DOI: http://dx.doi.org/10.26483/ijarcs.v15i2.7068

ISSN No. 0976-5697

Volume 15, No. 2, March-April 2024

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

An Ensemble Methods of Predicting the New Labels with Concept Drift from a High-Dimensional Data Stream

S.K. Komagal Yallini
Research Scholar
Department of Computer Science
Sri Ramakrishna College of Arts and Science,
Coimbatore, Tamil Nadu, India

Dr. N. Mahendiran
Assistant Professor,
Department of Computer Science,
Sri Ramakrishna College of Arts and Science,
Coimbatore, Tamil Nadu, India

Abstract: Multi-Label Learning (MLL) has arisen in data engineering to identify instances based on a specific feature associated with a collection of labels. Adaptive learning necessitates classifying features with New Labels (NLs) if a data stream contains newer perspectives. As a result, an MLL with Emerging Multiple NLs (MuEMNL) and managing High-Dimensional data streams (MuEMNLHD) approaches were developed that divides the NL sets into multiple NLs for efficient classification. However, it did not handle concept drift issues when huge amounts of data arrived at high speeds using limited resources. Hence, this article proposes an adaptive ensemble learning approach to cope with a huge amount of data streams and solve concept drift issues by constructing a MuEMNL-Ensemble Neural Network (ENN) rather than a random forest classifier. It defines the number of NNs in the ensemble, whether or not they use constructive pruning, how many hidden nodes each NN uses, and how many training samples are used to train each NN independently. Also, to solve the concept drifts, pairwise and non-pairwise diversity measures are analyzed while constructing ensemble NN for efficient training using the entire learning examples. Moreover, the tradeoff between the NN's precision and diversity is maintained simultaneously. At last, the test outcomes reveal that the proposed approach attains a better performance contrasted with the existing MLL approaches.

Keywords: multi-label learning, MuEMNL, concept drift, ensemble learning, neural network, diversity

I. INTRODAUCTION

Traditional supervised learning has become the most widely used artificial intelligence concept, with every aspect known as a distinctive feature vector connected with specific labels. Despite this learning is widespread, a distinctive trait might involve many labels in various contexts. For example, various labels often describe a scene shot [1]; a paper may have many topics [2]; and an audio recording may fit into multiple categories [3]. To cope with this kind of information, MLL was invented, a learning concept that has recently gained popularity [4].

All objects in traditional supervised learning have been defined with a particular trait in MLL when they are related to the collection of labels. For unspecified attributes, the technique keeps the appropriate label collections [5]. Previously, MLL was increasingly involved in particular fields of machine learning and was extensively utilized in various issues, like computerized efficiency with multimedia information [6]. Earlier MLL studies focused solely on multi-label text classification using a predefined set of labels [7]. But in many real-world situations, a dynamic case that might include more recent labels with perhaps the finest labels in the anticipated data stream structure was evaluated.

In a complicated world, a learning approach may reassign and transform an established framework into various arrangements. This approach needs to be capable of recreating an established framework in the MLL model for novel characteristics as well as updated categorization methods for every NL [8]. Excluding real-world learning information, there remains no Ground Truth (GT) for labels in the dynamic MLL configuration at any point in the data stream. As a result, the key challenges included discovering and simulating NLs [9]. Overall, it was quite hard to identify the characteristics of an NL. Because fresh labels were not truly present in earlier information and primarily co-existed with a few useful labels, it was extremely difficult to separate features from those with recognized labels from those with NLs. The rate of failure increased as the number of NLs in the data stream increased due to an incorrect categorization. As a result, developing appropriate frameworks for improving categorization performance in a data stream was a difficult challenge. To deal with this problem, numerous MLL algorithms with innovative ways of detecting relationships between labeled and unlabeled characteristics were presented [10].

Based on these considerations, Zhu et al. [11] proposed the MuENL to recognize and categorize characteristics with Emerging NLs (ENLs). This MuENL method is divided into several important stages: 1) categorizing the attributes associated with newly detected labels, 2) identifying the presence of an NL, and 3) developing a novel categorization system for all NLs that collaborate with the predictor for the recognized labels. Furthermore, MuENLHD was modified to deal with sparse high-dimensional data streams by decreasing size via a kernel Principal Component Analysis (PCA). In contrast, this approach could only deal with a specific NL in a given stage. In contrast, when the test set contains several NLs in a single stage, this approach may treat the number of NLs as an independent NL. This leads to a decrease in performance.

As a result, the MuEMNL and MuEMNLHD approaches [12] were developed to address the issues in a dynamic situation with a huge quantity of NLs. The NL set was separated into numerous newer CLs individually for adjusting the dynamic situation in this approach. This approach consists of four distinct stages: i) A linear classification system was built to optimize the pairwise label classification error on the set of labels, ii) a new

outlier identification is built using both primary and test data stream, iii) the MuEMNLforest and MuEMNLHD clusters were identified using the OPTICS technique, and iv) a classification refining technique is employed to integrate NLs to construct an efficient classification model. But it needs to be maintained up-to-date for huge amounts of data arriving at high speeds using limited resources since it tends to concept drift problems.

Therefore, in this article, an adaptive ensemble learning approach is proposed that deals with a huge amount of data streams and solves concept drift issues by constructing MuEMNL-Neural Network (NN) rather than a random forest classifier. It specifies the ensemble size, the amount of independent NNs applying a constructive-pruning technique, their hidden nodes, and their learning examples. Also, to solve the concept drifts, pairwise and non-pairwise diversity measures are analyzed while constructing ensemble NN for efficient training using the entire learning examples. Moreover, the accuracy and diversity of NNs are maintained simultaneously. Thus, the MLL is enhanced when dealing with huge amounts of data streams.

The remaining sections are prepared as follows: Section II studies the related works on MLL in various applications. Section III explains the MuEMNL-ENN and Section IV portrays its performance. Section V reviews the findings and gives upcoming improvements.

II. LITERATURE SURVEY

Different MLL algorithms have been developed by earlier researchers for different kinds of applications. Some of them are reviewed in this section.

For MLL with ENLs, Kongsorot et al. [13] created an Incremental Kernel Extreme Learning Machine (IKELM). A novel detector and a multi-label classifier were featured. To identify cases with NLs, the detector was adopted with user-defined threshold values. To assign a label to each occurrence, a new incremental multi-label classifier and its improved variant were used. However, it was unable to handle idea drift in MLL when using ENLs. To learn the linked characteristics from mechanical vibration data, Shen et al. [14] presented a deep MLL. This model for fault diagnosis was trained using unlabeled examples. However, because it did not simulate the label correlations, it was unable to handle the new fault categorization.

Xie & Huang [15] developed a Partial MLL with Noisy label Identification (PML-NI) method to concurrently reconstruct the GT data and recognize the noisy tags. First, two objectives were defined such as trace norm and -norm regularizer. Then, the multi-label classifier and noisy tag detector were together optimized via integrating the tag similarity and attribute-triggered noise models based on the noise-corrupted label matrix. Also, it was extended into multi-instance MLL to recognize noisy labels according to ambiguous instances. But its complexity was high and the concept drift problem was not considered.

Tan et al. [16] designed a new Probabilistic Label Enhancement Algorithm (PLEA) based on the maximum entropy-based tag allocation. At first, the supervised data in the tag manifold was used in the attribute manifold space formation. A strong linear regression was used to predict the tag allocations related to the mined decreased-size attributes. Moreover, the unknown true label distributions were predicted precisely. But the concept drift problem was not solved.

Zhang & Li [17] presented the MLL scheme named LF-LELC. The clustering was applied to the positive and negative data. The amount of clusters was assigned by the data kept in the label vectors. The clustering ensemble schemes were used to get robust clustering outputs using label correlations. After that, label-specific features were created for all labels, and the classification system was created via label correlations. However, the learning performance was influenced by the class imbalance problem.

Tan et al. [18] created a unique MLL model that learns instance space granularity and class labels. This model simultaneously trained classifiers and recovered label matrices. The instance and label manifolds were used to recreate and train the feature label mapping. The self-adaptive penalty variable traded off production losses for several labels. But the concept drift problem was not solved.

Rastogi & Mortaza [19] developed Imbalance Multi-label data learning with Label Specific Features (IMLSF) to resolve MLL's class imbalance problem by weighting positive and negative instances of a label. Additionally, the similarities among the set of possible labels were considered. But it did not solve the concept drift problem. Hao et al. [20] created MMFL, an MLL method with missing features and labels. Matrix decomposition restored missing attributes and tags. In matrix factorization, sparse tail tags were solved by adding a classifier. However, they did not consider the concept drift issue in MLL.

Zhao et al. [21] developed a multi-label weak-label training system to restore tag semantic area via combined tag similarity. Tag data reliability, attribute-tag dependency, and label correlations were employed to restore the semantic area and improve semantic views. Also, $l_{2,1}$-norm was used to solve the missing label space noise problem.

Liu et al. [22] created ELSMML, a tag similarity and multi-view training MLL. A tag similarity matrix defined high-order label associations. Multi-view training and dimensionality reduction found the high-level latent semantic tag and latent attribute data. An accelerated proximal gradient scheme was adopted to obtain the predictive classifier by optimizing the model parameters. But it did not deal with the concept drift issues in MLL.

From the literature, it is apparent that most researchers analyzed the challenges in MLL algorithms and solved them by developing new MLL algorithms for different applications. Though they utilize the label correlations for MLL, none of them deal with the concept drift issue in MLL, which impacts learning performance.

III. PROPOSED METHODOLOGY

This section describes the MuEMNL-ENN in detail. A general pipeline of this study is portrayed in Fig. 1.

First, the robust ensemble classifier is designed for MLL using both new and prior labels. As well, the Optics-based clustering is used to group many NLs exist concurrently in single iteration. Generally, the MLL problems exists in two stages: (1) one is to construct an identifier $\mathcal{D}_t$ to recognize NLs, and (2) the other is to get multiple NLs independently [12]. Such problems can be resolved by the MuENL/MuENLHD based on adaptive ensemble learning and Optics clustering. Once these procedures are performed, the classifier refinement is used stage-by-stage, and all stages are implemented according to earlier stage to design a robust ensemble classifier $\mathcal{H}_t = [h_{t,1}, \dots, h_{t,l}] \rightarrow \mathcal{H}_t' = [h_{t,1}, \dots, h_{t,l}, \mathcal{D}_t]$.

A. *New Label Identifier*

Consider a dataset with an attribute or label, which is modified version of one that is already exist. To identify NL and attributes, in [12], the MuENLForest classifier has been used that comprises g MuENLTree. But, the forest classifier did not handle the concept drift issue while data streams arriving concurrently at high speed. The term concept defines the

complete distribution of a data in a specified period. Being defined by the joint distribution $P(x, y)$, where x is the sample input and y is its label. So, the concept from which the data stream is created shifts after a particular time which provides the phenomena called concept drift. In other words, for all periods t, $P_t(x, y) \neq P_{t+\Delta}(x, y)$.

Learning data that modify in distribution over period makes classifying them in the training set no longer an easy process, because of the fact that the data is no longer consistent with current concepts. The major objective of this study is to adopt MuENL-ENNs that comprises n NNs rather than the g forest classifier with a proper tradeoff between accuracy and diversity of ensemble classifiers.

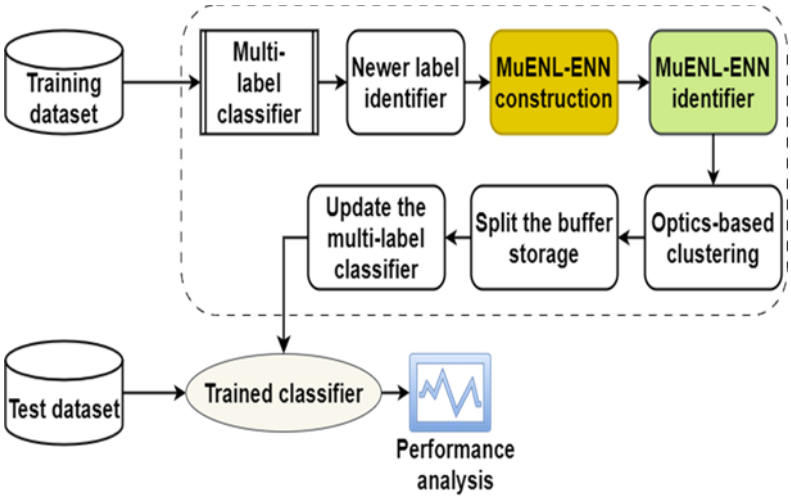


Figure 1. General Pipeline of the Proposed Study

1) Adaptive ensemble learning

The adaptive ensemble learning can automatically determine the total of base learner NNs and their structures in an ensemble in the learning stage. Fig. 2 depicts a design of an ensemble NN. Every NN in the ensemble is initially trained by the learning examples. The result of the ensembles is computed from the anticipated results of each NN.

The steps in this adaptive ensemble learning are given below.

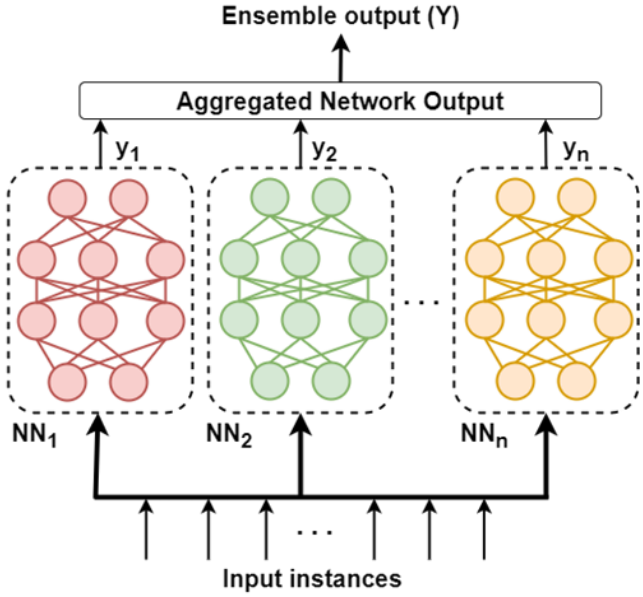


Figure 2. Design of Ensemble NN Classifier $(\mathcal{H})$

- S1: Generate an ensemble with tiniest structure containing 2 NNs, which have an input level, 2 hidden levels and an output level. The total neurons in the input and output levels are computed by the user. Then, a constructive strategy is applied depending on Ash's adaptive node construction scheme for the initial NN learning (later on the odd NNs in the ensemble). First, this NN establishes with a tiny structure comprising single node in all hidden layers. For the 2nd NN learning (later on even NNs in the ensemble), Reed's pruning scheme is applied. The hidden layer has too many neurons during NN pruning. Randomly initialize all NN connection weights.
- S2: Generate independent learning samples for all ensemble's NNs. Normally, sub-collection of learning samples for separate NNs are generated by arbitrarily choosing the primary collection of learning samples. In this study, learning collections are generated so when single NN trains using learning samples from the initial to the final, another NN trains using the final to the initial of similar learning samples.
- S3: Train the ensemble's NNs partly on data for a user-defined epoch counts by Negative Correlation Learning (NCL) [23].
- S4: Calculate the learning fault $\mathcal{E}_i$ for i^{th} NN in the ensemble based on below rule:

$$\mathcal{E}_i = 100 \frac{O_{max} - O_{min}}{N \times S} \sum_{n=1}^{N} \sum_{s=1}^{S} [(d(n,s) - F_i(n,s)^2) + \lambda P_i(n,s)] \quad (1)$$

In Eq. (1), O_{max} denotes the highest range and O_{min} denotes the least range of desired results, correspondingly, N defines sum amount of samples, S indicates the output neuron counts, $d(n,s)$ denotes the target result, and $F_i(n,s)$ represents the original result of neuron s in n^{th} learning sample. The principle in Eq. (1) is introduced by Reed and NCL for an NN fault. $\mathcal{E}_i$ describes separate dimension of the learning samples and the total output neurons.

- S5: Calculate the ensemble fault $\mathcal{E}$, where $\mathcal{E}$ denotes the mean of $\mathcal{E}_i$ of the base learner NNs. When $\mathcal{E}$ is lesser and tolerable, the ensemble structure is considered to have the maximum generalizability and provide an absolute ensemble. When $\mathcal{E}$ is not tolerable, either the ensemble structure or the separate base learner NNs experience modify.

- S6: Examine the neuron insertion or removal principle of separate NNs. In this principle, hidden neurons are inserted or removed when the fault of separate NNs doesn't modify after a predefined epoch selected by the system. When the principle is not encountered, the separate NNs are inefficient and the ensemble experiences adding novel learner NN.
- S7: Insert or remove hidden neurons to or from the NNs to encounter the insertion or removal principle, and remain learning by NCL.
- S8: Insert a novel NN to the ensemble when earlier NN insertion enhances the ensemble efficiency. Prepare and generate various learning collections for the novel NN, similar to S2. Return to S3 for advanced learning of the ensemble.

Therefore, the structure of ensemble is determined by executing Step 1 – Step 8. The principle of changing the learning samples enables the NNs to train various areas of the data distribution.

a) Nodes Insertion or Removal to/from Separate Neural Networks

During the learning stage of separate NNs, a few portions is exist that might essential or robust either for constructive or pruning schemes. When each NN in the ensemble learns either merely by constructive or pruning scheme, their training can be extremely identical. Although NCL guides the NNs to train using various data space, the training cannot be accurate when the NNs in the ensemble contain similar structure. Various structures of the NNs in the ensemble can give a distinct weight on the precision and diversity, which validates the use of hybridized constructive-pruning scheme in adaptive ensemble learning.

b) Neural Network Insertion to the Ensemble

In adaptive ensemble learning, constructive scheme is utilized to insert NNs in the ensemble. Novel NNs are inserted to the ensemble when earlier insertion enhances the ensemble efficiency. This insertion procedure is repeated hitherto the lowest ensemble fault is achieved.

c) Various Learning Collections for Separate NNs

Changing the samples into various learning collections facilitates effective training and could support the ensemble training from the entire learning samples. Learning collections are changed by keeping particular essential principle, i.e., learning collections must contain suitable amount of samples therefore separate NNs get the required data for training. In adaptive ensemble learning, when the initial NN in the ensemble trains by odd-situated learning samples, the 2nd NN trains by even-situated learning samples, and the 3rd NN trains by remaining learning samples in same manner. In a few scenarios, subsets of learning samples are generated by splitting or arbitrarily choosing. The algorithm for an adaptive ensemble learning is presented below.

Algorithm 1 Adaptive Ensemble Learning for Designing New Label Identifier

1. Initialize
2. Generate an ensemble having 2 NNs with tiniest structure of 1 input-2 hidden-1 output levels;
3. Determine the total neurons in input and output layers;
4. Use Ash's constructive scheme for adaptive node generation for the initial NN learning;
5. Use Reed's pruning scheme for the 2nd NN learning;
6. Generate separate training samples for all NNs;
7. Train NNs partly for predetermined epochs by NCL;
8. Calculate $\mathcal{E}_i$ for i^{th} NN as Eq. (1);
9. Calculate $\mathcal{E}$;
10. $\boldsymbol{if}(\mathcal{E} < tolerable)$
11. **Return** final ensemble
12. $\boldsymbol{end\ if}$
13. $\boldsymbol{if}\left(^{insertion\ or\ removal\ principle\ is\ not}_{\qquad\qquad reached}\right)$
14. Insert NN to ensemble;
15. Move to Step 6;
16. $\boldsymbol{else}$
17. Insert or remove hidden nodes to NN;
18. Move to Step 7;
19. $\boldsymbol{end\ if}$

2) Diversity in ensembling

Some fundamental measures of diversity that are utilized as a promising indication about how different ensemble classifiers are, for circumstances where data is static and its distribution does not modify in period. Thus, the diversity is modeled based on the pairwise non-pairwise diversity measures.

To determine all measures, consider a dataset of N samples, represented as x_i, all have a label y_i. The L base classifiers denoted by the set $H = \{h_1, h_2, \dots, h_L\}$ are trained with the training set $Tr = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$. All classifiers j have an output for x_i is $h_j(x_i)$ that relates to the weight w_j in the ensemble set of weights with a classification accuracy equal to p_j. The accurate or inaccurate decision is a $N * L$ matrix, known as oracle result O, whose components are represented in $\{-1,1\}$. In other words, $o_{ij} = 1$ when training sample x_i is classified accurately by the base classifier h_j, -1 or else. Oracle results are merely promising for a labeled dataset and they provide a common model to analyze a classifier ensemble.

Table 1. 2×2 Ensemble Element Correlation with Probabilities

	C_i accurate	C_i inaccurate
C_j accurate	a	b
C_j inaccurate	c	d

The mean classification precision of the base classifiers on the learning sample P is described by

$$P = \sum_{j=1}^{L} w_j p_j \tag{2}$$

$$P = 1 - \frac{\sum_{i=1}^{N} l_i}{NL} \tag{3}$$

In Eqns. (2) & (3), l_i is the product of L and sum of the weights of the base classifiers that categorize x_i inaccurately, denoted by $l_i = L \sum_{o_{ij}=-1} w_j$. To simplify the diversity measure computation, two base classifiers C_1 and C_2 are considered, which gives the outcomes in 2×2 matrix as illustrated in Table 1.

a) Pairwise Diversity Measures

Each pairwise measure is employed on a pair of base learners that can generate $\frac{L(L-1)}{2}$ pairwise diversity values. To obtain a unified value, it is essential to average across each part that contain the ensemble.

- Correlation coefficient: Correlation between two binary classifier outputs is computed for a pair of oracle outputs as the probabilities for the respective pair of

proper or improper outputs. Therefore, ρ is computed by

$$\rho_{1,2} = \frac{ad-bc}{\sqrt{(a+b)(d+c)(b+d)(a+c)}} \quad (4)$$

- Q-statistics: Yules Q-statistics for two classifiers C_1 and C_2 is calculated as:

$$Q_{1,2} = \frac{ad-bc}{ad+bc} \quad (5)$$

Eq. (5) signifies that Q and ρ have a similar sign and $|\rho| \leq |Q|$. In circumstances where the classifiers are statistically independent, the corresponding prior probabilities are equivalent to real probability, which tends to a value of $Q_{1,2} = 0$. In other scenarios, Q differs from -1 to 1, i.e., negative when classifiers commit errors on multiple objects and positive when they tend to detect a similar label properly.

Generally, L classifiers are ensemble to average the weighted vote and consider a group of pairs of classifiers to support the Q-statistics computation. Therefore, the mean Q-statistics over each pair of classifier is as:

$$Q_{mean} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} Q_{i,j} \quad (6)$$

- Disagreement measure: It is the probability that two distinct classifiers execute diversely on a similar training data. In other words, it is the proportion between the amount of observations on which certain classifier is accurate and another is inaccurate to the total observations. So, the diversity improves with the disagreement measure value. It is calculated as:

$$dis_{1,2} = b + c \quad (7)$$

For a group of L classifiers, this diversity measure is computed as the mean value over each pair of base classifiers.

$$dis_{mean} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} dis_{i,j} \quad (8)$$

- Double-Fault (DF) measure: The other option to measure the diversity within an ensemble is the DF value that is depending on the fact that it is more essential to identify when the concurrent errors are being committed than when both classifiers are accurate. In other words, it is the probability that the two classifiers C_1 and C_2 both serve incorrect during the classification task, i.e., the ratio of the cases that have been misclassified by both classifiers.

$$DF_{1,2} = d \quad (9)$$

For a group of L classifiers, the DF measure is computed by

$$DF_{mean} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} DF_{i,j} \quad (10)$$

b) Non-pairwise Diversity Measure

In addition to the pairwise diversity measures, the most general measures are also considered, where each classifier is considered as a complete responsibility; therefore, the output concerns directly only the ensemble and is associated with the oracle classifier outputs that are 1 for accurately labeled samples and -1 or else.

- Entropy measure: It is relied on the fact that the maximum diversity among a group of L classifiers for a certain x_j is manifested by $[L/2]$ of the votes with a similar value; either -1 or 1, and another $L - [L/2]$ alternate value. For example, when they all were -1s or 1s, there is no disagreement and the diversity is at its least value 0.

$$E = \frac{2}{L-1} \sum_{i=1}^{N} \min\left(\sum_{j=1}^{L} h_j(x_i), L - \sum_{j=1}^{L} h_j(x_i)\right) \quad (11)$$

For a C fraction of accurate outputs, the entropy is computed by

$$E = -a \log(a)(1-a) \log(1-a) \quad (12)$$

- Kohavi-Wolpert variance: The idea of bias-variance tradeoff is created novel decomposition principle of the classification error. Kohavi and Wolpert [24] provided an actual representation of the variability of the classified label y for x, across training sets, for a particular classifier model C_j:

$$Variance_x = \frac{1}{2}\left(1 - \sum_{i=1}^{c} P\left(y = {}^{w_i}/_{x}\right)^2\right) \quad (13)$$

Also, Kuncheva and Whitaker applied an improved version as:

$$KW = \frac{1}{NL^2} \sum_{i=1}^{N} l_i(L - l_i) \quad (14)$$

It is essential to observe that the diversity improves with values increasing of the KW variance.

- Inter-rater agreement: The inter-rater reliability (κ) is utilized to measure the level of agreement within a certain group of classifiers. So, the diversity improves if the classifiers disagree with the other, i.e., the value of κ decreases. It is computed by

$$\kappa = 1 - \frac{\sum_{i=1}^{N} l_i(L-l_i)}{NL(L-1)P(1-P)} \quad (15)$$

- Generalized Diversity (GD) measure: Partidge and Krzanowski [25] adopted the GD. They claimed that the maximum diversity is accomplished if the failure of one classifier is accompanied by accurate classification by the other classifier and minimum diversity exists if two classifiers fail together. For a random training sample x_i,

$$GD = 1 - \frac{\sum_{1}^{L} \frac{j(j-1)}{L(L-1)} T_j}{\sum_{1}^{L} T_j} \quad (16)$$

In Eq. (16), T_j is the probability that $l_i = j$, i.e., the probability that accurately j out of the L classifiers fail on a randomly selected input.

Thus, the primary advantages of the adaptive ensemble learning are (i) it achieves an automated ensemble, (ii) sustaining the accuracy and diversity of NNs simultaneously, and (iii) it needs least amount of variables set by the system.

Algorithm 2 presents an overall pseudocode of the MuEMNL-ENN, which consists of a multi-label ensemble

classifier $\mathcal{H}_0$, an outlier identifier $\mathcal{D}_t$, OPTICS clustering to divide NLs within the buffer memory into n NLs, and modifying the ensemble classifier $\mathcal{H}_t \rightarrow \mathcal{H}_{t+n}$.

Algorithm 2 MuEMNL-ENN for Low-Dimensional Dataset

Input: Initial learning set R_0, initial label set Y_0, and earlier known labels c_0

Output: Function set $\mathcal{H}_t$ for every instances that has a NL at time t (r_t)

1. Begin
2. Perform *Algorithm 1* to construct an ensemble classifier $\mathcal{H}$;
3. Obtain a primary $\mathcal{H}_0$ by learning R_0, Y_0;
4. Build an initial NL identifier $\mathcal{D}_0$ depending on R_0;
5. Initialize sampling eight vector $s_0 = 1_{|r_0|}$;
6. $\mathcal{H}_1 = [\mathcal{H}_0, \mathcal{D}_0]$; $\mathcal{D}_1 = \mathcal{D}_0$;
7. Repeat
8. Get a new instance r_t, $R_t = [R_{t-1}; r_t^{\mathrm{T}}]$;
9. Expand the sampling weight vector $s_t = [s_{t-1}; 1]$ concurrently;
10. $\boldsymbol{if}(\mathcal{D}_t(r_t) \geq 1)$
11. Include r_t to buffer;
12. $\boldsymbol{if}\big(|Buffer| \geq maximum\ buffer\ storage\ (BS_{Max})\big)$
13. Use Optics clustering to split buffer storage into n clusters for n new labels;
14. $\boldsymbol{while}(i > n)$
15. Create $\mathcal{D}_{t+i}$ and $\mathcal{H}_{t+i}$ from $i = 0$ and every $\mathcal{D}_{t+i}$ depends on $\mathcal{D}_{t+i-1}$ iteratively;
16. ***end while***
17. Empty buffer;
18. $l \leftarrow l + n$; $c_t = c_{t-1} \cup \{l\}$;
19. Update $s_t \leftarrow 0.8 s_t$;
20. ***end if***
21. ***end if***
22. $c_t = c_{t-n}$; $\mathcal{D}_t = \mathcal{D}_{t-n}$; $\mathcal{H}_t = \mathcal{H}_{t-n}$;
23. Until
24. Get $\mathcal{H}_t$;
25. End

Similar to *Algorithm 2*, the MuEMNL-ENN is applied for high-dimensional datasets by creating a random feature map [12] before constructing the ensemble classifier.

IV. Result and Discussion

The MuEMNL-ENN approach is analyzed by executing it in MATLAB 2019b using 5 multi-label standard datasets [26] including birds, CAL500, emotions, Enron and yeast, as well as 20Newsgroup dataset. The information of such datasets are given in [12]. The metrics considered for performance analysis are mean precision, F1-score and micro-F1. For a test collection (r_n, Y_n), $h(r_n)$ is the set of predicted labels for n^{th} sample, $f(r_n, y)$ denotes the certainty that r_n fits to the label y.

- **Mean precision:** It is the average proportion of positive labels organized higher than the particular positive label.

$$Mean\ precision = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i|}\sum_{y\in Y}\frac{|l_p|}{sort_{f(r_n,y)}} \quad (17)$$

Where,

$$l_p = \left\{y' | sort_{f(r_n,y')} \leq sort_{f(r_n,y)}, y' \in Y_i\right\} \quad (18)$$

Here, Y_i indicates the collection of positive labels, n denotes the total test data, l_p denotes the collection of anticipated positive labels that are organized less than label y for r_n.

- **F1-score:** It is the harmonic mean of precision and recall for all samples.

$$F1 - score = \frac{1}{n}\sum_{i=1}^{n}\frac{2\times Precision_i \times Recall_i}{Precision_i + Recall_i} \quad (19)$$

- **Micro-F1:** It is calculated by

$$Micro\ F1 = \sum_{i=1}^{n}\frac{2\times Precision_i \times Recall_i}{Precision_i + Recall_i} \quad (20)$$

- **Accuracy:** It determines the efficiency of MuEMNL-ENN to properly estimate label of new data.

$$Accuracy = \frac{True\ Positive\ (TP) + True\ Negative\ (TN)}{TP + TN + False\ Positive\ (FP) + False\ Negative\ (FN)} \quad (21)$$

In Eq. (21), TP is the count of positive instances estimated as themselves, TN is the count of negative instances estimated as themselves, FP is the count of positive instances estimated as negative, and FN is the count of negative instances estimated as positive.

- **Hamming loss:** It is the fraction of data that had one of their labels incorrectly estimated or left out altogether.
- **One-error:** It is the fraction of data whose top-ordered estimated label is not in the GT label collection.
- **Coverage:** It quantifies the average number of steps needed to move down an instance's ranked label collection to cover every related labels.
- **Ranking loss:** It is the average proportion of misranked label sets, i.e. an inappropriate label of a data is arranged above its appropriate label.

A. *Comparing MuEMNL-ENN effectiveness on low-dimensional datasets*

The performance of MuEMNL-ENN is evaluated with the IKELM [13], LF-LELC [17], MuENL [11] and MuEMNLForest [12] on 5 distinct low-dimensional datasets.

Fig. 3 compares the mean precision between proposed and existing MLL algorithms. It is shown that the MuEMNL-ENN algorithm reaches a higher mean precision than other MLL algorithms, e.g., when the MuEMNL-ENN algorithm is tested on the Enron dataset, the mean precision is increased by 40.12%, 22.28%, 14.06%, and 7.35% compared to the IKELM, LF-LELC, MuENL, and MuEMNLForest algorithms.

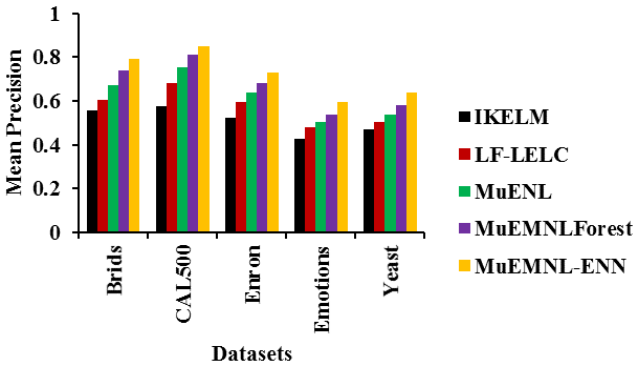


Figure 3. Mean Precision of Different MLL Algorithms on Low-dimensional Datasets

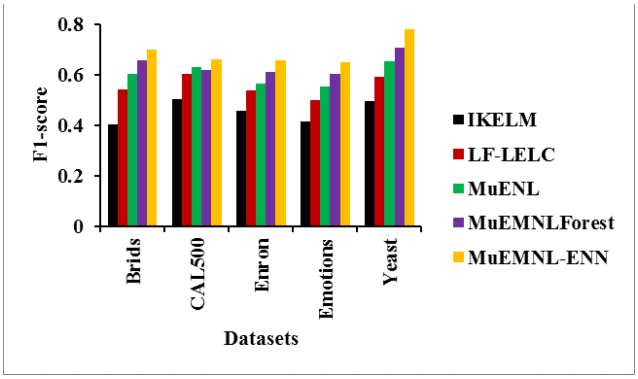


Figure 4. F1-score of Different MLL Algorithms on Low-dimensional Datasets

Fig. 4 plots the F1-score values of the proposed and existing MLL algorithms. It is noticed that the MuEMNL-ENN algorithm accomplishes a higher F1-score than other MLL algorithms, e.g., when the MuEMNL-ENN algorithm is tested on Birds dataset, the F1-score is increased by 74.38%, 29.57%, 15.87%, and 6.37% compared to the IKELM, LF-LELC, MuENL, and MuEMNLForest algorithms.

Fig. 5 illustrates the **Δ**Micro F1 values of the proposed and existing MLL algorithms. It is noted that the MuEMNL-ENN algorithm increases the **Δ**Micro F1 than other MLL algorithms, e.g., when the MuEMNL-ENN algorithm is tested on the Emotions dataset, the **Δ**Micro F1 value is increased from 0.21 to 0.5 compared to the other MLL algorithms.

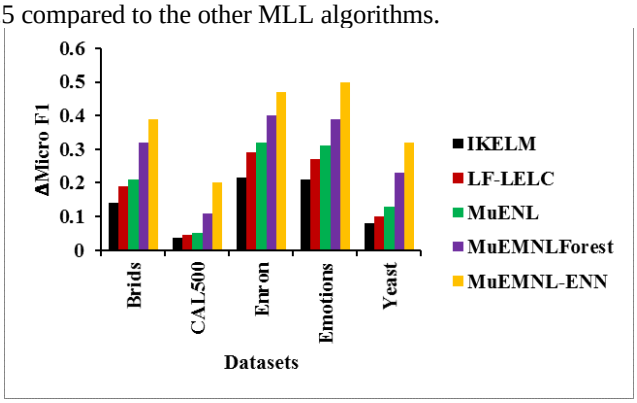
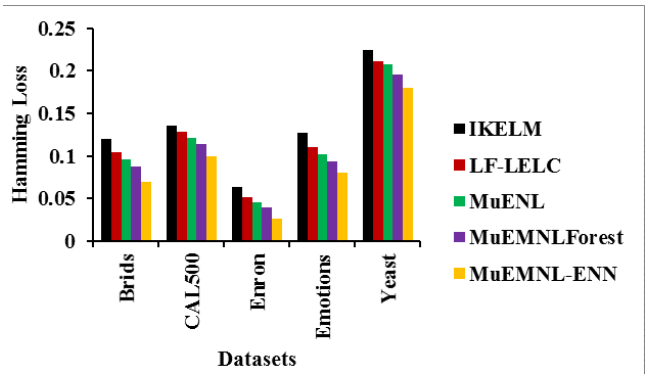
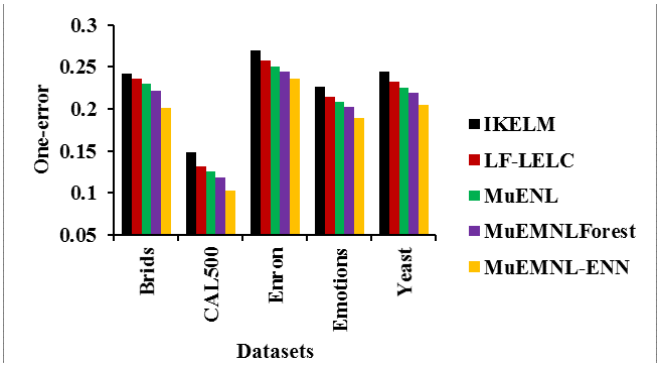


Figure 5. **Δ**Micro F1 of Different MLL Algorithms on Low-dimensional Datasets



(a)



(b)

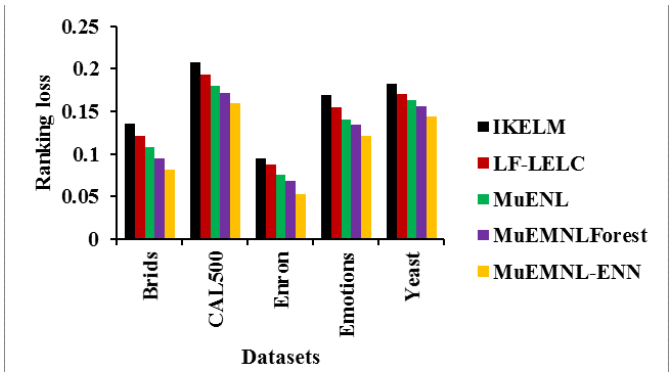
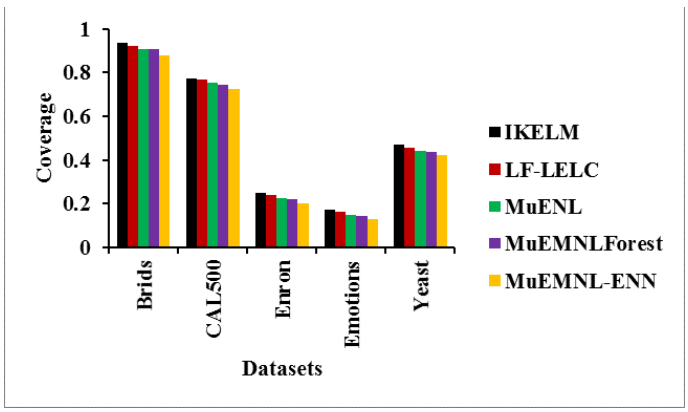
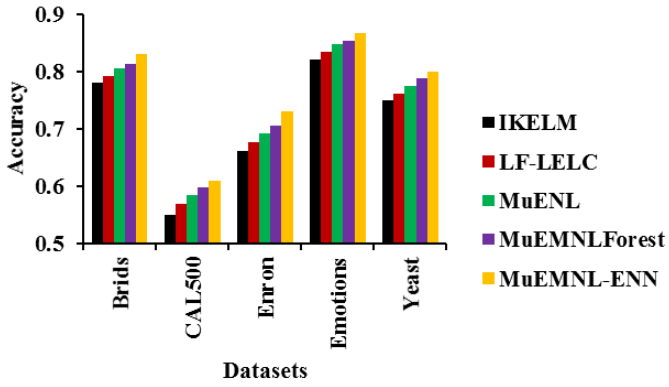


Figure 6. Comparison of Different MLL Algorithms on Low-dimensional Datasets (a) Hamming Loss, (b) One-Error, and (c) Ranking Loss



(a)



(b)

Figure 7. (a) Coverage and (b) Accuracy Comparison between Proposed and Existing MLL Algorithms on Low-dimensional Datasets

Fig. 6(a)-6(c) demonstrates the hamming loss, One-error, and ranking loss, respectively for various MLL algorithms. In

the case of CAL500 dataset, the hamming loss of MuEMNL-ENN is reduced by 26.47%, 21.88%, 17.36%, and 12.28% in contrast with the IKELM, LF-LELC, MuENL, and MuEMNLForest algorithms, respectively. The one-error of MuEMNL-ENN algorithm is decreased by 30.41%, 21.37%, 17.6%, and 12.71% compared to the IKELM, LF-LELC, MuENL, and MuEMNLForest, respectively. The ranking loss of MuEMNL-ENN is reduced by 23.08%, 17.1%, 11.11%, and 6.98% contrasted with the IKELM, LF-LELC, MuENL, and MuEMNLForest algorithms, respectively.

Fig. 7(a) & 7(b) portray the coverage and accuracy, respectively for proposed and existing MLL algorithms. In the case of yeast dataset, the coverage of MuEMNL-ENN is decreased by 10.64%, 8.3%, 5.19%, and 3.45% compared to the IKELM, LF-LELC, MuENL and MuEMNLForest algorithms, respectively. The accuracy of MuEMNL-ENN is increased by 6.81%, 4.99%, 3.36%, and 1.52% compared to the IKELM, LF-LELC, MuENL and MuEMNLForest, respectively. Thus, it is realized that the MuEMNL-ENN can perform better than the other MLL algorithms on low-dimensional datasets regarding various metrics.

B. Comparing MuEMNL-ENN effectiveness on high-dimentional dataset

MuEMNLHD-ENN is tested on the 20Newsgroup dataset, and its efficacy is compared to that of PML-NI [15], PLEA [18], and MuEMNLHDForest [12].

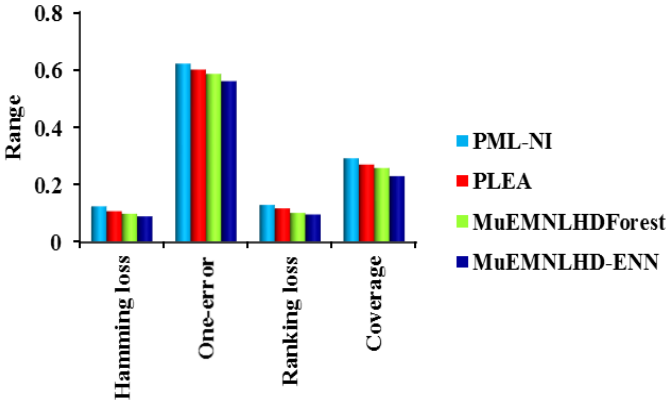


Figure 8. Comparison of Hamming Loss, One-Error, Ranking Loss, and Coverage between Proposed and Existing MLL Algorithms on 20Newsgroup Dataset

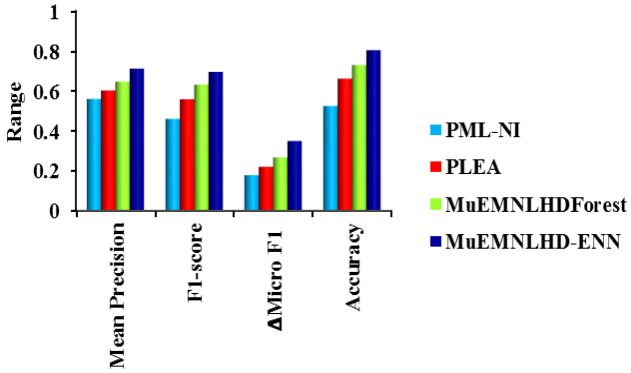


Figure 9. Comparison of Precision, F1-score, and Accuracy between Proposed and Existing MLL Algorithms on 20Newsgroup Dataset

Fig. 8 depicts the mean precision, F1-score, **Δ**Micro F1, and accuracy for proposed and existing MLL algorithms utilizing the 20Newsgroup dataset. It is observed that the mean precision of MuEMNLHD-ENN is increased by 26.96%, 18.11% and 10.06% compared to the PML-NI, PLEA and MuEMNLHDForest algorithms, respectively. The F1-score of MuEMNLHD-ENN is increased by 51.09%, 24.55% and 10.14% compared to the PML-NI, PLEA and MuEMNLHDForest algorithms, respectively. The **Δ**Micro F1 of MuEMNLHD-ENN is increased by 96.07%, 58.64% and 30.71% compared to the PML-NI, PLEA and MuEMNLHDForest algorithms, respectively. The accuracy of MuEMNLHD-ENN algorithm is increased by 53.24%, 21.48% and 10.15% compared to the PML-NI, PLEA and MuEMNLHDForest, respectively.

Fig. 9 illustrates the hamming loss, one-error, ranking loss, and coverage for proposed and existing MLL algorithms. It is noticed that the hamming loss of MuEMNLHD-ENN is decreased by 27.92%, 16.57%, and 9.36% compared to the PML-NI, PLEA, and MuEMNLHDForest, respectively. The one error of MuEMNLHD-ENN is reduced by 9.82%, 6.67%, and 4.27% compared to the PML-NI, PLEA, and MuEMNLHDForest, respectively. The ranking loss of MuEMNLHD-ENN decreased by 25.62%, 18.05%, and 5.2% compared to the PML-NI, PLEA, and MuEMNLHDForest, respectively. The coverage of MuEMNLHD-ENN decreased by 21.23%, 14.81%, and 10.85% compared to the PML-NI, PLEA, and MuEMNLHDForest, respectively. Thus, it is realized that the MuEMNLHD-ENN performs better than existing MLL algorithms on a high-dimensional dataset.

V. CONCLUSION

In this study, an adaptive ensemble learning approach was developed that builds the MuEMNL-ENN for solving concept drift problems in MLL. The number of separate NNs was determined by the constructive technique, and the total hidden nodes of separate NNs was determined by the constructive-pruning strategy. As well, pairwise and non-pairwise diversity measures were examined to maintain the tradeoff between accuracy and diversity of ensemble NNs, which supports efficient learning of whole data with NLs. The test results show that on average, the MuEMNL-ENN on low-dimensional datasets reaches 72.18% mean precision, 76.76% accuracy, 69.08% F1-score, 0.112 ranking loss, 0.4706 coverage, 0.1868 one-error, and 0.0912 hamming loss contrasted with the existing MLL algorithms. Also, the MuEMNLHD-ENN on the high-dimensional dataset reaches 71.1% mean precision, 69.5% F1-score, 80.3% accuracy, 0.0967 ranking loss, 0.23 coverage, 0.0901 hamming loss, and 0.56 one-error contrasted with the existing MLL algorithms.

VI. REFERENCES

[1] T. Chen, L. Lin, R. Chen, X. Hui and H. Wu, "Knowledge-guided multi-label few-shot learning for general image recognition", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 3, pp. 1371-1384, 2020, doi:10.1109/TPAMI.2020.3025814.

[2] A. Siraj, S. Taneja, Y. Zhu, H. Jiang, S. Luthra and A. Kumar, "Hey, did you see that label? It's sustainable!: Understanding the role of sustainable labelling in shaping sustainable purchase behaviour for sustainable development", Business Strategy and the Environment, 31(7), 2820-2838, 2022, doi:10.1002/bse.3049

[3] B. Tiple and M. Patwardhan, "Multi-label emotion recognition from Indian classical music using gradient descent SNN model", Multimedia Tools and Applications, 81(6), 8853-8870, 2022, doi:10.1007/s11042-022-11975-4

[4] J. S. Suri, M. Bhagawati, S. Paul, A. D. Protogerou, P. P. Sfikakis, G. D. Kitas and L. Saba, "A powerful paradigm for cardiovascular risk stratification using multiclass, multi-label, and ensemble-based machine learning

paradigms: A narrative review", Diagnostics, vol. 12, no. 3, pp. 722, 2022, https://www.mdpi.com/2075-4418/12/3/722#

[5] M. K. Xie, J. Xiao and S. J. Huang, "CCMN: A General Framework for Learning With Class-Conditional Multi-Label Noise", IEEE Transactions on Pattern Analysis and Medical Intelligence. Vol. 45, no. 1, pp. 154-156, 2023, http://doi.org/10.1109/tpami.2022.3141240

[6] J. Bogatinovski, L. Todorovski, S. Džeroski and D. Kocev, "Comprehensive comparative study of multi-label classification methods", Expert Systems with Applications, vol. 203, pp. 117215, 2022, doi:10.1016/j.eswa.2022.117215

[7] L. Maltoudoglou, A. Paisios, L. Lenc, J. Martínek, P. Král and H. Papadopoulos, "Well-calibrated confidence measures for multi-label text classification with a large number of labels", Pattern Recognition, vol. 122, pp. 108271, 2022. doi:10.1016/j.patcog.2021.108271

[8] M. Han, H. Wu, Z. Chen, M. Li and X. Zhang, "A survey of multi-label classification based on supervised and semi-supervised learning", International Journal of Machine Learning and Cybernetics, vol. 14, no. 3, pp. 697-724, 2023, doi:10.1007/s13042-022-01658-9

[9] H. M. Gomes, M. Grzenda, R. Mello, J. Read, M. H. Le Nguyen and A. Bifet, "A survey on semi-supervised learning for delayed partially labelled data streams", ACM Computing Surveys, vol. 55, no. 4, pp. 1-42, 2022, doi:10.1145/3523055

[10] D. Li and C. Wang, "In-Depth Research and Analysis of Multilabel Learning Algorithm", Journal of Sensors, 2022, doi:10.1155/2022/7737166

[11] Y. Zhu, K. M. Ting and Z. H. Zhou, "Multi-label learning with emerging new labels", IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 10, pp. 1901-1914, 2018, doi:10.1109/TKDE.2018.2810872

[12] K. Y. S. S. Kanagaraj and M. Nallappan, "Methods for Predicting the Rise of the New Labels from a High-Dimensional Data Stream", International Journal of Intelligent Engineering & Systems, vol. 16, no. 1, 2023, http:// doi.org/10.22266/ijies2023.0228.30

[13] Y. Kongsorot, P. Horata and P. Musikawan, "An incremental kernel extreme learning machine for multi-label learning with emerging new labels", IEEE Access, vol. 8, pp. 46055-46070, 2020, doi:10.1109/ACCESS.2020.2978648

[14] J. Shen, S. Li, F. Jia, H. Zuo and J. Ma, "A deep multi-label learning framework for the intelligent fault diagnosis of machines", IEEE Access, vol. 8, pp. 113557-113566, 2020, doi:10.1109/ACCESS.2020.3002826

[15] M. K. Xie and S. J. Huang, "Partial multi-label learning with noisy label identification", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 7, pp. 3676-3687, 2021, doi:10.1109/TPAMI.2021.3059290

[16] C. Tan, S. Chen, G. Ji and X. Geng, "A novel probabilistic label enhancement algorithm for multi-label distribution learning", IEEE Transactions on Knowledge and Data Engineering, vol. 34, no. 11, pp. 5098-5113, 2021, doi:10.1109/TKDE.2021.3054465

[17] C. Zhang and Z. Li, "Multi-label learning with label-specific features via weighting and label entropy guided clustering ensemble", Neurocomputing, vol. 419, pp. 59-69, 2021, doi:10.1016/j.neucom.2020.07.107

[18] A. Tan, X. Ji, J. Liang, Y. Tao, W. Z. Wu and W. Pedrycz, "Weak multi-label learning with missing labels via instance granular discrimination", Information Sciences, vol. 594, pp. 200-216, 2022, doi:10.1016/j.ins.2022.02.011

[19] R. Rastogi and S. Mortaza, "Imbalance multi-label data learning with label specific features", Neurocomputing, vol. 513, pp. 395-408, 2022, doi:10.1016/j.neucom.2022.09.085

[20] X. Hao, J. Huang, F. Qin and X. Zheng, "Multi-label learning with missing features and labels and its application to text categorization", Intelligent Systems with Applications, vol. 14, pp. 200086, 2022, doi:10.1016/j.iswa.2022.200086

[21] D. Zhao, H. Li, Y. Lu, D. Sun, D. Zhu and Q. Gao, "Multi-label weak-label learning via semantic reconstruction and label correlations", Information Sciences, vol. 623, pp. 379-401, 2023, doi:10.1016/j.ins.2022.12.047

[22] B. Liu, W. Li, Y. Xiao, X. Chen, L. Liu, C. Liu and P. Sun, "Multi-view multi-label learning with high-order label correlation", Information Sciences, vol. 624, pp. 165-184, 2023, doi:10.1016/j.ins.2022.12.072

[23] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble", IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 29, no. 6, pp. 716-725, 1999, doi:10.1109/3477.809027

[24] J. H. Friedman, "On Bias, Variance, 0/1—Loss, and the Curse-of-Dimensionality", Data Mining and Knowledge Discovery, vol. 1, pp. 55–77, 1997, doi:10.1023/A:1009778005914

[25] D. Partridge and W. Krzanowski, "Software diversity: practical statistics for its measurement and exploitation", Information and software technology, vol. 39, no. 10, pp. 707-717, 1997, doi:10.1016/S0950-5849(97)00023-2

[26] Multi-label Classification Dataset Repository, Knowledge Discovery and Intelligent Systems KDIS University of Crdoba. [Online]. Available: https://www.uco.es/kdis/mllresources/. [Accessed: 06-March-2020].

