEIGHBOR SET

To identify the adaptive sets of instances, there is a concept of fetching mutual nearest neighbors within a lazy learning setting. This concept has already been used in single and multi-label classification [5]. The algorithm works in a similar fashion as of the original kNN method. The only difference is that during the instance search process, it does not directly select the instances from the neighborhood but it first acknowledge every neighbor. If the neighbor instance gives the heads up, then this instance will be added to adaptive sets and will be used for classification. In this way, it will ignore all noisy and irrelevant instances no matter how near the neighbor instance is. This will make the decision boundary more adaptive and robust even in case of k-means that usually takes spherical shape in the data space.

Formally, let $B_k(E)$ be the set of kNN for a new test example $E$. The set $B_{mut}(E)$ will have the elements $E_i$ that follows:

$$
E_i \in B_{mut}(E) \quad \text{if} \quad E_i \in B_k(E) \quad \text{and} \quad E \in B_k(E_i)
$$

This method requires the instances in $B_{mut}(E)$ to be among the nearest neighbors of the instances in $B_{mut}(E)$. In this way, it restricts the number of instances in $B_{mut}(E)$. 

---

**FIGURE 3.** Hierarchy of labels in HOMER.
2) NON-MUTUAL NEIGHBOR SET
Non-mutual neighbor set is another strategy that was introduced first in multi-class domain and adapted by multi-label community due to its effectiveness. The working principle of this method is quite similar to mutual neighbor set. The only difference is when collecting the neighbors from the surrounding, it does not only consider the direct neighbors but also considers the neighbors of the neighbor. In this way, the decision boundary will get broader and the bucket of adaptive sets will get richer because it has more relevant neighbors for taking the decision. Note that, this strategy might not work for small datasets as it requires a good number of instances.

Formally, the set of non-mutual nearest neighbors \( B_{\text{nmut}} \), consists of examples \( E_i \) which verify

\[
E_i \in B_{\text{nmut}}(E) \quad \text{if} \quad E_i \in B_k(E) \quad \text{and} \quad E \in B_k(E_i)
\]

3) CROSS VALIDATION
In order to evaluate our models, we have used a technique called K-fold cross validation. To re-sample the data, The K-fold cross validation consist of only one parameter i.e. k. It refers to the number of groups to be split for a given sample data. We have used used exhaustive cross-validation with \( k = 10 \), using an average score of each evaluation metric for the experiments.

4) EUCLIDEAN DISTANCE
To find out the similarity of instances in the data space, we have used Euclidean distance which is a straight-line distance between two points. Let \( a \) and \( b \) be the points in Euclidean space, then the distance between \( a \) and \( b \) will be the length of line segment joining them together and can be formulated as:

\[
d(a, b) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}
\]

5) PRIOR PROBABILITIES
Prior probability refers to the probability of an event before new data is collected. This probability will be updated as new instances come into the dataset. This helps to produce a more accurate outcome. This updated probability transforms into posterior probability which can be calculated using Bayes theorem mentioned below

\[
P(H_i^1) = (s + \sum_{i=1}^{m} \tilde{y}_i(l))/(sx2 + m)
\]

\[
P(H_0) = 1 - P(H_1^1);
\]

where ‘s’ is a smoothing factor; \( \tilde{y}_i(l) \) is the count vector; ‘m’ is the number of examples in the set; \( P(H|0) \) the probability of not observing an example with some label; \( P(H|1) \) the probability of observing an example with some label.

6) MAP ESTIMATE
One of the most popular approach for estimating the density is MAP (Maximum a Posteriori), which is a probabilistic framework. It works by computing the conditional probability of observing the data given a model weighed by a prior probability or belief. It involves calculating the conditional probability of one outcome given another outcome can be stated as follows:

\[
\tilde{y}_i(l) = \arg \max_{b \in \{0, 1\}} P(H_i^1|H_{b}^{l}) P(E_{C(l)}|H_{b}^{l})
\]

IV. EXPERIMENTAL STUDY
The performance of these techniques have been analyzed by means of a detailed experiment. These new techniques were evaluated and compared with state-of-the-art multi-label lazy learners, space partitioning, and instance selection techniques. The models are built by processing through cross-validation procedure. The experiment were conducted on a PC machine. We have used mutual and non-mutual sets for our experiments along with two famous clustering algorithms for space partitioning, k-means and Gaussian mixture model. This section is structured as follows: datasets used in experiments are detailed with their characteristics in Section A; secondly, the metrics used for evaluation of multi-label classification models are explained with their intuitions (Section B); the complete experimental setup is explained in Section C; several results of the complete experiments are discussed in Section D; and at the end, an overall conclusion and future work is presented in the concluding Section V.

A. DATASETS
In multi-label data, the number of labels assigned to each example may vary and that can be very small as compared to the total number of labels present in the dataset. However, there may be instances where the labels associated with them are quite large. As a result this could become a parameter that can influence different multi-label methods. In [7], the authors have proposed the concept of label cardinality and density. According to this, the average number of labels assigned to each instance in the dataset \( D \) can be termed as Label Cardinality.

\[
\text{Label Cardinality} = \frac{1}{m} \sum_{i=1}^{m} |Y_i|
\]

The average number of labels associated with examples in \( D \) divided by total number of labels \( q \) is regarded as the Label Density.

\[
\text{Label Density} = \frac{1}{m} \sum_{i=1}^{m} \frac{|Y_i|}{q}
\]

In general, label cardinality has no relation with the total number of labels \( q \) in the classification problem. It is basically used to get the alternative labels present in the training for characterizing the examples. The label density takes into account the number of labels associated with each example.
account the total number of labels present in the dataset. With this in hand, two datasets with same cardinality but with a difference in the number of labels can react differently to the same multi-label classification method.

The datasets used for evaluation are summarised in Table 8. These datasets are available online on Mulan repository. Some characteristics of the table are shown which include name, domain, number of features, number of instances, number of labels, label cardinality (the average number of labels of each example), and label density (label cardinality divided by the number of labels). These datasets are used as it is and there with no transformation. The five benchmark datasets we used in our experiments are already used in various studies and evaluation of methods in multi-label domain. Our selection of problem domains are based on a different scale and from a variety of application domains. From Table 8, we can infer that the size of the dataset varies from 593 to 2417 instances. The number of features ranges from 72 to 1449; labels from 6 to 45 and the average number of labels per example is between 1.074 to 4.237. All these datasets are already pre-divided into training and testing parts, so there is no split required. However, we have applied cross validation on the training set to minimize the bias in the model. With respect to the original split, training set usually has two-third of the whole dataset while test set has the remaining one-third.

### Table 8. Benchmark dataset with their characteristics.

| Dataset   | Domain   | Instances | Features | Labels | Cardinality |
|-----------|----------|-----------|----------|--------|-------------|
| Yeast     | Biology  | 2417      | 103      | 14     | 4.237       |
| Medical   | Text     | 978       | 1449     | 45     | 1.245       |
| Emotions  | Music    | 593       | 72       | 6      | 1.869       |
| Scene     | Multimedia | 2407   | 294      | 6      | 1.074       |
| Genbase   | Biology  | 662       | 1186     | 27     | 1.232       |
| Enron     | Text     | 1702      | 1001     | 53     | 3.378       |
| Bibex     | Text     | 7395      | 1836     | 159    | 2.402       |

Three different domains have been considered with these datasets. This includes multimedia, text, and biology. Yeast is one of the most popular dataset discussed in multi-label literature. Every record in this dataset represents a gene associated with 14 biological functions. On the other hand, scene and emotions are the two datasets that belong to the multimedia category. In emotions, every instance is a piece of music that can be labelled as relaxing-calm, quiet-still, sad-lonely, angry-aggressive, amazed-surprised, and happy-pleased. Scene is among popular classification datasets. Each instance of this dataset is annotated with 6 different labels (beach, sunset, field, fall-foliage, mountain, and urban). Medical dataset belongs to the domain of text categorization. This dataset was used in the Medical Natural Language Processing Challenge3 in 2007. Every instance of medical dataset represents a document with free-text summary of a patient’s symptoms history. The purpose of classification is to label each example (document) in the dataset with diseases from International Classification of Diseases.

The label correlations of these dataset have been depicted in Figure 4 to 8.

![Figure 4. Label correlation in yeast dataset.](image-url)

![Figure 5. Label correlation in medical dataset.](image-url)

![Figure 6. Label correlation in emotion dataset.](image-url)

### B. EVALUATION MEASURES

The evaluation methods for multi-label classification are different from those of multi-class classification. In this section, we will discuss some common evaluation measures that were used for multi-label domain.
1) HAMMING LOSS
Hamming loss comes under the category of bipartitions evaluation measure. It calculates the average difference between the predicted and actual label sets over all instances of the dataset. There are other types of measures in bipartitions that evaluate each label separately by averaging over all labels subsequently and so they are known as label-based evaluation measures. The Hamming loss is defined as follows:

$$\text{Hamming Loss} = \frac{1}{m} \sum_{i=1}^{m} |Y_i \Delta Z_i| / M$$

where $\Delta$ stands for the symmetric difference of two sets, which can be interpreted as the equivalent to XOR operation in Boolean algebra.

2) SUBSET ACCURACY
Subset accuracy can be defined as the rate of perfectly classified instances. It requires that the predicted set of labels to be an exact match of the true set of labels for every instance. The subset accuracy is defined as follows:

$$\text{Subset Accuracy} = \frac{1}{m} \sum_{i=1}^{m} |Y_i = Z_i| / M$$

3) MICRO PRECISION
Precision is the proportion of predicted correct labels to the total number of actual labels, averaged over all instances. We have used micro averaged precision for evaluation in this experiments, as it is very good in determining the effect of data-set’s size when the data-set sizes are variable. The micro-precision is defined as follows:

$$\text{Micro-Precision} = \frac{1}{m} \sum_{i=1}^{m} \frac{|Y_i \cap Z_i|}{Z_i}$$

4) MICRO RECALL
Recall is the proportion of predicted correct labels to the total number of predicted labels, averaged over all instances. We have used micro average recall for evaluation for the same reason on which we opted for micro-averaged precision. The micro-recall is defined as follows:

$$\text{Micro-Recall} = \frac{1}{m} \sum_{i=1}^{m} \frac{|Y_i \cap Z_i|}{Y_i}$$

5) F-MEASURE
Micro F-measure counts the total true positive, false positive, and false negatives globally. It is simply the harmonic mean between micro recall and micro precision. It can be formulated as:

$$\text{Micro F1} = \frac{2 \cdot \text{micro}\_\text{precision} \cdot \text{micro}\_\text{recall}}{\text{micro}\_\text{precision} + \text{micro}\_\text{recall}}$$

In multi-class classification, the higher the value of accuracy, precision, recall and F1-score, it shows the better the performance of the learning algorithm.

C. PROPOSED APPROACH

1) EXPERIMENTAL SETUP
The experimental study was performed with 10-fold cross validation. Each dataset is iterated 10 times. In each iteration, one fold or a subset is used for model testing while 9 other folds are used for model training. This process is repeated until every fold gets a chance to become a test set. After all iterations, the final results are gained by taking out the average results of all executions.

The flow of our experiments is depicted in Fig. 9. In this figure, for space partitioning, we have used two techniques; one is K-means and the other one is Gaussian Mixture Model (GMM). The intuition behind using these techniques is that K-means (as described in Section 4.2) is a famous clustering algorithm which follows an iterative procedure to estimate data partitions with their center of mass (aka centroids). This algorithm computes a proximity measure from each data point to the centroid. In this way, it finds out the relevancy of the data point to the cluster. GMM, on the other hand, is another popular algorithm for data partitioning. The beauty of GMM over K-means is that it can create clusters with very oblong shapes, unlike K-means which can only create spherical cluster shapes or hyper-spherical clusters for high
dimensional data. Since GMMs are based on mixture model, they can determine the presence of sub-population within an overall population. This matches with the nature of the problem we are dealing with. The primary motivation behind splitting the input space into multiple sub-spaces is to take local information into account when doing classification. Since, it is possible that a label may have various regions with different influences in a single space. We should take these influences into account during searching good neighbors in the neighborhood.

Afterwards, the prior probabilities of the labels will be computed for every region and within each region for every label. This will be used when applying MAP principle during classification. We have used Laplace Smoothing in our experiments for both prior and probabilities. Since original input space has been broken down into multiple sub-spaces, we now have a single representative (centroids) from every region. These centroids will be used for identifying the most relevant region for new examples by some proximity measure (Euclidean distance in our case). Having found the right region, we now have some relevant neighbors to identify. Two approaches have been used for this purpose: mutual sets and non-mutual sets.

The next step is to calculate the posterior probability of the mutual/non-mutual sets in their regions. We used the MAP (maximum a posteriori) estimate to calculate the final probabilities of the labels that will be used to classify them.

The performance of the sets is evaluated using some traditional and state-of -the-art techniques. These include MlkNN [4], BrkNN [3], Rakel [29], HOMER [24], IBLR [21], LAMLKNN [6], HDLSSm [2] and HDLSSo [2]. Most of them are based on lazy learning, while others are based on space partitioning. With the exception LAMLKNN, HDLSSm and HDLSSo, all approaches are available in mulan. The source code for the rest of them are available online. These approaches were selected because they were considered to be the best classifiers in benchmark [30] and other review papers. The value of k in nearest neighbor was used from 8 to 12 as this range is standard for mulan repository (inferred from many research papers mentioned in Section 3.2). Similarly the value of the number of clusters is varied from 3 to 20. This selection comes by hitting and checking the data sets listed in the Section 5.1. We compared our approach with nine algorithms including MLKNN, BRkNN-a, BRkNN-b, Rakel, LAMLKNN, HOMER, IBLR, HDLSSm and HDLSSo. Detailed pseudo-code is give below:

Algorithm 1: **DASMLKNN**

\[
\begin{align*}
&\text{Compute the clusters for given input space using k-means } (c_1, c_2, c_3, \ldots, c_m); \\
&\text{kmeans } = \text{KMeans}(c, X); \\
&\text{Computing prior probabilities } P(H_l|\text{region}) \text{ for every region (cluster);} \\
&\text{for region } \in c \text{ do} \\
&\quad \text{for } l \in Y \text{ do} \\
&\quad\quad P(H^t_l|\text{region}) = (s + \sum_{i=1}^{m} y_{xi}[\text{region}(l)])/(sx2 + |\text{region}|); \\
&\quad\quad P(H_0^t|\text{region}) = 1 - P(H^t_l|\text{region}); \\
&\text{end} \\
&\text{Identify the nearest cluster } (c) \text{ for new instance from previously extracted regions;} \\
&\text{Identify } k \text{ nearest neighbors } N(t) \text{ using mutual nearest neighbor strategy;} \\
&\text{Calculate posterior probabilities of every label in } Y; \\
&\text{for } l \in Y \text{ do} \\
&\quad \tilde{y}_t(l) = \arg \max_{b \in [0, 1]} P(H_b^t|\text{region})P(E_{C,b}^t|H_b^t); \\
&\text{end}
\end{align*}
\]

D. RESULTS AND DISCUSSION

In this section, all the results from our experimental study are gathered and compared with other methods. The significance
of our method is also discussed in detail. The comparison is based on the performance measures explained in the previous section.

1) COMPARATIVE ANALYSIS

First of all, we want to discuss the experimental results on a group of algorithms which are using some improvements on kNN. These are MLkNN, BRkNN-a, BRkNN-b, RaKEL, LAMLKNN, and DASMLKNN. Consider the Table 9 and Table 10, it is clear from the table that in term of hamming loss our proposed algorithm performed quite well on three datasets these are Yeast, Enron and Bibtex. Its working is comparable to MLkNN and RaKEL of all other datasets.

| TABLE 9. Hamming loss for DASMLKNN on Yeast,Medical,Emotion and Enron dataset. |
|-----------------------------------------------|
|          | Yeast | Medical | Emotion | Enron          |
| MLkNN    | 0.208±0.020 | 0.019±0.021 | 0.399±0.143 | 0.014±0.001 |
| BRkNN-a  | 0.204±0.026 | 0.226±0.033 | 0.297±0.152 | 0.021±0.012 |
| BRkNN-b  | 0.347±0.092 | 0.312±0.072 | 0.414±0.122 | 0.024±0.016 |
| RaKEL    | 0.211±0.072 | 0.020±0.016 | 0.286±0.216 | 0.011±0.011 |
| LAMLKNN  | 0.208±0.022 | 0.021±0.073 | 0.241±0.342 | 0.024±0.011 |
| DASMLKNN | 0.201±0.016 | 0.022±0.012 | 0.290±0.210 | 0.015±0.011 |

Similarly, the next table 11 and 12 it can be seen that our proposed approach performed quite well in term of micro-precision. It clearly outperforms BRkNN and LAMLKNN and very comparative to MLkNN and RaKEL. It is clearly evident that in Yeast, Enron and Bibtex where the label cardinality is quite our algorithm performed quite well. In term of micro-recall Table 13 and Table 14, the performance of our proposed approach on all benchmark datasets is comparable. In a subset of dataset like medical,enron and bibtex which are of text type and from different domains. Our approach able to produce the higher micro-recall for all these datasets and perform comparable to MLkNN and RaKEL which are considered state-of-the-art in this domain.

Finally, we would like to discuss subset accuracy, which is quite a strict measure since it requires the predicted set of labels to be an exact match of the true set of labels. It also penalizes prediction that may be almost correct or/and totally wrong. In term of subset accuracy our approach could not perform very well as can be seen from the Table 15 and Table 16, it is not very surprising as due to it strict nature the benchmark datasets with high-cardinality ratio is ultimately affected by the rigid ordering of the target labels. Despite, the subset accuracy is comparable to all other variants of

| TABLE 12. Micro Precision for DASMLKNN on Scene, Genbase and Bibtex datasets. |
|-----------------------------------------------|
| Scene | Genbase | Bibtex |          |
| MLkNN  | 0.711±0.004 | 0.742±0.022 | 0.902±0.014 |
| BRkNN-a | 0.587±0.013 | 0.553±0.041 | 0.874±0.028 |
| BRkNN-b | 0.583±0.033 | 0.561±0.013 | 0.890±0.022 |
| RaKEL  | 0.664±0.027 | 0.584±0.034 | 0.897±0.011 |
| LAMLKNN | 0.637±0.025 | 0.661±0.011 | 0.876±0.031 |
| DASMLKNN | 0.656±0.011 | 0.682±0.031 | 0.898±0.029 |

| TABLE 13. Micro Recall for DASMLKNN on Yeast,Medical, Emotion and Enron datasets. |
|-----------------------------------------------|
| Yeast | Medical | Emotion | Enron |          |
| MLkNN  | 0.213±0.011 | 0.512±0.012 | 0.682±0.016 | 0.092±0.011 |
| BRkNN-a | 0.158±0.012 | 0.431±0.014 | 0.589±0.014 | 0.061±0.014 |
| BRkNN-b | 0.110±0.031 | 0.431±0.014 | 0.592±0.011 | 0.077±0.021 |
| RaKEL  | 0.105±0.023 | 0.381±0.022 | 0.610±0.009 | 0.081±0.019 |
| LAMLKNN | 0.132±0.021 | 0.410±0.023 | 0.587±0.012 | 0.027±0.013 |
| DASMLKNN | 0.163±0.012 | 0.432±0.010 | 0.648±0.011 | 0.051±0.021 |

| TABLE 14. Micro Recall for DASMLKNN on Scene, Genbase and Bibtex datasets. |
|-----------------------------------------------|
| Scene | Genbase | Bibtex |          |
| MLkNN  | 0.642±0.023 | 0.432±0.009 | 0.082±0.013 |
| BRkNN-a | 0.561±0.010 | 0.401±0.021 | 0.067±0.013 |
| BRkNN-b | 0.551±0.022 | 0.399±0.033 | 0.075±0.013 |
| RaKEL  | 0.555±0.021 | 0.401±0.021 | 0.072±0.021 |
| LAMLKNN | 0.513±0.023 | 0.387±0.030 | 0.052±0.011 |
| DASMLKNN | 0.543±0.023 | 0.411±0.022 | 0.041±0.017 |

| TABLE 15. Subset Accuracy for DASMLKNN on Yeast,Medical, Emotion and Enron dataset. |
|-----------------------------------------------|
| Yeast | Medical | Emotion | Enron |          |
| MLkNN  | 0.272±0.011 | 0.512±0.012 | 0.682±0.016 | 0.092±0.011 |
| BRkNN-a | 0.161±0.018 | 0.431±0.014 | 0.589±0.014 | 0.061±0.014 |
| BRkNN-b | 0.102±0.022 | 0.431±0.014 | 0.592±0.011 | 0.077±0.021 |
| RaKEL  | 0.103±0.023 | 0.381±0.022 | 0.610±0.009 | 0.081±0.019 |
| LAMLKNN | 0.121±0.021 | 0.410±0.023 | 0.587±0.012 | 0.027±0.013 |
| DASMLKNN | 0.160±0.017 | 0.430±0.021 | 0.641±0.013 | 0.031±0.018 |
kNN for multi-label classification except for MLkNN which perform better in this case.

### TABLE 16. Subset Accuracy for DASMLKNN on Scene, Genbase and Bibtex datasets.

|       | Scene       | Genbase     | Bibtex     |
|-------|-------------|-------------|------------|
| MLkNN | 0.642 ±0.023 | 0.432 ±0.009 | 0.088 ±0.013 |
| BRkNN-a| 0.561 ±0.010 | 0.401 ±0.021 | 0.067 ±0.013 |
| BRkNN-b| 0.551 ±0.022 | 0.399 ±0.033 | 0.075 ±0.013 |
| RaKeL | 0.555 ±0.012 | 0.401 ±0.011 | 0.077 ±0.021 |
| LAMLkNN | 0.513 ±0.023 | 0.387 ±0.030 | 0.052 ±0.011 |
| DASMLkNN | 0.512 ±0.010 | 0.421 ±0.022 | 0.041 ±0.017 |

2) STATISTICAL ANALYSIS

Fig 10. organizes an intuitive reflection of the winning ratios in the form of spyder charts when compared to previous multi-label works. By looking at the figure, we can infer that DASMLKNN is highly competitive for approaches concerning yeast, emotions, and genbase datasets. DASMLKNN has outperformed almost all techniques on these datasets. Comparing DASMLKNN with MIkNN and Rakel, it got 100% winning ratio on yeast, emotions and genbase but under performed for other two datasets (scene and medical). It is also completely superior to BrKNN-b and BrKNN-a, the only exception is on medical dataset where there are no wins in case of BrKNN-a. LAMLkNN on the other hand, performed worst when compared to DASMLKNN only on yeast dataset. It has outperformed DASMLKNN on all other datasets. Fig 11 shows comparison of DASMLKNN with HOMER, IBLR, HDLSSm and HDLSSo techniques. It can be inferred from the given figure that DASMLKNN has outperformed HOMER in three out of five datasets, whereas, IBLR has beaten DASMLKNN in four datasets. The only dataset that won in this race is medical. HDLSSm and HDLSSo has similar artifacts; both of them have failed in yeast, scene, and genbase, but beat DASMLKNN on other two datasets. From the above details, it can be concluded that DASMLKNN has showed progress on almost all datasets. As a consequence of the “no free lunch” theorem [31], the discrepancy in their actions can be explained.

3) RESULTS BASED ON EVALUATION MEASURES

Figures 10 and 11 summarizes the results of the experiments along with some more details on evaluation measures in figures 12 to 26. These figures show the comparison between the techniques that use k-nearest neighbors and other state-of-the-art techniques commonly discussed in multi-label literature.(Some of these techniques relate to our approach as they are based on instance selection techniques (HDLSSm and HDLSSo) and others for taking local information into account for label classification (IBLR)). For LAMLkNN,
the authors have fixed the value of $k$ to 10. Therefore, we have compared our best value only with LAMLkNN. The reason behind taking the range of $k$ from 8 to 10 is that it seems to be the standard for MULAN datasets (inferred from many papers that worked with lazy techniques on MULAN).

Looking at the results, it can be concluded that DASMLKNN outperformed all the methods included in our study for the dataset that belongs to biological domain that has functional classes of genes (Yeast) when Hamming loss is compared. Similar observations can be drawn from Emotion (collection of songs from different categories) and genbase
Our proposed technique is highly competitive on datasets having strong cardinalities and with good number of labels.

At a glance, it can be seen that DASMLKNN is best suited for Hamming Loss measure. In addition, it also behaves moderately for F-Measure throughout the datasets on which Hamming loss works best. As expected, the worst algorithm in this family is BrKNN-b due to negligence of labels with lower confidence. Fig 12 to 27 shows the consistency of hamming loss and F-Measure on multiple values on \( k \).

The reason why our approach is not balanced when taking both measures at a time is because of the metric definition. Zhang in [32] discussed this in detail. In his view, the current multi-label metrics take diverse aspects into consideration when dealing with the performances of different natures. Multi-label classifiers try to learn from training examples by implicitly or explicitly optimizing one particular metric. Different studies demonstrate that methods that try to maximize one measure would not be good in terms of performance when compared to others. The solution for this relies on optimizing surrogate loss function rather than the original loss function directly. We would like to explore surrogate loss function in our future work.

There were two more setups we have worked on. One is using non-mutual neighbors instead of mutual neighbors. In this setup, all those neighbors of the instances are included in the classification decisions that are direct neighbors of the test instances. In this way, the decision boundary will become more flexible and will allow more instances to take part in the classification. In another setup of this work, we have used Gaussian Mixture Model (GMM) for clustering the regions of the input space. The intuition behind using Gaussian Mixture is that they can take any oblong shape in the input space as compared to K-means that can only produce spherical clusters. Unfortunately both of these strategies have not produced good results on our benchmark datasets. The reason behind this is that the size of our benchmark datasets is not too large. The clusters created using GMM or adaptive sets fetched with
non-mutual strategy both require reasonably large quantity of instances to be in hand for better performance. Although these strategies have not worked on the datasets, they can be analyzed for datasets that have bigger input space. Due to this, we have omitted the results produced by these strategies in this report.

V. CONCLUSION AND FUTURE WORK

In this paper, we have attempted to improve the lazy learning methods for solving the multi-label classification problem. The proposed method has used mutuality strategy to grab the adaptive neighbors in the field and take local information of the instances into account by splitting the input space into multiple smaller and more relevant spaces. The influence of every label in different region of the space will be different. The approach has been evaluated against few $k$nn based instance selection based and few other techniques that try to take the local information for multi-label classification. A broad range of datasets from different domains have been selected for experiments. The results revealed that our proposed approach has almost outperformed all the state-of-the-art lazy learning based approaches. Multi-label learning is gaining incredible attention in the research community. There are many aspects in this domain which are still unexplored. Our incorporation of mutuality strategies with local information can be useful for future researches such as the development of new methods that besides capturing adaptive neighbors, also apply different instance selection procedures on the dataset. Generally, there is one common disadvantage when we talk about the instance based methods and that is the training time when the size of the dataset is much higher. For these problems, we can apply several reduction techniques as they will be applied only once to the original dataset.

As a future line of research, we have proposed the following things:

1) One of the disadvantages of lazy learning techniques is that they have large memory requirements. They have to search over all data points in a dataset. Therefore, the classification process is slow. Without applying any instance selection/editing process, the training set also contains noisy instances which can decrease the generalization accuracy. k-d trees is one option through which the search of the nearest neighbor can be tuned. They only benefit by reducing the search time. They do not reduce the storage overhead and neither reduce the noise from the dataset.

2) The performance of our proposed approach is highly dependent on how good our sub-spaces are. Finding optimal sub-spaces with some nature inspired algorithm might increase the accuracy.

3) We would like to apply the combination of non-mutual nearest neighbor for finding adaptive sets and Gaussian Mixture Model for extracting the optimal sub-spaces on datasets that have larger input space with strong cardinalities.

REFERENCES

[1] M. S. Sorower, “A literature survey on algorithms for multi-label learning,” Tech. Rep., 2010.
[2] À. Armai-González, J.-F. Diez-Pastor, J. J. Rodríguez, and C. García-Osorio, “Local sets for multi-label instance selection,” Appl. Soft Comput., vol. 68, pp. 651–666, 2018.
[3] E. Spyromitros, G. Tsoumakas, and I. Vlahavas, “An empirical study of lazy multilabel classification algorithms,” in Artificial Intelligence: Theories, Models and Applications, J. Darzentas, G. A. Vouros, S. Vosnakis, and A. Armellos, Eds. Berlin, Germany: Springer, 2008, pp. 401–406.
[4] M.-L. Zhang and Z.-H. Zhou, “ML-KNN: A lazy learning approach to multi-label learning,” Pattern Recognit., vol. 40, no. 7, pp. 2038–2048, Jul. 2007.
[5] I. C. Chen, N. Spaulo, J. Valverde-Rebaza, and M. C. Monard, “Lazy multi-label learning algorithms based on mutuality strategies,” J. Intell. Robotic Syst., vol. 80, no. 1, pp. 261–276, 2015.
[6] D. Wang, J. Wang, F. Hu, L. Guoqing, and X. Zhang, “A locally adaptive multi-label k-nearest neighbor algorithm,” in Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining, 2018, pp. 81–93.
[7] G. Tsoumakas, I. Katakis, and I. Vlahavas, Mining Multi-label Data. Boston, MA, USA: Springer, 2010, pp. 667–685.
[8] J. Fünnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, “Multilabel classification via calibrated label ranking,” J. Mach. Learn., vol. 73, no. 2, pp. 133–153, Nov. 2008.
[9] R. E. Schapire and Y. Singer, “BoosTexter: A boosting-based system for text categorization,” Mach. Learn., vol. 99, nos. 2–3, pp. 135–168, Mar. 2000.
[10] F. A. Thabet, P. Cowling, and Y. Peng, “MMAC: A new multi-class, multi-label associative classification approach,” in Proc. 4th IEEE Int. Conf. Data Mining (ICDM), Nov. 2004, pp. 217–224.
[11] A. Veloso, W. Meira, M. Gonçalves, and M. Zaki, “Multi-label lazy associative classification,” in Knowledge Discovery in Databases: PKDD, J. N. Kok, J. Koronacki, R. Lopez de Mantaras, S. Matwin, D. Mladenić, and A. Skowron, Eds. Berlin, Germany: Springer, 2007, pp. 605–612.
[12] K. Crammer and Y. Singer, “A family of additive online algorithms for category ranking,” J. Mach. Learn. Res., vol. 3, pp. 1025–1058, Mar. 2003.
[13] S. Kanji, F. Abdallah, T. Denœux, and K. Tout, “Editing training data for multi-label classification with the k-nearest neighbor rule,” Pattern Anal. Appl., vol. 19, pp. 145–161, Feb. 2016.
[14] W. Liu and I. W.-H. Tsang, “Large margin metric learning for multi-label prediction,” in Proc. AAAI, 2015.
[15] C. Liu and L. Cao, “A coupled k-nearest neighbor algorithm for multi-label classification,” Advances in Knowledge Discovery and Data Mining, T. Cao, E.-P. Lim, Z.-H. Zhou, T.-B. Ho, D. Cheung, and H. Motoda, Eds. Cham, Switzerland: Springer, 2015, pp. 176–187.
[16] J. Calvo-Zaragoza, J. J. Valero-Mas, and J. R. Rico-Juan, “Improving KNN multi-label classification in prototype selection scenarios using class proposals,” Pattern Recognit., vol. 48, pp. 1608–1622, May 2015.
[17] D. Wang, J. Wang, F. Hu, L. Li, and X. Zhang, “A locally adaptive multi-label k-nearest neighbor algorithm,” in Advances in Knowledge Discovery and Data Mining, D. Phung, V. S. Tseng, G. I. Webb, B. Ho, M. Gani, and L. Rashidi, Eds. Cham, Switzerland: Springer, 2018, pp. 81–93.
[18] G. Nan, Q. Li, R. Dou, and J. Liu, “Local positive and negative correlation-based k-labelsets for multi-label classification,” Neurocomputing, vol. 318, pp. 90–101, Aug. 2018.
[19] W. Weng, Y. Lin, S. Wu, Y. Li, and Y. Kang, “Multi-label learning based on label-specific features and local pairwise label correlation,” Neurocomputing, vol. 273, pp. 383–394, Jan. 2018.
[20] X. Lin and X. wen Chen, “Mt-KNN: Soft relevance for multi-label classification,” in Proc. CIKM, 2010, pp. 349–358.
[21] W. Cheng and E. Hüllermeier, “Combining instance-based learning and logistic regression for multilabel classification,” Mach. Learn., vol. 76, nos. 2–3, pp. 211–225, 2009.
[22] O. Reyes, C. Morell, and S. Ventura, “Effective lazy learning algorithm based on a data gravitation model for multi-label learning,” Inf. Sci., vols. 340–341, pp. 159–174, May 2016.
[23] H. Liu, X. Wu, and S. Zhang, “Neighbor selection for multilabel classification,” Neurocomputing, vol. 182, pp. 187–196, Mar. 2016.
[24] R. Cerri, R. C. Barros, and A. C. P. L. F. de Carvalho, “Hierarchical multi-label classification using local neural networks,” J. Comput. Syst. Sci., vol. 80, no. 1, pp. 39–56, Feb. 2014.
[25] H. Brighton and C. Mellish, “Advances in instance selection for instance-based learning algorithms,” Data Mining Knowl. Discovery, vol. 6, no. 2, pp. 153–172, 2002.
[26] Z.-W. Wang, S.-K. Wang, B.-T. Wan, and W. W. Song, “A novel multi-label classification algorithm based on K-nearest neighbor and random walk,” *Int. J. Distrib. Sensor Netw.*, vol. 16, no. 3, Mar. 2020, Art. no. 155014772091189.

[27] J. Ma and S. Zhou, “Metric learning-guided k nearest neighbor multilabel classifier,” *Neural Comput. Appl.*, pp. 1–15, Jul. 2020.

[28] L. Feremans, B. Cule, C. Vens, and B. Goethals, “Combining instance and feature neighbours for extreme multi-label classification,” Tech. Rep., Feb. 2020.

[29] G. Tsoumakas and I. P. Vlahavas, “Random k-labelsets: An ensemble method for multilabel classification,” in *Proc. ECML*, 2007, pp. 406–417.

[30] A. Pakrash, D. Greene, and B. M. Namee, “Title benchmarking multilabel classification algorithms,” Tech. Rep., 2017.

[31] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[32] M.-L. Zhang and Z.-H. Zhou, “A review on multi-label learning algorithms,” *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.

**MUHAMMAD RAFI** (Member, IEEE) was born in Karachi, Pakistan. He received the B.S. and M.S. degrees from the FAST-Institute of Computer Science affiliated with the University of Karachi, Pakistan, in 1996 and 2000, respectively, all in computer science, and the Ph.D. degree in computer science, in 2017. He has more than ten years of experience in software development and also working as a Consultant for local software industry. His current research interests include algorithm development, machine learning, information retrieval, text/data mining, time series analysis, and natural language processing. He has received several travel grant awards for presenting his work at the top conferences. He has served as a Judge and a Technical Quality Review Team at many versions of IIEEXtreme Programming competitions. He has served as a reviewer for various international journals of high impact.

**MUHAMMAD ATIF TAHIR** has over 14 years’ research experience in data analysis, computer vision, and machine learning. Before joining NUCES, he worked as a Senior Lecturer with Northumbria University, U.K. He is currently a Professor and the Director of the Video Surveillance Laboratory, which is one of the affiliated laboratory of National Centre for Big Data and Cloud Computing sponsored by the Planning Commission of Pakistan and HEC. He has more than 50 publications in top quality international journals and conferences, such as IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Journal of Machine Learning Research*, and *Pattern Recognition*. His research interests include image analysis and classification for medical and security applications, face recognition, multi-label classification, and event detection. His method on image retrieval using kernel discriminant analysis ranked first in many international competitions on concept detection in multimedia data, such as Pascal 2009 and ImageCLEF 2010.