Neural Comput & Applic (2018) 29:245–256
https://doi.org/10.1007/s00521-016-2446-x

ORIGINAL ARTICLE

A comparative study of fuzzy PSO and fuzzy SVD-based RBF neural network for multi-label classification

Shikha Agrawal[1] · Jitendra Agrawal[1] · Shilpy Kaur[1] · Sanjeev Sharma[1]

Received: 10 March 2015 / Accepted: 1 July 2016 / Published online: 19 July 2016
© The Natural Computing Applications Forum 2016

Abstract In multi-label classification problems, every instance is associated with multiple labels at the same time. Binary classification, multi-class classification and ordinal regression problems can be seen as unique cases of multi-label classification where each instance is assigned only one label. Text classification is the main application area of multi-label classification techniques. However, relevant works are found in areas like bioinformatics, medical diagnosis, scene classification and music categorization. There are two approaches to do multi-label classification: The first is an algorithm-independent approach or problem transformation in which multi-label problem is dealt by transforming the original problem into a set of single-label problems, and the second approach is algorithm adaptation, where specific algorithms have been proposed to solve multi-label classification problem. Through our work, we not only investigate various research works that have been conducted under algorithm adaptation for multi-label classification but also perform comparative study of two proposed algorithms. The first proposed algorithm is named as fuzzy PSO-based ML-RBF, which is the hybridization of fuzzy PSO and ML-RBF. The second proposed algorithm is named as FSVD-MLRBF that hybridizes fuzzy c-means clustering along with singular value decomposition. Both the proposed algorithms are applied to real-world datasets, i.e., yeast and scene dataset. The experimental results show that both the proposed algorithms meet or beat ML-RBF and ML-KNN when applied on the test datasets.

Keywords Multi-label classification · Fuzzy PSO · Fuzzy SVD · RBF neural network

✉ Jitendra Agrawal
jitendra@rgtu.net

Shikha Agrawal
shikha@rgtu.net

Shilpy Kaur
shilpykaur191@gmail.com

Sanjeev Sharma
sanjev@rgtu.net

[1] Department of Computer Science and Engineering, Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal, M.P., India

1 Introduction

Classification is a process in which each record in the dataset is assigned to a particular class from the set of classes. Classification can be single label or multi-label.

In single-label classification, learning is performed for a set of examples that are associated with single-label c from a set of disjoint labels L. For example, an instance will belong either to class "C1" or class "C2" or class "C3" from the set of class {C1, C2, C3}. However, several modern applications such as text categorization, medical analysis, protein function categorization, music classification and semantic scene categorization require examples to be associated with a set of labels. For example, a text document consisting information about the attacks of 26/11 can be categorized as news, movie and terrorist attack. Similarly in medical diagnosis, a patient may be suffering from cancer, diabetes and kidney failure at the same time. In semantic scene classification, a snap can belong to more than one conceptual class, such as beach, forest, city and people at the same time. Similarly, a protein can perform many functions simultaneously. For example, enzymatic proteins increase metabolism for digestion in the stomach, functioning of pancreas, blood clotting and convert glycogen into glucose. Thus, in multi-label classification, each example is associated with a subset of labels Yi in the

given label set C, i.e., Yi $\subseteq$ C. Such classification problems can be solved either by problem transformation or by algorithm adaptation approach.

The structure of this paper is organized as: Sect. 2 covers the problem transformation methods, and Sect. 3 briefly describes the work done by various researchers to enhance the algorithm adaptation methods. Section 4 gives detailed description of both the proposed algorithms, while Sect. 5 reports experiments and results. Finally, Sect. 6 concludes the paper.

2 Problem transformation

Problem transformation approach generally transforms the multi-label classification problem into several single-label problems and then classifies it. The multi-label dataset can be transformed into single-label dataset using various simple transformation techniques [1] such as copy, copy weight, select-max, select-min, select random and ignore.

Label powerset, pruned problem transformation, random *k*-label set, binary relevance, ranking pairwise comparison and calibrated label ranking are some problem transformation methods. Among them, label powerset (LP) is simplest in which new classes for single-label classification are formed by combining each set of labels in the multi-label dataset. This leads to creation of many class labels, and some of them may occur rarely. Tables 1 and 2 show an illustration of transformed multi-label dataset using LP.

To overcome this drawback, pruned problem transformation (PPT) method is proposed in [2]. In PPT, all the label sets that occur less than predefined threshold are pruned.

In random k-label set (RAkEL) [3], small subsets of label sets are obtained by randomly breaking large label sets. For each of these subsets, LP classifier is trained.

In binary relevance (BR), the multi-label dataset is divided into d-datasets one for each label that is present in multi-label dataset and then, each single-label dataset so formed can be classified using traditional classification algorithms. Tables 3, 4, 5 and 6 illustrate the transformed dataset using BR method.

The ranking by pairwise comparison (RPC) [4] works in two phases: In the first phase, for each pair of label a binary dataset is obtained by transforming the multi-label dataset. The binary classifier is then trained for each of binary datasets. In the second phase, ranking is obtained by counting the votes received by each label.

Calibrated label ranking (CLR) [5] extends RPC by providing additional information about ranking of labels. It adds an additional calibrated label to deal with multi-label ranking. The additional label acts as a split point between relevant and irrelevant sets of labels.

3 Algorithm adaptation

The algorithm adaptation approach uses various algorithms to directly handle the entire multi-level dataset. There exist several algorithms which have been categorized into two types: one that considers multi-label dataset as a whole and operate on the entire dataset and labels simultaneously,

Table 1 Example of multi-label dataset

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	C_1, C_2, C_3
B	X_1, X_2, X_3, X_4	C_1, C_4
C	X_1, X_2, X_3, X_4	C_2, C_3, C_4

Table 2 Transformed dataset for label power set

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	$C_{1,2,3}$
B	X_1, X_2, X_3, X_4	$C_{1,4}$
C	X_1, X_2, X_3, X_4	$C_{2,3,4}$

Table 3 Transformed dataset using binary relevance for label C_1

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	C_1
B	X_1, X_2, X_3, X_4	C_1
C	X_1, X_2, X_3, X_4	$\neg C_1$

Table 4 Transformed dataset using binary relevance for label C_2

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	C_2
B	X_1, X_2, X_3, X_4	$\neg C_2$
C	X_1, X_2, X_3, X_4	C_2

Table 5 Transformed dataset using binary relevance for label C_3

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	C_3
B	X_1, X_2, X_3, X_4	$\neg C_3$
C	X_1, X_2, X_3, X_4	C_3

Table 6 Transformed dataset using binary relevance for label C_4

Objects	Attributes	Label set
A	X_1, X_2, X_3, X_4	$\neg C_4$
B	X_1, X_2, X_3, X_4	C_4
C	X_1, X_2, X_3, X_4	C_4

while another set of algorithms first transforms the multi-label dataset into single-label and then operate on the transformed dataset. This section gives the brief review of research work done under algorithm adaptation.

3.1 Boosting algorithms

The main idea behind boosting is to combine many weak classifiers to produce a strong classifier. This is done by iteratively selecting a training set where each instance is assigned a label. Set of weights are uniformly distributed "Dt" over the instances and labels. These weights are then fed to the weak learner which produces weak hypotheses. Error is computed using summation of distribution "Dt." Finally, based on this error value, the weights of incorrectly classified instances are increased so that the examples that were classified incorrectly are fed back to the algorithm and the weak learner is forced to focus on the hard examples in the training set, whereas correctly classified examples are removed. Based on this concept, simplest version of Adaboost (adaptive boosting) is proposed. However, maintaining a set of weights over training examples does not solve the problem of multi-class and multi-label. So to deal with such problems, set of weights are maintained over training examples and labels. During boosting process, the training examples and their corresponding labels get incrementally higher weights that are difficult to predict, while lower weights are maintained over the examples and labels that are easy to classify. Schapire and Singer [6] proposed two extensions of Adaboost algorithms for multi-class, multi-label classification problems. The first boosting algorithm, named as Adaboost.MH, is derived by reducing the multi-label data to binary data. At next step, binary Adaboost is applied on these binary data. The goal is to predict only correct labels. It uses hamming loss and updated learning algorithms to increase the accuracy of classification task. Adaboost.MR is the second algorithm that performs label ranking such that the correct labels receive the highest ranks. Classification probabilistic accuracy is improved by ranking loss.

3.2 Generative models

Probabilistic generative models are generally used to generate a sequence of observable data using some probability distribution. The naive Bayes model is a conditional probability learning method that uses Bayes theorem. Bayes theorem relates the probability of the occurrence of an event to the occurrence or non-occurrence of an associated event, i.e., the probability that an event A occurs given that another event B has already occurred is equal to the probability that the event B occurs given that A has already occurred multiplied by the probability of occurrence of event A and divided by the probability of occurrence of event B. This conditional probability relationship between two events A and B is given by:

$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{B}|\mathrm{A})P(\mathrm{A})}{P(\mathrm{B}|\mathrm{A})P(\mathrm{A}) + P(\mathrm{B}|\bar{\mathrm{A}})P(\bar{\mathrm{A}})} \quad (1)$$

Due to the fact that naive Bayes is fast and highly scalable, it is being used for multi-label classification

Ueda and Saito [7] applied probabilistic generative model for multi-label text categorization problem. To detect multiple categories of text simultaneously, two probability generative models, namely PMM1 (parametric mixture model 1) and PMM2 (parametric mixture model 2), are proposed in their work. These proposed models use word-based representation, Bag-Of-Words (BOW) representation. It is based on the assumption that "mixture of characteristic words which appear in single-labeled text belongs to each category of multi-categories" [7]. In PMM1, approximation of a class-dependent probability is done. This is regarded as "first-order" approximation. But according to author, PMM2 is a more flexible model. This is because the parameter vectors of duplicate category are also used to approximate class-dependent probability. When experimented on real-world dataset, i.e., yahoo.com Web pages, PMMs proved to be much faster than naive Bayes, k-nearest neighbor and three-layer neural networks.

Nadia Ghamrawi et al. [8] proposed two models based on conditional random field (CRF). The first model is collective multi-label classifier (CML) that captures the label co-occurrences but do not account for the presence of particular feature values in objects. The second model known as collective multi-label with features classifier (CMLF) maintains parameters for observational features. This can be understood by simple example, say a text document can be categorized in one of the three classes {ORANGE, MANGO, RICE}. The presence of a word fruits in the document increases the likelihood of being correctly classified to ORANGE and MANGO and reduces the likelihood of belonging to RICE. The experimental results show that the proposed model outperforms the single-label counterparts on standard text corpora.

Multi-label classification algorithms discussed above have high computational cost. In order to reduce computational cost, a novel multi-label classifier is proposed by Zhihua Wei et al. [9]. In this research work, naive Bayesian multi-label classification (NBML) algorithm is proposed that incorporates two-step feature selection strategy. In feature selection, subsets of discriminative features that occur in training set are selected to improve classification quality and reduce computational complexity. In the first step, document frequency and X2 are used for feature selection. In the second step, FCFB (fast correlation-based filter selection) is applied on the results obtained in first

step which leads to the reduction in feature dimensionality. When experimented on real World Wide Web dataset, NBML performs equivalent to other multi-label algorithms.

All the methods mentioned above do not consider the correlations among labels resulting into low classification performance. Many researchers have worked upon to deal with label correlations in multi-label classification problem which is discussed in the following paragraphs.

In [10], a generative probabilistic model, the correlated labeling model (Col Model) is proposed. The main aim of Col Model is to capture the information conveyed in the class membership, to exploit the in-depth relation between the classes and words: via the latent topic factors and to predict the potential classes in an unseen document classification. It is a supervised model and employs multivariate normal distribution to capture the correlation between the classes. Experimental results show that Col Model possesses good precision and recall values and classification performance increases significantly when correlation among classes is considered.

Zhang propose a multi-label naive Bayes (MLNB) [11] method to deal with multi-label classification problem. The author incorporates a two-stage filter wrapper feature selection strategy to improve the performance of MLNB. In the first stage, principal component analysis (PCA) is used to eliminate irrelevant and redundant features. In the second stage, genetic algorithm (GA) is used to optimize the classification by explicitly considering correlations among labels of different instances through fitness function. Experiments show that the proposed approach performs effectively on synthetic as well as on real-world datasets. Though incorporation of PCA and GA improves the performance, at the same time it increases the time complexity for high dimensional dataset.

A second-order CRF (conditional random field) model is proposed [12] for multi-label image classification, to capture the semantic associations between labels. In the proposed model, the feature weights are initialized differently and then voting technique is applied to improve the performance. Multiple CRFs are obtained iteratively, and each CRF vote for several labels. For each label, if the vote number exceeds the predefined threshold, it is regarded as final labels for an image. The results show the effectiveness of this method when applied to MSRC dataset.

To address the inherent correlations among multiple labels, Haiping Ma et al. [13] proposed a generative model named labeled four-level pachinko allocation model (L-F-L-PAM). The proposed algorithm is based on labeled LDA model, and an additional latent correlations level is added to enhance the performance. Apart from this, pruned Gibbs sampling is used for inferring the unlabeled test documents which results in reduced inference time in the test stage. The results of the experiments conducted on text dataset: Reuters-21578 corpus and Web pages from yahoo.com show that by considering the relations between multiple labels, the overall performance and computational efficiency of multi-label classification task is improved.

3.3 Support vector machines

Support vector machines (SVM) is technically defined by a hyperplane. The hyperplane is produced as an output of the SVM algorithm. This hyperplane classifies the new example in the labeled training dataset. An optimal hyperplane is the one that passes as far as possible from all points.

Elisseeff and Weston [14] present rank SVM for multi-label classification. This is a linear model that uses ranking loss as its cost function where ranking loss is defined as average fractional pairs of labels that are ordered incorrectly. Rank SVM aims to minimize this cost function. When experiments were conducted, rank SVM results in improvements when compared to other multi-label algorithms. However, the author reported that the worst case space complexity of this proposed algorithm is mQ2, which increases when Q is large.

The SVMs and other discriminative classification methods are designed to assign an instance to one among the set of disjoint classes, but high degree of correlation and overlapping exist among classes in multi-labeled data. In order to deal with this problem, Godbole and Sarawagi [15] proposed two enhancements for support vector machine. The first improvement is the algorithm that deals with the correlation between classes. This is done by extending original dataset with |C| extra features. The binary classifiers are then trained on this extended dataset. The second improvement is handling the overlapping classes in multi-label classification by modifying the margins of SVMs. This is achieved by one of the two methods: (1) by removing similar negative instances that lie very close to the resultant hyperplane and (2) removing all the training instances of confusion classes in confusion matrix.

In [16], two algorithms based on SVMs are proposed for multi-label classification problem. The first algorithm is based on "one against rest" strategy in which the multi-label training set is decomposed into binary classifiers and then k binary SVM sub-classifiers are trained and the membership vector is obtained as per sub-classifiers according to which classification of the text is performed. The second algorithm HSMC is based on the hyper-sphere, multi-label classification algorithm for classification of a dataset having higher number of samples and more classes. The experimental result made on Reuters 21578 shows the effectiveness of both algorithms.

Hariharan et al. [17] proposed a max-margin multi-label classification formulation referred to as M3L. The author incorporates the prior knowledge about densely correlated labels to improve the performance to M3L. Further, SMO (sequential minimal optimization) is adapted for optimizing the above formulation. Basically, SMO breaks large quadratic programming (QP) problem into set of smallest possible QP problems, each of which is solved analytically. Experiments show that by incorporating prior knowledge, M3L (max-margin multi-label classification) could improve prediction accuracy over independent methods.

In order to overcome the drawback of rank SVM, Jianhua Xu [18] proposed multi-label support vector machine (ML-SVM) algorithm. In this novel algorithm, a zero label is added to proposed SVM architecture derived from rank SVM. A new form of cost function is also introduced that reduces the computational cost. The multi-label problem is first decomposed into several subproblems by applying problem transformation technique, and then, Frank-Wolfe method is used to train these subproblems. When experiments were conducted, SVM-ML proved to be stronger than the ML-KNN, ML-RBF, ML-NB, BP-MLL and rank SVM.

3.4 k-Nearest neighbor

The k-nearest neighbor is based on instance-based learning. The classification of a test tuple is done based on the majority vote of the k neighbors that are closest to the test tuple. Due to its simplicity and high performance when subjected to large training set, it is applied to multi-label classification.

Multi-label k-nearest neighbor (ML-kNN) is introduced by Zhang and Zhou [19] which uses the basic concept of k-nearest neighbor. For each test tuple, it first identifies its k-nearest neighbors and according to the classes assigned to these neighbors, test tuple is classified using maximum a posteriori (MAP). Results of experiments prove that the performance of ML-kNN is equivalent to rank SVM and higher than boosting algorithms.

Spyromitros et al. [20] proposed k-NN in conjunction with binary relevance (BR) problem transformation method known as BR-kNN. When BR is paired with k-NN, same process of calculating kNN is performed L (total number of labels) times. In the proposed BR-kNN, independent predictions are made for each label followed by single k-nearest neighbor search. The author identifies two extensions of BR-kNN to improve the performance. The first extension known as BR-kNN-a handles the empty set that may be produced as an output of BR. In such case, BR-kNN-a outputs the label with highest confidence. The second extension BR-kNN-b works in two steps: In the first step, it calculates the average size of label set of k-nearest neighbor, and in second step, the label with highest confidence is produced. Results show that BR-kNN-a dominates in the scene and the emotion dataset, whereas BR-kNN-b dominates in yeast dataset.

In [21], multi-label k-nearest Michigan particle swarm optimization (ML-KMPSO) hybridizes MPSO (Michigan particle swarm optimization) and ML-kNN (multi-label k-nearest neighbor). At first, MPSO breaks the MLC into subclassification problems without considering the label correlation. And then ML-kNN is used to establish the correlation among classes. When experimented on two real-world datasets: yeast and scene, the proposed algorithm outperforms other multi-label classification algorithms.

3.5 Neural network

An ANN (artificial neural network) is a computational model that is analogous to the human brain. An ANN is constructed with few basic building blocks, called nodes or neurons. These neurons are connected to each other with the help of connection link. Each connection link is associated with weights that store information. This information is then used by ANN to solve the particular problem. Due to their ability to solve complex problems for which algorithmic solutions do not exist, ANN has become very popular for solving multi-label classification problem.

Zhang and Zhou [22] proposed first neural network-based algorithm for multi-label classification and named it as backpropagation for multi-label learning (BP-MLL). In this work, a single hidden layer feed-forward BP-MLL neural network is used with sigmoidal neurons and bias parameters in the hidden and input layers. The number of output layer neuron is equal to the number of labels. Training is based on the traditional BP (backpropagation) algorithm. But to deal with the correlation between labels, a global error function is proposed in this paper:

$$E = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p \times \bar{Y}_p} e^{-(c_r^p - c_s^p)}}{Y_p \times \bar{Y}_p} \tag{2}$$

Here, m is number of multi-label instances in training set. $Y_p \subseteq Y$ is set of labels assigned to pth training instances. C_q^p is the actual output of the qth neuron. $\bar{Y}_p$ is the complementary set of Y_p.

If the output value of neuron is higher than predefined threshold value, then corresponding label belongs to the input instance else not. Experiments in functional genomics and text categorization dataset show that it performs better than well-established multi-label learning algorithms.

In [23], author proposes some modifications in the error function proposed in [22]. The first modification is the

incorporation of the threshold value into the error function used in BP-MLL. A generalization of error function is done by adding independent thresholds for different labels. The results show that proposed modification improves the performance of multi-label classifiers based on neural network.

In [24], radial basis neural network for multi-label (ML-RBF) learning is proposed. The training procedure of ML-RBF is a two-stage process. In first step, k-means clustering is applied on set of samples of each possible class. The centroids so obtained are then used to determine the parameters of the basis functions. In the second stage, weights are adjusted to minimize the sum-of-square error function. This algorithm when applied to three real-world datasets proves its efficiency as well as its effectiveness in comparison with other algorithms.

Apart from RBF, ART (adaptive resonance theory) is also applied for multi-label classification. Sapozhnikova [25] presents the extension of fuzzy ARTMAP for multi-label classification called multi-label-FAM. In the proposed methodology, a best category set with high activation values are produced based on the fact that if the relative difference in activations of a category lies below a predefined threshold, then it is included in the set. After normalizing these activation values, the resultant prediction is obtained by calculating weighted sum of individual predictions. Postprocessing filter is used to produce the labels, having score more than predefined fraction of the highest score. When experimented on yeast dataset, the performance of the proposed classifier is comparable with the performance of other multi-label classifiers except for ML-kNN.

Data mining system-based new hierarchy extraction (HE) algorithm is proposed in [26] in which fuzzy ARTMAP is hybridized with HE. The proposed HE algorithm deals with the hierarchies that exist between classes, taking into account relationships between the labels assigned by the classifier by building association rules from label co-occurrences. The main advantage of this data mining system is that it enables HE for sparse multi-label data even if many instances remain single-labeled. Experimental results show that the proposed approach is suitable for extracting class hierarchies from predicted multi-labels.

De Souza et al. [27] proposed an effective machine learning technique which provides fast training and testing along with simple implementation for automatic multi-label text categorization systems known as VG-RAM WNN (virtual generalizing random access memory weightless neural networks). RAM-based neural networks use RAM to store knowledge instead of connections. The networks input values are used as the RAM address, and the value stored at this address is the neuron's output. When tested on two real-world datasets, i.e., categorization of free text descriptions of economic activities and categorization of Web pages, VG-RAM WNN outperforms ML-kNN.

Implementation simplicity and high computational speed during the training phase of probabilistic neural network (PNN) motivates Ciarelli et al. [28] to propose a modified version of PNN to solve the multi-label classification problem. Basically, PNN is an implementation of a statistical algorithm called kernel discriminant analysis in which the operations are organized into a multilayered feed-forward network with four layers. However, the proposed version of PNN is composed of three layers but like original PNN requires only one training step. Comparative experimental evaluation on a yahoo and economic activities database proved that PNN is superior to other algorithms.

Chen et al. [29] proposed an algorithm that consists of two stages of a multilayer perceptron (MLP), named multi-instance multi-label neural network (MIMLNN). The first stage of MLP is used to establish the relationship between image regions and labels, whereas the second stage of MLP captures the label correlation needed for multi-label classification. The training of MIMLNN is done by Rprop, which is a refined form of backpropagation algorithm. Results of experiments conducted on synthetic dataset and the popular coral dataset demonstrate the superior performance of MIMLNN for multi-instance multi-label image classification.

Although many approaches have been proposed by different researchers to solve multi-label classification problem, population-based meta-heuristics approaches are yet to be explored. So, this paper introduces two new algorithms for multi-label classification: The first algorithm is fuzzy PSO-based ML-RBF in which fuzzy PSO algorithm is used to optimize the RBF networks connection weights w, whereas in the second algorithm, RBF is trained using traditional training method singular value decomposition (SVD) and is named as FSVD-MLRBF.

4 Proposed methodology

The proposed algorithms are based on ML-RBF (multi-label radial basis function) neural network as shown in Fig. 1. The training of ML-RBF [24] can be done in two stages: At the first stage, we determine the centers and RBF (radial basis function) by applying fuzzy c-means clustering and Gaussian activation function, respectively. This leads to the formation of nodes in the hidden layer. In the second stage, we determine the weights between hidden layer and output layer. In our proposed methods, we have trained the neural network using two different approaches: In the fuzzy PSO-based ML-RBF, we use fuzzy PSO

(fuzzy particle swarm optimization) to obtain optimized weights, whereas in FSVD-MLRBF, the weights are determined using traditional SVD.

4.1 Phase 1: determination of hidden layer

Let D be a multi-label dataset and $L = 1, 2, \ldots, L$ be the label set, such that $D = \{(x_i, Y_i)|1 \leq i \leq m\}$, where x_i is a single instance and $Y_i \epsilon L$ is the set of labels to which x_i belongs.

In the first step, fuzzy c-means clustering is applied on the set of instances with label $l \in L$ so as to obtain k_l clustered groups and j centroids for each $l \in L$. The number of clusters in each case is determined using following formula used in [24], i.e.,

$$k_l = \alpha \times |U_l| \tag{3}$$

where α is a fraction of the number of instances U_l. k_l and c_j together form the basis function. Here, Gaussian activation function is used as the basis function which is given by:

$$\Phi_j(x_i) = \exp\left(-\frac{\text{dist}(x_i, c_j)^2}{2\sigma^2}\right) \tag{4}$$

Here $\text{dist}(x_i, c_j)$ is the Euclidean distance between x_i and the jth centroid c_j. σ is the smoothing parameter, and for all the centroids, its value remains same and is calculated using:

$$\sigma = \mu \times \left(\frac{\sum_{p=1}^{K-1}\sum_{q=p+1}^{K}\text{dist}(c_p, c_q)}{K(K-1)/2}\right) \tag{5}$$

where μ is the scaling factor. Thus, for each label k_l, basis functions are obtained. Hence, the total number of basis functions hold on to the hidden layer is given as $K = \sum_{l=1}^{L} k_l$. All the basis functions so obtained are then put together and re-indexed.

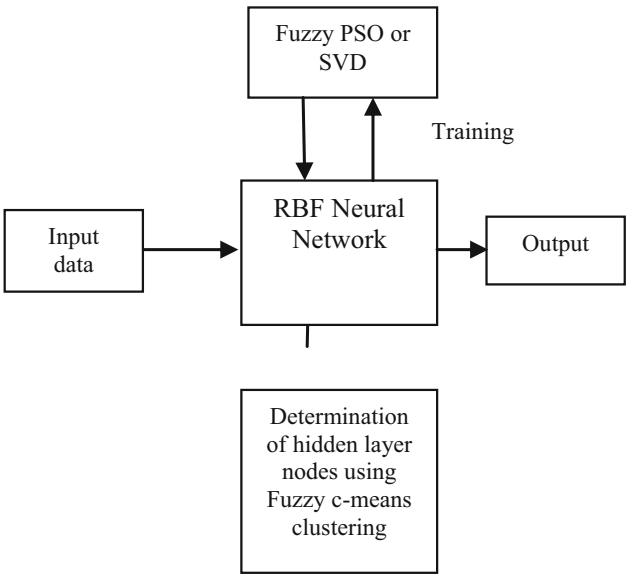

Fig. 1 Proposed architecture of fuzzy PSO-based ML-RBF and FSVD-MLRBF for multi-label classification

4.2 Phase 2: determination of weights between hidden layer and output layer

Different methods can be applied to train a neural network. The two methods that are used in our proposed work are described in the next paragraph.

4.2.1 Weight adjustment using fuzzy PSO algorithm

In fuzzy PSO [30], instead of only one best particle in the neighborhood, multiple particles in the neighborhood are allowed to influence other particles. Each group member is assigned with the multiplier known as charisma which is a fuzzy variable. The influence of each best particle to others is calculated using this charisma variable, and summation is then applied to the original formulation. Thus, each particle will update its position and velocity using following two formulas:

$$x_{id} = x_{id} + V_{id} \tag{6}$$

$$V_{id} = wV_{id} + c_1 * rand() * (p_{id} - x_{id}) + \sum_{h=1}^{k} c_2 * rand() * \Psi(p_{hd})(p_{hd} - x_{id}) \tag{7}$$

where w is the inertia weight, c_1 and c_2 are the acceleration coefficients, *rand*() is independent random number, and $\Psi(p_{hd})$ is charisma function and is defined as follows:

$$\Psi(p_h) = \frac{1}{1 + \left(l\frac{f(p_h) - f(p_g)}{f(p_g)}\right)^2} \tag{8}$$

$f(\cdot)$ is the fitness function, and l is a user-specified parameter. The key of the fuzzy PSO algorithm is to choose the fitness function. Here, sum-of-square error is used as a fitness function as shown below:

$$E = \frac{1}{2}\sum_{i=1}^{m}\sum_{l=1}^{L}\left(y_l(x_i) - t_l^i\right)^2 \tag{9}$$

In this formula, t_l^i is the desired output of x_i on the lth class. $y_l(x_i)$ is the actual output of x_i on the lth class and is given by $y_l(x_i) = \sum_{j=1}^{L+1} w_{jl}\Phi_j(x_i)$.

The objective is to find such a particle that minimizes the output of the fitness function. At first step, initialize the position and velocity of each particle. Set the values of k and l. At second step, calculate the fitness of each particle using fitness function according to formula (9). In each iteration of neural network training, the particle with smallest fitness function value is considered as gbest,

Table 7 Multi-label datasets statistics (LCard—label cardinality, LDen—label density, DL—distinct label sets, PDL—proportion of distinct label)

Dataset	Domain	Instances	Distinct label	LCard	LDen	DL	PDL
Yeast	Biology	2417	14	4.237	0.303	198	0.081
Scene	Multimedia	2407	6	1.074	0.179	15	0.006

which is best position of all the particles till now. Each particle will update its position and velocity using Eqs. (6) and (7), respectively. This process terminates when the minimum fitness function value is achieved or maximum number of iterations is met.

4.2.2 Weight adjustment using SVD

Due to the quadratic nature of the error function, it can be solved using set of linear equations. In order to find the solution, differentiate Eq. (9) with respect to w_{jl} and set the derivative to zero. This gives the normal equation for the least sum-of-square problem:

$$\left(\Phi^T \Phi\right) W = \Phi^T T \tag{10}$$

Here, Φ has dimensions $m \times (K+1)$ with elements $\Phi_j(x_i)$, W has dimensions $(K+1) \times L$ with elements w_{jl}, and T has dimensions $m \times L$ with elements t_l^i. The weights are calculated using Eq. (8) using linear matrix inversion techniques of SVD.

Correlation between the classes in RBF is achieved as all the nodes in the hidden layer are connected to all the nodes in the output layer. Thus, information present in the hidden layer nodes is completely used to determine the optimized weights as well as for predicting instances for class l.

5 Experiments

5.1 Dataset

The performance of the two proposed approaches is evaluated on the yeast dataset and scene dataset. Yeast dataset describes the genes of Yeast saccharomyces cerevisiae. Each gene is described by the concatenation of micro-array expression data and phylogenetic profile and is associated with a set of functional classes whose maximum size can be potentially more than 190. In order to make it easier, Elisseeff and Weston [14] preprocessed the dataset where only the known structure of the functional classes is used. Actually, the whole set of functional classes is structured into hierarchies up to 4 levels deep3. In this paper, the same dataset as used in the literature [14] is adopted. In this dataset, only functional classes in the top hierarchy are considered. The resulting multi-label dataset contains 2,417 genes each represented by a 103-dimensional feature vector. There are 14 possible class labels, and the average number of labels for each gene is $4{:}24 \pm 1{:}57$.

The scene dataset is composed of 2407 instances each of which is represented by 294 feature vector. This dataset contains the scenes data. There are 6 classes used in this dataset. The summary of both the datasets is given below (Table 7).

5.2 Parameter setting

For comparative study, we coded these methods by using the java-based eclipse. For our experimental tests, we used i3-2330 processor @2.20 GHz and with 3 GB RAM. The values of different parameters used are discussed in Table 8.

The other parameters that govern the performance of RBF neural network are α and μ, whose value is set to $\alpha = 0.01$ and $\mu = 1$ as in [24]. Fuzzy PSO search dimension (RBF networks weights number) is the product of number of nodes in the hidden layer and number of nodes in the output layer of the neural network. Here, the number of hidden nodes is obtained as a result of clustering, and the number of output nodes is equals to the number of classes in the dataset. So for yeast dataset, number of nodes in output layer are 14, and in case of scene dataset, they are 6. Results are obtained in 50 runs of fuzzy PSO. The termination criterion of fuzzy PSO is either 1000 iterations or till the value of fitness function reaches 0.001.

The experimental results are obtained by performing tenfold cross-validation on the yeast and scene dataset. In detail, the original dataset is randomly divided into ten parts each with approximately the same size. In each fold,

Table 8 Values of various parameters used in experiments

Parameter	Range
w	0.4
c_1	2
c_2	2
k	3
l	1.6
E	0.001
m	1.25
$V_{\max}$	0.5, 1, 2
Number of particles	20

one part is held out for testing and the learning algorithm is trained on the remaining data. The above process is iterated ten times so that each part is used as the test data exactly once, where the averaged metric values out of ten runs are reported for the algorithm.

5.3 Classification results

The performance of multi-label classifier is measured using five evaluation measures which are: Hamming loss, one error, coverage, ranking loss and accuracy.

1. *Hamming loss* is defined as the number of times an instance is misclassified. It is given by

$$\mathrm{HL} = \frac{1}{n}\sum_{i=1}^{n}\frac{|Z_i \Delta Y_i|}{L} \tag{11}$$

Δ is the symmetric difference between two sets. Z_i is the set of labels predicted by the classifier, and Y_i is the desired set of labels for a given test instance x_i.

2. *One error* measures the number of times top-ranked label is not in the set of labels of an instance. It is given by

$$\mathrm{OE} = \frac{1}{n}\sum_{i=1}^{n}\partial([\arg\max f(x_i, l)] \notin Y)_i \tag{12}$$

Here, ∂ is a function that produces 1 if the argument is true and 0 otherwise.

3. *Coverage* is used to determine the number of steps needed to cover all the proper labels of the instance, given by

$$\mathrm{Coverage} = \frac{1}{n}\sum_{i=1}^{n}\max \mathrm{rank}(x_i, l) - 1 \tag{13}$$

4. *Ranking loss* determines the number of label pairs that are reversely ordered for an instance. It is given by

$$\mathrm{RL} = \frac{1}{n}\sum_{i=1}^{n}\frac{|D_i|}{|Y_i||\bar{Y}_i|} \tag{14}$$

where $\bar{Y}_i$ is complementary of Y_i, while

$D = \{(l_1, l_2)|f(x_i, l_1) \leq f(x_i, l_2), (l_1, l_2) \in Y_i \times \bar{Y}_i\}$.

All the performance measures described above have best performance when value is zero which implies that smaller the value, the better the performance of the classifier.

Table 9 shows the experimental results of proposed algorithms, ML-RBF and ML-KNN on yeast dataset. The values of ML-RBF used in Tables 9 and 11 have been taken from [24] and of ML-KNN used in Tables 9 and 11 have been taken from [19]. The results have been reported as mean ± SD.

Table 10 shows statistical unpaired t test (at 95 % confidence interval) results of all the four algorithms. As shown in the tables, for each evaluation metric, $A < B$ indicates that performance of B is significantly better than that of A.

From the experimental results of Tables 9 and 10, the two proposed algorithms, i.e., fuzzy PSO-based MLRBF and FSVD-MLRBF, outperforms ML-RBF and ML-KNN in terms of one error and performs equally on the other three measures. In terms of one error, fuzzy PSO-based MLRBF is outperforming FSVD-MLRBF, while in case of

Table 9 Comparison of performance (mean ± SD) on yeast dataset

Evaluation measures	Fuzzy PSO-based ML-RBF (A_1)	FSVD-MLRBF (A_2)	ML-RBF (A_3)	ML-KNN(A_4) ($k = 8$)
Hamming loss	0.260 ± 0.110	0.291 ± 0.151	0.195 ± 0.011	0.195 ± 0.010
One error	0.181 ± 0.076	0.306 ± 0.382	0.233 ± 0.037	0.233 ± 0.032
Coverage	5.277 ± 0.570	3.743 ± 2.145	6.352 ± 0.244	6.291 ± 0.0238
Ranking loss	0.298 ± 0.170	0.224 ± 0.157	0.169 ± 0.017	0.169 ± 0.016

Table 10 Statistical analysis of each algorithm on yeast dataset

Algorithm	Hamming loss	One error	Coverage	Ranking loss
Fuzzy PSO-based ML-RBF—FSVD-MLRBF	$A_1 \approx A_2$	$A_1 > A_2$	$A_1 < A_2$	$A_1 \approx A_2$
Fuzzy PSO-based ML-RBF—ML-RBF	$A_1 \approx A_3$	$A_1 > A_3$	$A_1 > A_3$	$A_1 < A_3$
Fuzzy PSO-based ML-RBF –ML-KNN	$A_1 \approx A_4$	$A_1 > A_4$	$A_1 > A_4$	$A_1 < A_4$
MLRBF—ML-KNN	$A_3 \approx A_4$	$A_3 \approx A_4$	$A_3 \approx A_4$	$A_3 \approx A_4$
FSVD-MLRBF—ML-RBF	$A_2 \approx A_3$	$A_2 \approx A_3$	$A_2 > A_3$	$A_2 \approx A_3$
FSVD-MLRBF—ML-KNN	$A_2 \approx A_4$	$A_2 \approx A_4$	$A_2 > A_4$	$A_2 \approx A_4$
Total order : (fuzzy PSO-based ML-RBF(2) = FSVD-MLRBF(2))>(ML-RBF(−2)>ML-KNN(−2))				

Table 11 Comparison of performance (mean ± SD) on scene dataset

Criteria	Fuzzy PSO-based ML-RBF (A_1)	FSVD-MLRBF (A_2)	ML-RBF (A_3)	ML-KNN(A_4)
Hamming loss	0.211 ± 0.097	0.226 ± 0.119	0.163 ± 0.015	0.169 ± .016
One error	0.150 ± 0.183	0.196 ± 0.214	0.294 ± 0.033	0.300 ± .046
Coverage	0.901 ± 0.595	0.892 ± 0.711	0.904 ± 0.087	0.939 ± .100
Ranking loss	0.503 ± 0.162	0.552 ± 0.113	0.158 ± 0.020	0.168 ± .024

Table 12 Statistical analysis of each algorithm on scene dataset

Algorithm	Hamming loss	One error	Coverage	Ranking loss
Fuzzy PSO-based ML-RBF—FSVD-MLRBF	$A_1 \approx A_2$	$A_1 \approx A_2$	$A_1 \approx A_2$	$A_1 \approx A_2$
Fuzzy PSO-based ML-RBF—ML-RBF	$A_1 \approx A_3$	$A_1 > A_3$	$A_1 \approx A_3$	$A_1 < A_3$
Fuzzy PSO-based ML-RBF—ML-KNN	$A_1 \approx A_4$	$A_1 > A_4$	$A_1 \approx A_4$	$A_1 < A_4$
FSVD-MLRBF—ML-RBF	$A_2 \approx A_3$	$A_2 > A_3$	$A_2 \approx A_3$	$A_2 < A_3$
FSVD-MLRBF—ML-KNN	$A_2 \approx A_4$	$A_2 > A_4$	$A_2 \approx A_4$	$A_2 < A_4$
MLRBF—ML-KNN	$A_3 \approx A_4$	$A_3 \approx A_4$	$A_3 \approx A_4$	$A_3 \approx A_4$
Total order: fuzzy PSO-based ML-RBF(0) = FSVD-MLRBF(0) = ML-RBF(0) = ML-KNN(0)				

Table 13 Comparison of accuracy of each algorithm on yeast and scene dataset

Dataset	Fuzzy PSO-based ML-RBF	FSVD-MLRBF	ML-RBF	ML-KNN
Yeast	0.515	0.539	0.464	0.435
Scene	0.679	0.714	0.581	0.576

coverage, FSVD-MLRBF is outperforming fuzzy PSO-based MLRBF on yeast dataset.

Taking the above situation into consideration, a score is assigned to each compared algorithm in order to give an overall performance assessment of the algorithm as stated in [24]. Concretely, for each evaluation metric and each possible pair of algorithms A and B, if A > B holds, the score of A is added by 1 and that of B is subtracted by 1 accordingly. With the accumulated score of each algorithm, a total order ">" can be defined on the set of all compared algorithms as shown in the last lines of Tables 10 and 12. Here, A > B denotes that A outperforms B on the corresponding dataset and the accumulated score of each algorithm is reported in the parentheses. From this score, it is clear that both the proposed algorithms outperform ML-RBF and ML-KNN.

Tables 11 and 12 show experimental results and statistical unpaired t test (at 95 % confidence interval) results of all the four algorithms on scene dataset.

Experimental results on scene dataset as shown in Tables 11 and 12 allow us to conclude that the two proposed algorithms, i.e., fuzzy PSO-based MLRBF and FSVD-MLRBF either meets or beats its competitors ML-RBF and ML-KNN in terms of all the three measures Hamming loss, one error and coverage except ranking loss. This may be because we have more number of instances of scene dataset compared to [24]. Overall score shows that the performances of all the algorithms are equivalent when compared to scene dataset.

5. *Accuracy (A):* Accuracy for each instance is defined as the proportion of the predicted correct labels to the total number (predicted and actual) of labels for that instance. Overall accuracy is the average across all instances.

$$A = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i \cap Z_i}{Y_i \cup Z_i}\right| \tag{15}$$

where Z_i is the set of predicted labels for X_i, L is the set of finite set of possible labels, n is the total number of test samples, and Y_i is the proper label set for an instance X'_i. For the optimum performance, the accuracy of the algorithm must be high.

Table 13 reports the comparative results of average accuracy of all the four algorithms on yeast and scene dataset which were obtained on 10 independent runs. Experimental result shows that both the proposed algorithms outperform ML-RBF and ML-KNN, and among the two, FSVD-MLRBF gives superior performance on the basis of accuracy.

6 Conclusion

Multi-label classification is a generalization of multi-class, where each instance is assigned to a subset of labels. Researchers have worked to solve the multi-label problem using both the approaches, i.e., via problem transformation and algorithm adaptation. However, through this paper, we have focused on the work under algorithm adaptation and concluded that the algorithms described here handle various issues of multi-label classification, especially the exploitation of correlations among labels, predicting multiple classes for unseen instances, minimization of ranking as well as Hamming loss.

In this paper, we have also proposed two approaches: fuzzy PSO-based MLRBF and FSVD-MLRBF. The experiments were conducted on two real-world datasets, i.e., yeast and scene dataset. Results of experiments show that both these algorithms can be successfully used for solving multi-label classification task. The analysis of results proves that both the proposed fuzzy PSO-based MLRBF and FSVD-MLRBF algorithms are equally efficient and effective when applied on RBF network to solve multi-label classification problem. However, the proposed FSVD-MLRBF algorithm gives more accurate results on both the datasets as compared to all the other three algorithms. In future, one can also use other multi-label datasets of varying complexity to fully evaluate the performance of both the proposed algorithms.

Compliance with ethical standards

Conflict of interest We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome. We also confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed.

References

1. Tsoumakas G, Katakis I (2007) Multi-label classification: an overview. Int J Data Wareh Mining 3(1):1–13
2. Read J (2008) A pruned problem transformation method for multi-label classification. In: Proceedings of the 2008 New Zealand computer science research student conference, NZCSRS '08, pp 143–150
3. Tsoumakas G, Vlahavas I (2007) Random k-labelset: an ensemble method for multi-label classification. In: Proceedings of the 18th European conference on machine learning, ECML'07, Warsaw, pp 406–417
4. Hullermeier E, Furnkranz J, Chang W, Brinker K (2008) Label ranking by learning pairwise preferences. Artif Intell 172(16–17):1897–1916
5. Furnkranz J, Hullermeier E, Mencia EL, Brinker K (2008) Multi-label classification via calibrated label ranking. Mach Learn 73(2):133–153
6. Schapire RE, Singer Y (2000) Boostexter: a boosting-based system for text categorization. Mach Learn 39(2–3):135–168
7. Ueda N, Saito K (2002) Single-shot detection of multiple categories of text using parametric mixture models. In: Proceedings of the KDD, pp 626–631
8. Ghamrawi N, McCallum A (2005) Collective multi-label classification. CIKM, Bremen
9. Wei Z, Zhang H, Zhang Z, Li W, Miao D (2011) A naive bayesian multi-label classification algorithm with application to visualize text search results. Int J Adv Intell 3(2):173–188
10. Wang H, Huang M, Zhu X (2008) A generative probabilistic model for multi-label classification. Eighth IEEE international conference on data mining, pp 1550–4786
11. Zhang M-L, Pena Jose M, Robles V (2009) Feature selection for multi-label naive Bayes classification. Inf Sci 179:3218–3229
12. Wang X, Liu X, Shi Z, Shi Z, Sui H (2010) Voting conditional random fields for multi-label image classification. In: 3rd International congress on image and signal processing (CISP), vol 4, pp 1984–1988
13. Ma H, Chen E, Xu L, Xiong H (2012) Capturing correlations of multiple labels: a generative probabilistic model for multi-label learning. Neurocomputing 92:116–123
14. Elisseeff A, Weston J (2002) A kernel method for multi-labelled classification. In: Dietterich TG, Becker S, Ghahramani Z (eds) Advances in neural information processing systems, vol 14., MIT PressCambridge, MA, pp 681–687
15. Godbole S, Sarawagi S (2004) Discriminative methods for multi-labeled classification. LNCS Adv Knowl Discov Data Mining 3056(1):22–30
16. Qin Y-P, Wang X-K (2009) Study on multi-label text classification based on SVM. In: Proceeding FSKD'09 Proceedings of the 6th international conference on fuzzy systems and knowledge discovery, vol 1, pp 300–304
17. Hariharan B, Vishwanathan SVN, Varma M (2012) Efficient max-margin multi-label classification with applications to zero-shot learning. Mach Learn J 88(1):127–155
18. Jianhua Xu (2012) An efficient multi-label support vector machine with a zero label. Expert Syst Appl 39:4796–4804
19. Zhang M-L, Zhou Z-H (2007) ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn 40(7):2038–3048
20. Spyromitros E, Tsoumakas G, Vlahavas I (2008) An empirical study of lazy multilabel classification algorithms. In: Proceedings of the 5th Hellenic conference on artificial intelligence (SETN 2008) Springer, LNAI, vol 5138, pp 401–406
21. Coelho TA, Esmin AAA, Junior WM (2007) Particle swarm optimization for multi-label classification. GECCO ACM, New York
22. Zhang M-L, Zhou Z-H (2006) Multi-label neural networks with applications to functional genomics and text categorization. IEEE Trans Knowl Data Eng 18(10):1338–1351
23. Grodzicki R, Mandziuk J, Wang L (2008) Improved multi-label classification with neural networks. LNCS Adv Knowl Discov Data Mining 5199(1):409–416
24. Zhang M-L (2009) ML-RBF: RBF neural networks for multi-label learning. Neural Process Lett 29(2):61–74
25. Sapozhnikova EP (2009) Art-based neural networks for multi-label classification. In: Adams NM, Robardet C, Siebes A, Boulicaut J-F (eds) IDA, series. Lecture notes in computer science, vol 5772. Springer, New York, pp 167–177
26. Benites F, Brucker F, Sapozhnikova E (2010) Multi-label classification by ART-based neural networks and hierarchy extraction. In: Proceedings of the international joint conference on neural networks (IJCNN), pp 1–9
27. De Souza AF, Pedroni F, Oliveira E, Ciarelli PM, Henrique WF, Veronese L, Badue C (2009) Automated multi-label text categorization with VG-RAM weightless neural networks. Neurocomputing 72:2209–2217

28. Ciarelli PM, Oliveria E, Badue C, De Souza AF (2009) Multi-label text categorization using a probabilistic neural network. Int J Comput Inf Syst Ind Manag Appl (IJCISIM) 1:133–144. ISSN: 2150-7988
29. Chen Z, Chi Z, Hong Fu, Feng D (2013) Multi-instance multi-label image classification: a neural approach. Neurocomputing 99:298–306
30. Abdelbar AM, Abdelshahid S, Wunsch II DC (2005) Fuzzy PSO: a generalization of particle swarm optimization. In: Proceedings of IEEE international joint conference on neural networks (IJCNN), vol 2, Montereal