Sensitivity analysis on initial classifier accuracy in fuzziness based semi-supervised learning

Muhammed J. A. Patwary, Xi-Zhao Wang*

Big Data Institute, College of Computer Science and Software Engineering, The Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, 518060, China

Abstract

Semi-supervised learning can be described from different perspectives, which plays a crucial role in the study of machine learning. In this study, a new aspect of semi-supervised learning is explored by investigating the divide-and-conquer strategy based on fuzziness to improve the performance of classifiers. In such an approach, adding a category of samples with low fuzziness in the training set can improve the training accuracy, which is experimentally confirmed and explained in the theory of learning from noisy data. The significance of initial accuracy of a base classifier in improving classifier's performance is further studied. It is observed that the initial accuracy of a base classifier has a significant impact on the improvement of classifier's performance. Experimental results exhibit that the improvement of accuracy, which is sensitive to the base classifier, attains its maximum when the initial accuracy is between 70% and 80%.

Keywords:
Fuzziness, Semi-supervised learning, Generalization, Divide-and-conquer technique, Fuzzy classifier, Learning from noisy data.

1. Introduction

Semi-supervised learning (SSL) is a machine learning prototype which is considered as one of the most potential techniques in the study of machine learning, data mining and pattern recognition. It exploits the benefits of both supervised and unsupervised learning techniques to create a new type of learning technique. The learning task of semi-supervised learning is handled by using both labeled and unlabeled data [32] [50] . Studies on machine learning show that, in general, using of unlabeled data along with a little amount of labeled data can substantially improve the accuracy of a learning model [45] [13]. However, usage of labeled data in a learning system faces several challenges including: (1) obtaining labeled data is costly and time-consuming, (2) domain knowledge is also required in order to deal with and process such data, (3) labels, annotations etc. are rarely available for common uses. On the other hand, unlabeled data are prevalent and

*Corresponding author
Email Address: jamshed@szu.edu.cn (Muhammed J. A. Patwary), xizhaowang@ieee.org,xwang@szu.edu.cn (Xi-Zhao Wang)

reasonable in nature. Semi-supervised learning mostly uses unlabeled data in association with some labeled data to improve classification accuracy.

The term fuzziness was first introduced by Lotfi A. Zadeh in connection with his famous fuzzy set theory [41] [28]. Fuzziness refers to the inexactness prevailing in an unclear event which is very common in our daily 15 life activities. The concept of fuzziness has been successfully applied to many machine learning applications as well as in many other application areas. For example, a classifier can classify an instance to a class with a degree of belief, which says that, to what extent the instance belongs to that specific class.

Over the course of last couple of decades semi-supervised learning was considered as one of the most important learning techniques. Several studies have been carried out to investigate the different aspects of it [10, 22, 24, 34] [19, 17, 15, 25]. In the light of recent events in the study of semi-supervised learning, now there is some concern about the issues of fuzziness. As a result, fuzziness based SSL has received much attention from researchers [5, 33, 46, 50] [3, 17, 27, 35] [28, 40, 46, 22]. In general, the working principle of SSL can be viewed as two folds technique. In the first step an initial classifier is trained with small volume of training data. In its second step huge amount of unlabeled data is used for assigning each data-point to one of several class labels.

One of the important aims of this study is to clarify the interrelationship between fuzziness of a classifier 25 and it's generalization ability. Generalization indicates the capability of a classifier to correctly classify unseen samples, which is an important criterion to evaluate classifier's performance. An interesting aspect of classifier's generalization is the relationship between fuzziness of a classifier and its prediction accuracy. Extensive studies have been conducted to investigate the issue of generalization. Re-sampling methods 30 [5, 36] [38, 44, 24, 31] are important in the study of generalization. Among others, capability of online learning prototypes on instances, approaching from dependent data sources [1][43], estimation of generalization error bounds [9] [41] to deal with the problem of over-fitting, assessment of dependencies founded on experimental data, technique to maximize the boundary between the training samples and decision border [7] [4, 29, 8] etc. are also found to be important in connection with the study of generalization.

This paper investigates that, for a given trained classifier we get the output as a membership vector, 35 using that we can calculate the fuzziness of each input sample. Then input samples are sorted out into three classes, for example, low, medium and high fuzziness samples according to the fuzziness quantity of input samples.

Our main contributions in this paper are as follows:
(1) It is experimentally shown that if we add the low fuzziness instances from the test dataset to the original training dataset then its training accuracy can be improved.
(2) The phenomenon pointed out in (1) is theoretically explained based on the theory of learning from noisy data.
(3) It is found that, regarding the phenomena in (1) and (2), the initial accuracy of the base classifier has a significant impact on the improvement in accuracy.
(4) From rigorous experiments, it is observed that the maximum improvement of accuracy of the classifier

is attained when the initial accuracy is between 70% and 80%.

This paper is arranged as follows. Section 1 is an introduction. Section 2 gives an overview of semi-supervised learning. Section 3 introduces our proposed fuzziness based semi-supervised learning technique. Section 4 explains the sensitivity of initial accuracy of a classifier. Section 5 experimentally analyses this technique and its performance. Finally, section 6 concludes the paper.

2. An overview of semi-supervised learning

The primary idea of semi-supervised learning was first introduced by H. J. Scudder [31] [27, 48, 2] in 1965. Later, many researchers worked on it, consequently it has become one of the most prominent research fields of machine learning, pattern recognition and data mining. Semi-supervised learning is something in between supervised and unsupervised learning. It is considered as a combination of supervised and unsupervised learning technique. We know that, in supervised learning settings only labeled data are required to build a model. On the other hand, in unsupervised learning technique only unlabeled data are used. In contrast to both supervised and unsupervised learning, semi-supervised learning uses labeled data as well as unlabeled data to construct an improved classifier.

Let the labeled training dataset and unlabeled dataset be $(x_1, y_1) = \{x_{1:l}, y_{1:l}\}$ and $x_u = \{x_{l+1:n}\}$ respectively where, x_l is the l^{th} instance and y_l is its corresponding label. Then the classifier function g is defined as $g : x \rightarrow y$. This classifier is used to classify unlabeled data to label them. Then the model is retrained with newly labeled data along with previously labeled data to improve its accuracy.

The basic concept of semi-supervised learning is well explained by the following figure 1:

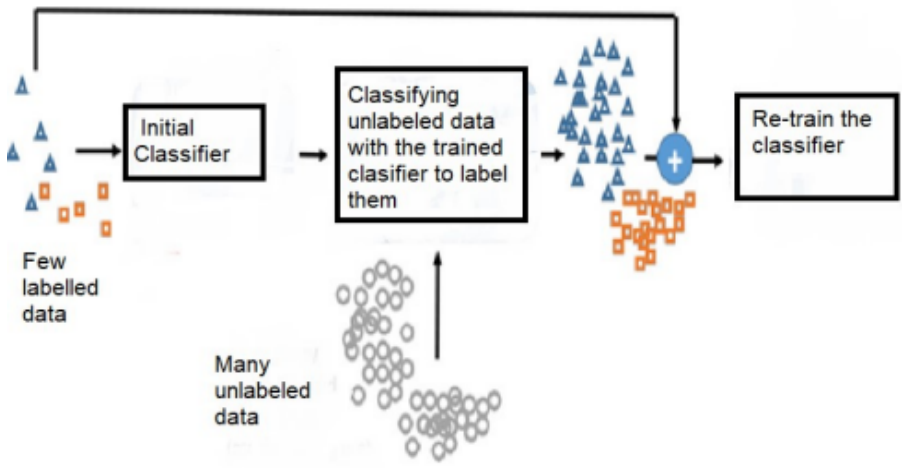


Figure 1: Basic principles of semi-supervised learning

In semi-supervised learning, both labeled and unlabeled data are used, which is mostly useful where unlabeled data are abandoned. For example, documents can be downloaded from the web,

images can be obtained from satellites and surveillance cameras and voice can be gathered from broadcast etc., each of these data types is typically unlabeled. Generally, obtaining labeled data is difficult, expensive and time consuming. Expert human is required to label or deal with it which adds some extra overhead to the processing of the data. On the contrary to labeled data, unlabeled data is most common, inexpensive and require almost no expert skill to process it. Semi-supervised learning uses mostly unlabeled data along with small volume of labeled data to build a better classifier. If we look at the history of machine learning particularly semi-supervised learning, the following methods are found to be important to the researchers, such as self-training, co-training and multi-view models, EM with generative mixture models, graph-based methods and transductive SVM(TSVM).

2.1. Self-training

Of all the semi-supervised learning algorithms, self-training is considered as the oldest one, which is one of the most frequently used techniques for semi-supervised learning. It is a form of learning which is designed to 80 train a classifier along with small volume of labeled data, thus occasionally called weakly-supervised learning. Then the unlabeled data are classified using the classifier and added to the training data. Subsequently the classifier is retrained using the newly added data and the process is continued as long as certain conditions are fulfilled. The self-training algorithm can be summarized as follows:

1. Train the predictor, g from labeled training dataset (x_1, y_1).
2. Predict on unlabeled data, $(x \; \epsilon \; x_u)$.
3. Add (x, g(x)) into labeled data.
4. Iterate until prediction accuracy of the initial predictor is satisfied.

Yarowsky [39] [34, 12] used the self-training based semi-supervised learning algorithm to figure out some natural language processing (NLP) tasks. Riloff et al.[28] [6, 36, 47] also used self-training algorithm to find out subjective nouns from a document which is an important task of NLP.

2.2 Co-training and multi-view learning

Co-training is another important type of semi-supervised learning technique which is considered as one of the special cases of multi-view learning. It is useful when two different views of the data are used to construct a pair of classifiers with an assumption that these two views are independent but sufficient for learning the 95 class targets. For example, one can build a classifier to classify web-pages based on the URL features. At the same time, another classifier can be built based on the text features of the web-page. Co-training algorithm follows the following steps:

1. Train two predictors $g^{(1)}$ and $g^{(2)}$ respectively from labeled data $(x_l^{(1)}, y_l)$ and $(x_1^{(2)}, y_1)$.

2. Predict unlabeled samples, $x \; \epsilon \; x_u$ with $g^{(1)}$ and $g^{(2)}$ distinctly.

3. Add $g^{(1)}$'s most assured samples $(x, g^{(1)}(x))$ to $g^{(2)}$'s labeled data and $g^{(2)}$'s most assured samples $(x, g^{(2)}(x))$ to $g^{(1)}$'s labeled data.

4. Iterate until prediction accuracy of both of the initial predictors is satisfied.

Maeireizo et al. [21] [26] classified dialogues as emotional or non-emotional using the co-training algorithm. Zhou et al. [47] [23, 18, 32] recommended tri-training where three classifiers were used.

Multiview learning is the natural extension of co-training. It has been widely used in semi-supervised learning. Multiview learning was successfully used in semi-supervised regression by Sindhwani et al. [33] [20].

Zhao Zhang et al. [42] [35] proposed a robust inductive semi-supervised label prediction model which is different from traditional label propagation algorithm that perform weight learning before label propagation. Nie et al. [25] [6, 11, 14] proposed auto-weighted multi-view learning for image clustering and semi-supervised classification. Zhang et al. [43] [10] proposed a special label propagation technique for semi-supervised classification. The main idea of co-training is that these two models are complementary to one another, moreover, helpful to correct them because each of the models is likely to make different mistakes. In general, this process is repeated until some convergence criteria are met.

2.3. Generative Model

The generative model is probably the most widely used semi-supervised learning technique. It considers a model which is based on conditional probability, i.e., $p(\frac{x}{y}) = \frac{p(x,y)}{p(y)}$ where $p(\frac{x}{y})$ follows Gaussian distribution [48]. To calculate $p(\frac{x}{y})$, here $p(x)$ plays an important role. Nigam et al. [26] [45, 37] demonstrated text classification from the labeled and unlabeled documents with the help of EM algorithm. EM is an iterative method which is used to estimate the maximum likelihood of parameters in a statistical model. The same algorithm was used by Baluja [4] [30] based on the EM algorithm along with both unlabeled and labeled data for probabilistic modeling of face recognition. Maximum entropy principle was used by Fujino et al. [14] [21] to extend the generative mixture model by introducing the phrases bias correction and discriminative training.

2.4. Graph-based Methods

The graph-based semi-supervised learning technique has been an active research area for the last couple of years [23] [33]. This technique assumes that there is a graph $G = V, E$ where the set of vertices V represent the labeled and unlabeled training samples, and the undirected edges E represent the connection between instances i, j with weight $w_{i,j}$ which represents the proximity of x_i and x_j . Large wi, j suggests that a preference for the predictions of $g(x_i)$ and $g(x_j)$ to be very close [48] [49]. Zhou et al. [46] used the directed graph to learn from mixed types of data, where they considered both the structure and direction of the graph to classify unlabeled vertices with the help of labeled vertices.Behpour et al. [6] proposed adversarial robust cuts for semi-supervised and multi-label classification. Zhang et al. proposed graph based semi-supervised learning using label propagation [44] [3]

2.5. Transductive SVM(TSVM)

Transductive reasoning refers to the task of inference from observed and specific cases. The term transduction was first by Vapnik in the 1990s. Transductive reasoning deals with the only specific

type of problems while inductive reasoning deals with the general type of problems. Transductive reasoning cannot handle unseen data but inductive reasoning can. However, transductive support vector machine (TSVM) is an extension of SVM.Ding et al.[12] conducted a research to give an overview on semi-supervised support vector machine. In SVM, only labeled data are used to find a maximum border hyperplane that separates two classes of data while in TSVM unlabeled data are also used for the same purpose [48]. The purpose is to find the labeling of unlabeled data so that a hyperplane has the maximum margin on both the original labeled data and the newly labeled data.

A very well posed question is that which semi-supervised learning technique should one use to solve her learning problem? Actually, there is no straightforward answer to this question because semi-supervised learning methods are based on some strong model assumptions. So, one can use any one of the techniques whose assumptions best matches with the given problem structure.

3. Proposed fuziness based semi-supervised learning

In this section we first discuss the concept of fuzziness along with its properties and introduce the base classifier. Then we briefly describe two base classifiers namely non-iterative single hidden layer feed forward neural network and back-propagation based neural network to illustrate our proposed fuzziness based semi supervised learning algorithm.

3.1. Fuzziness and its properties

The term fuzziness was proposed by Lotfi A. Zadeh in 1965 [41] [9] where he described fuzziness as an inexactness existing in an ill-defined event. Fuzziness is associated with the famous fuzzy set theory. Zadeh tried to relate the probability measure of an event with fuzzy events and suggested to use entropy to describe uncertainty connected with a fuzzy event. Fuzziness was described by Luca and Termini [20] [16] as an uncertainty associated with fuzzy sets and they used non-probabilistic entropy to measure fuzziness. They proposed three properties regarding fuzziness.

Property 1: The degree of fuzziness is at its maximum when all the members of a fuzzy set have equal value.
Property 2: The degree of fuzziness is at its minimum when all the members of a fuzzy set are equal to either 0 or 1.
Property 3: Fuzziness degree lies between its maximum and minimum values when the members of the fuzzy set have the values which is neither 0 nor 1 and even do not have equal values.

Sanchez and Trillas [30] measured the fuzziness under different uses of fuzzy sets using the function

$P \rightarrow [0,1]^X$ that satisfies the following axioms:

(i) $P(\theta) = 0$ if and only if θ is a crisp set.
(ii) $P(\theta)$ gets its maximum value on the following condition $\theta(x) = 0.5 \forall_x \in X$.
(iii) if $\theta \leq \sigma$ then $P(\theta) \geq P(\sigma)$.

(iv) $P(\theta) = P(\theta)$, where $\theta(x) = 1 - \theta(x)$ for $\forall_x \epsilon X$.

(v) $\mathrm{P}(\theta \cup \sigma) + P(\theta \cap \sigma) = P(\theta) + P(\sigma)$.

Considering the third axiom, the following inequality is proposed by Luca and Termini [20]
$\theta \leq \sigma \leftrightarrow min(0.5, \theta(x)) \geq min(0.5, \sigma(x))$ & $max(0.5, \theta(x)) \leq max(0.5, \sigma(x))$

Definition 1: Let A $= \theta_1, \theta_2, , \theta_n$ be a fuzzy set. Luca and Termini [20] defined the fuzziness of A as follows:

$$P(A) = -\frac{1}{n} \sum_{i=1}^{n} \theta_i \ log\theta_i + (1 - \theta_i) \ log(1 - \theta_i) \tag{1}$$

Equation (1) satisfies axioms (i-v).

3.2. Base classifier

In general, the term base classifier refers to the basic element of a multiple classifier system. As far as semi-supervised learning is concerned, base classifier means the initial classifier that uses labeled data to build a classifier, later, which is used to label huge amount of unlabeled data.

There are many base classifiers that give output as the form of a fuzzy vector, such classifier includes back-propagation based Neural Networks, Support Vector Machines, Fuzzy decision trees, non-iterative single layer feed-forward neural networks etc. Each component of the fuzzy vector output corresponds to the membership degree of the testing object belonging to a class. Sometimes it is needed to transform the initial output of the classifier to a form of a fuzzy vector if the elements of the initial output are not between 0 and 1. Two classifiers have been selected for the experimental demonstration. They are Extreme Learning Machine (ELM) which is a non-iterative single hidden layer feed-forward neural network and back-propagation based neural networks.

The output of a classifier can be associated with fuzziness. There are many classifiers [18, 40] that have output in the form of a fuzzy vector. Each component of this fuzzy vector represents the degree of membership of testing instance belonging to a certain class. Let the training samples be x_i where $i = 1, 2, ..., N$ and there are C number of classes. The output of the classifier represents the membership degree of each instance to C classes. A membership matrix $M = (\theta_{ij})_{C*N}$ can be used to describe the partition among samples where $\theta_{ij} = \theta_i(x_j)$ represents the membership of jth sample belonging to the ith class. The elements in the membership matrix M have to follow the following criteria.

$$\sum_{i=1}^{C} \theta_{ij} = 1; 0 < \sum_{i=1}^{N} \theta_{ij} < N, \theta_{ij} \in [0, 1] \tag{2}$$

After the training process of a classifier we get the membership matrix M. The trained classifier provides an output vector represented in the form of a fuzzy set $\theta_j = (\theta_{1j}, \theta_{2j}, \ldots, \theta_{Cj})^T$. Now the fuzziness of θ_j is obtained by

$$P(\theta_j) = -\frac{1}{C} \sum_{i=1}^{C} \theta_{ij} \ log\theta_{ij} + (1 - \theta_{ij}) \ log(1 - \theta_{ij}) \tag{3}$$

3.3. Fuzziness based divide-and-conquer technique

Wang et al. [37] [1] suggested a divide-and-conquer based new technique to handle fuzziness. According to them, the given dataset is divided into the training set and the testing set. The testing samples are sorted out into three groups i.e., low, medium and high fuzziness samples, based on the degree of fuzziness. Then samples with low fuzziness are added to the training set. Now the new training set is retrained. Based on the idea of Wang et al., here we propose our new algorithm for fuzziness based semi-supervised learning (Algorithm 1). Although many classification algorithms with fuzzy vector output can be used along with

Algorithm 1 Fuzziness based semi-supervised learning algorithm

Input: Dataset, # hidden layer Z, # node in each layer Q.

Output: Maximum improvement of accuracy over the initial accuracy, corresponding initial accuracy.

1: Randomly partition the dataset into a training dataset $\mathbf{X}tr$ and a testing dataset $\mathbf{X}te$.

2: $x \leftarrow 1$

3: $k \leftarrow 1$

4: **while** $x <= Z$ **do**

5: $n \leftarrow 1$

6: **while** $n <= Q$ **do**

7: Train the classifier $\mathbf{C}$ according to a training algorithm.

8: Get the training accuracy tr_{accB}

9: Get the fuzzy vector $A_i = \{\theta_1, \theta_2, ..., \theta_n\}$ for each sample in testing set by classifier $\mathbf{C}$.

10: Calculate the fuzziness $P(A_i)$ of each sample in testing set by:$P(A_i) = -\frac{1}{n}\sum_{i=1}^{n} \theta_i \log \theta_i + (1 - \theta_i) \log(1 - \theta_i)$

11: Sort the samples by the fuzziness $P(A_i)$, and group testing set $\mathbf{X}te$ into three fractions: $\mathbf{X}telow$, $\mathbf{X}temedium$ and $\mathbf{X}tehigh$.

12: Get new training set $\mathbf{X}trnew$ by adding the low-fuzziness samples $\mathbf{X}telow$ to the original training set $\mathbf{X}_{tr}$.

13: Retrain new classifier $\mathbf{C}new$ according to the given training algorithm with $\mathbf{X}_{tr}new$.

14: Again record the training accuracy traccA by classifier $\mathbf{C}new$ with $\mathbf{X}trnew$

15: Record $diff[k] = tr_{accA} - tr_{accB}$

16: $n = n + 1$

17: $k = k + 1$

18: **end while**

19: $x = x + 1$

20: **end while**

21: Find the maximum of $diff$ and the corresponding initial accuracy

our proposed algorithm. We have selected non-iterative single hidden layer feed-forward neural network and

back-propagation based neural network to demonstrate the effectiveness of our algorithm.

3.3.1. Non-iterative single hidden layer feed-forward neural network

In this subsection, non-iterative single hidden layer feed-forward neural network also called extreme learning machine (ELM) [11, 16, 19] is briefly described.In ELM, weights between input and hidden layers are chosen arbitrarily and weights between hidden and output layer are determined analytically. For a given training set $\xi = \{(x_i, t_i) \mid x_i \in \mathbb{R}^n, t_i \in \mathbb{R}^m, i = 1, 2, \ldots, N\}$ activation function $h(x)$ and number of hidden layer node Q, ELM algorithm follows the following steps [16].

Step 1: Arbitrarily allocate input weight w_i and bias b_i, where, $i = 1, 2, \ldots, Q$
Step 2: Compute the hidden layer output matrix H.
Step 3: Compute the output weight β where, $\beta = (H^T H)^{-1} H^T T$.

In our 1st experiment, suppose, Q is the number of hidden layer node, n and m are two integer numbers and m must be greater than n. Here the number of hidden layer node is carefully chosen. We initially assign n to Q and continue to change the number of hidden layer node and train the model so that the initial accuracy of the base classifier can be changed. During each iteration of the algorithm, we split the dataset into two parts namely the training set and the testing set respectively then train the model using non-iterative single hidden layer neural network, i.e., ELM algorithm. After that, we calculate the fuzziness of each sample and record the initial accuracy of the base classifier. At that time, based on the quantity of fuzziness of the testing samples, we categorize them into three groups i.e., low, medium and high fuzziness samples and add the samples with low fuzziness from the testing set to the original training set. Then we retrain the ELM on the new training set. We again calculate the training accuracy and record the difference in accuracy. If we continue to run the algorithm until Q < m then it can be easily observed that for which value of initial accuracy of base classifier maximum improvement in accuracy is obtained.

3.3.2. Back-propagation based neural network

In this sub-section, we briefly describe the back-propagation based neural network [2, 29] [7, 42] with vector output where each component of the vector are number between 0 and 1. This number indicates that to what extent an instance belongs to a specific class. Although there are many different architectures of neural networks, however, we consider multilayer feed-forward neural network for experimental purpose. In this architecture, there are one input layer, two or more hidden layers and one output layer. Let the training set be (x_l, y_l) here $x_l = (x_{l1}, x_{l2}, \ldots, x_{ln})$ is the input features and t_l is the desired output. Given input is (x_l) and the actual output is (y_l). For each training sample (x_l) the inputs are propagated forward through the network. Each unit in the hidden and output layers takes its net input and then uses an activation function to it. For each unit in the output layer the errors are propagated backwardly from output layer to the input layer. The process is continued until the following cost function is minimized [8]: $E = \frac{1}{2} \sum_{i=1}^{C} (t_l - y_l)^2$.

In our 2*nd* experiment, suppose, Z and Q are number of hidden layer and number of node in each layer respectively, x, y, n and m are four integer numbers, where y is greater than x and m is greater than n. In

this experiment, we initialize x to Z and n to Q and continue to change the value of Q and Z. During each iteration of the algorithm, we divide the dataset into the training set and the testing set and train the model using backpropagation based neural network. Then, we calculate the magnitude of fuzziness of each sample and record the initial accuracy of the base classifier. The testing samples are categorized into three groups namely, low, medium and high fuzziness groups. Then we add low fuzziness testing samples to the original training samples and retrain the model with new samples and then calculate the accuracy and find the difference between new accuracy and initial accuracy. The algorithm is run until $Q < m$ and $Z < y$. From this experiment, we can easily notice that, for which value of initial accuracy of base classifier maximum improvement is achieved.

4. Sensitivity of initial accuracy

In the study of machine learning algorithms sensitivity analysis plays an important role to clearly understand the relationship between the input and the output of a learning algorithm. In general, sensitivity analysis is the study of how the output of an algorithm can be significantly dependent on different input parameters. The procedure of recalculating outputs under alternative input parameters to estimate the impact of input parameters can be useful for a range of purposes, such as, investigating the robustness of the results of a learning algorithm, understanding the relationships between input and output variables in a learning algorithm, simplification of algorithms through fixing the inputs that have less effect or even no effect on the output, reducing uncertainty by the identification of algorithm inputs that cause uncertainty in the output etc.

4.1. Main idea of sensitivity analysis

In semi-supervised setting, when we use initial classifier to classify huge amount of unlabeled data several events may turn out. Such as, the accuracy of initial classifier may be very low, or medium or even high. Here, we will briefly discuss about every phenomenon.

Firstly, when the accuracy of the initial classifier is very low, for instance about 50%, then if we use that classifier to predict unlabeled data, the newly predicted labels may have a lot of noises. We consider those noisy samples as bad samples. And when we add those so-called bad samples with wrongly-predicted labels to original training data and retrain the model, the accuracy of the classifier is surely not improved.

Secondly, when the accuracy of the initial classifier is medium, for example, around 75% and if we use that classifier to classify huge amount of unlabeled data then that classifier works well, that is, it gives fairly proper prediction results for unlabeled samples. In this case, newly predicted labels have a few noises and these samples are considered as quality samples. If those supposed quality samples with a few wrongly predicted labels are added to the original training set and retrain the model then the accuracy may be substantially improved and these noisy data have a very less effect on the learning performance.

Finally, when the initial classifier's accuracy is high, for example around 95% and if we use that classifier to predict unlabeled data it may generate a very few wrongly-predicted labels, that is, a very few noises. But 265 in this case, if we add these new samples with their predicted labels to the original training data to retrain a new model, we are not sure that whether it significantly improves the classification accuracy or not. It is reasonable to think that, the classifier's accuracy will less likely be improved because the initial accuracy is already very high.

From our main idea we can conclude that, there is a trade-off between the initial accuracy of a classifier 270 and the amount of improvement over the initial accuracy. Although many researchers think that noisy samples have a very bad effect on the learning performance but we observe a phenomenon where noisy samples have a less effect on the learning performance. In the next section, we are going to discuss about a mathematical model to support our main idea to learning from noisy data.

4.2. Mathematical model to learn from noisy data

In semi-supervised learning, when we add the low fuzziness samples from the test dataset to the original training dataset, then, in this new training dataset there are some noises. This noises usually have some bad effect on the learning performance. But according to Lin Gui et al. [15] under some circumstances, noisy samples have a very less effect on the learning performance.

To describe the mathematical model for learning from noisy data we first introduce and then estimate 280 the rate of class noise. In fact, class noise rate is usually referred to as an independent likelihood of the class labels being changed on the entire training dataset. Suppose $(x_i, y_i) \in$ D and $(x_i, y_i') \in$ D$'$ are samples from clean distribution and noisy distribution respectively. Here, the label y_i' is possibly different from actual label y_i. After a rigorous proof, Lin Gui et al. derived the following inequality to approximate the class noise rate, (Nc) for the noisy sample (x_i, y_i'):

$$N_c(x_i) \geq 1 - 0.5 \times exp(-\frac{\sum_{j=1}^{k} s_i j.d_i j)^4}{2(||s_i||_1||s_i||_2)^2}) \tag{4}$$

In the inequality (4), d_{ij} is the dissimilarity between the ith sample and its j near most labels, here, j = 1, 2, ..., k and d_{ij} is defined by the following sign function:

$$d_{ij} = \left\{ \begin{array}{ll} 1, & if\ y_i' \neq y_j' \\ -1, & if\ y_i' = y_j' \end{array} \right\}$$

s_{ij} is the similarity between the i^{th} sample and its j near most labels, here, j = 1, 2, ..., k. s_{ij} is normalized in the range [0, 1].

k is the number of nearest neighbors, $||s_i||_1$ and $||s_i||_2$ are l_1 and l_2 norms respectively.

Suppose L(g(x), y) and L$'$ (g(x), y$'$) are loss functions for the clean distribution, D and noisy distribution, D$'$ respectively. When we use the estimated rate of class noise, $N_c(x_i)$ for every single instance and change

the loss function for the noisy distribution with the perceived labels then three kinds of risks may come out and these risks can be stated as follows:

Definition 2.1: The Expectation of L-risk for the clean distribution D is given by: $\mathrm{R}_{L,D}(\mathrm{g}) = \mathrm{E}_{(x,y)\sim D}(\mathrm{L}(\mathrm{g}(\mathrm{x}), \mathrm{y}))$.

Definition 2.2: The Expectation of L′-risk for the noisy distribution D′ is given by: $\mathrm{R}_{L',D'}(\mathrm{g}) = \mathrm{E}_{(x,y')\sim D'}(L'(g(x), y'))$.

Definition 2.3: The Empirical L′-risk is stated on the training dataset as follows: $\mathrm{R}_{L'}(\mathrm{g}) = \frac{1}{n}\sum_{i=1}^{n} L'(g(x_i), y_i')$.

Once we know the class noise rate, it can be used to approximate the loss function for noisy distribution to the loss function for clean distribution. Loss function for noisy distribution is defined as follows:

$$L'(g(x_i), y_i') = \frac{(1 - N_c(x_i)) * L(g(x_i, y_i')) - N_c(x_i) * L(g(x_i, -y_i'))}{1 - 2 * \frac{\sum_{i=1}^{n} N_c(x_i)}{n}} \tag{5}$$

In the right hand side of the equation (5), the numerator has two components. The first component stats the loss function with their observed labels, which is multiplied by the probabilities of their label correctness. The next component indicates the penalty for the loss function along with their inverted labels, which is multiplied by the rate of class noise. The denominator is dependent on the average rate of class noise, which guarantees that the value of the loss function of both noisy and clean training data is almost same.

From the preceding discussion, some issues become apparent. Such as,

(i) When the training data is noisy and its size is very big then the empirical L′-risk of training data converges to the Expectation of L′-risk, $\mathrm{R}_{L',D'}(\mathrm{g})$.

(ii) If the rate of class noise is estimated correctly then the Expectation of L′-risk under noisy distribution converges to the Expectation of L-risk under the clean distribution i.e., $E_{y'}[(L'(g(x), y')]$ = L(g(x), y).

From this discussion we can intuitively conclude that, under the above mentioned circumstances noises has no effect on the learning performance because first the empirical L′-risk, $\mathrm{R}_{L'}(\mathrm{g})$ converges to the Expectation of L′-risk, $\mathrm{R}_{L',D'}(\mathrm{g})$ under the noisy distribution, and eventually $\mathrm{R}_{L',D'}(\mathrm{g})$ converges to the Expectation of L-risk, $\mathrm{R}_{L,D}(\mathrm{g})$ under clean distribution of training data.

5. Experimental results and discussion

To experimentally validate the performance of our proposed approach, twelve standard datasets were obtained from UCI machine learning repository [13]. Before using those datasets, we normalized them to ensure the uniformity of the output. Apart from the 12 UCI datasets we used 2 real-world databases namely YALE and ORL[1] face databases to evaluate our proposed model. Description of the datasets is given in Table 1. YALE database contains images of 15 persons with 11 different types of images for each person. The images are different in various lighting conditions, facial expressions (such as normal, happy, sad etc.)

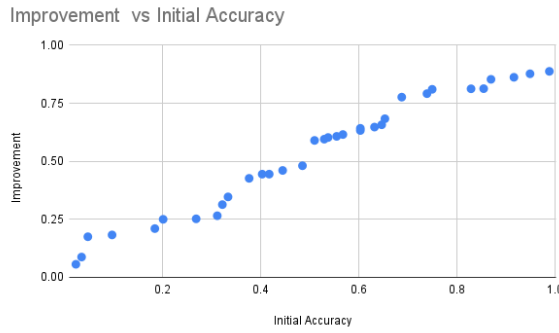
and each image size is 32 × 32 pixels with 256 gray levels per pixel. ORL database contains images of 40 persons with 10 different types of images for each person. The images are different in expressions (open or close eyes, smiling or non-smiling) and various facial expressions (with glasses or without glasses) and each image size is 32 × 32 pixels with 256 gray levels. Before actual training, the gray level of each image is normalized to the range of [0, 1]. We use PCA [49] to reduce the dimension of original images. After applying PCA we consider the first 40 principal components of both YALE and ORL datasets to make the processing of these datasets easier. We randomly choose different number of images from each person for training and leave others for testing. As we change the number of training data it leads us to have different initial training accuracy for different models. Three different experimental setups were built to conduct and demonstrate the effectiveness of our proposed algorithm.In the first experiment, we used non-iterative single hidden layer feed-forward neural network i.e., ELM as a classifier in our proposed fuzziness based semi-supervised technique.The experimental results are represented in figure 2 (a - n). Here, the x-axis represents the initial accuracy of base classifier and the y-axis represents the improvements over the initial accuracy.

Table 1: Dataset description

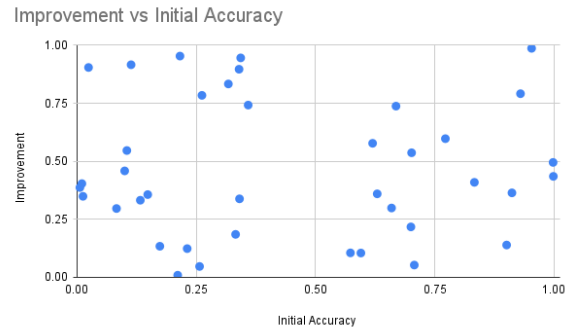
Dataset	**# Instances**	**# Features**	**# Classes**
Blood Transfusion Service Center Dataset	749	4	3
Indian Liver Patient Dataset (ILPD)	582	10	2
Phishing Dataset	1354	9	3
Pima-indians-diabetes	769	8	2
HIGGS-30000 dataset	29841	28	2
letter-recognition	20000	16	26
magic04 dataset	19020	10	2
waveform	5000	21	3
vehicle	846	18	4
Ecoli	336	7	8
sonar	208	60	2
Parkinson	195	22	2
YALE Dataset	165	1024	15
ORL Dataset	400	1024	40

The most striking result to emerge from the datasets is that, the correlation between the initial accuracy of a base classifier and improvement of classifier's performance is worth mentioning, because, for a specific base classifier, experiment showed that changes in the initial accuracy of base classifier significantly changed the performance of the classifier. Particularly, when low fuzziness instances from test datasets are added to the original training data, classifier's performance was improved. We further demonstrate that the improvement

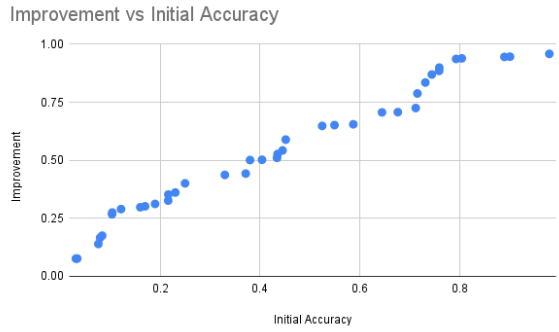
of classifier's performance is largely dependent on the initial accuracy of the base classifier and experiments also reveal that the improvement of accuracy reaches its maximum when the initial accuracy of base classifier is nearly 75%. Figures 2(a - n) describe that, when the initial training accuracy of a classifier is around 75% then the improvement of training accuracy is maximum. For example, in the Figure 2(a), we can see that when the initial accuracy is 0.770089286 then we got the maximum improvement of accuracy which is 0.03283 and all the datasets under this experiment exhibit almost the same results.



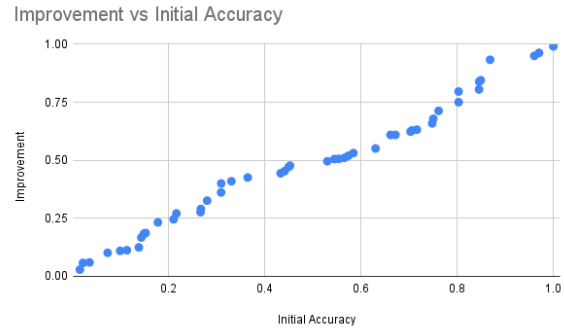
(a) Blood Transfusion Service Center Dataset



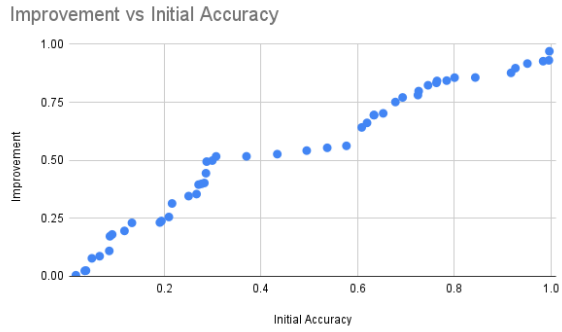
(b) Indian Liver Patient Dataset



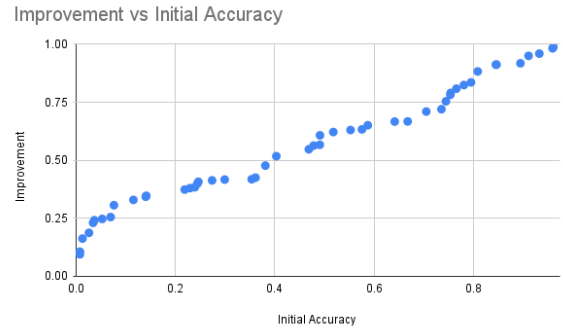
(c) Phishing Dataset



(d) Pima-indians-diabetes

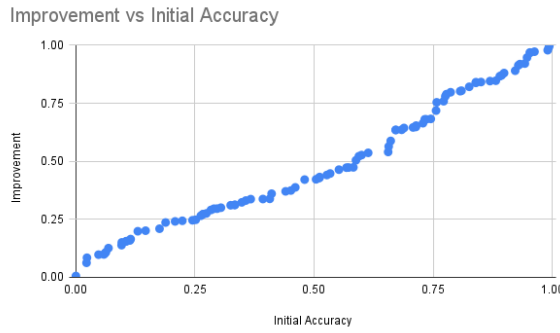


(e) HIGGS-30000 dataset

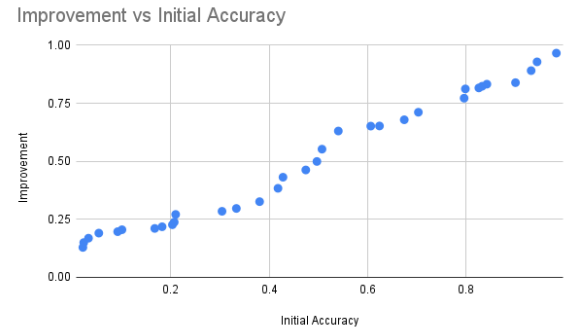


(f) Letter recognition

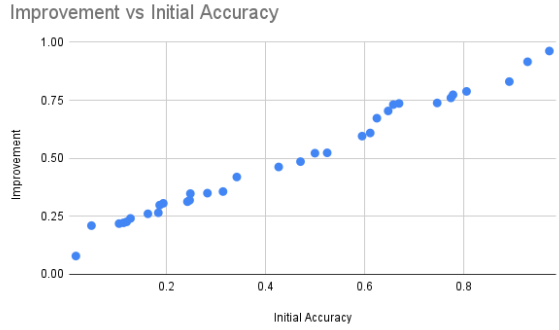
Figure 2: Results of 1st experimental setup (ELM used as base classifier)



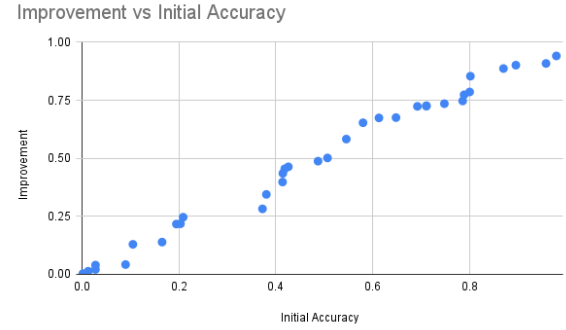
(g) magic04 dataset



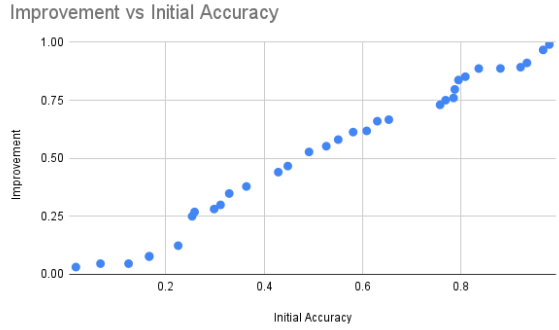
(h) waveform dataset



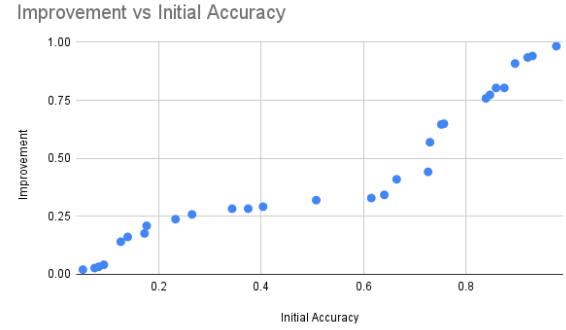
(i) Vehicle dataset



(j) Ecoli

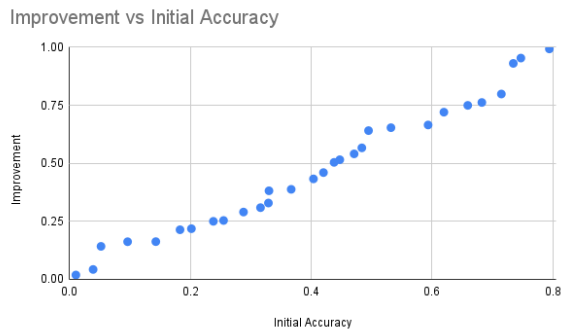


(k) Sonar dataset

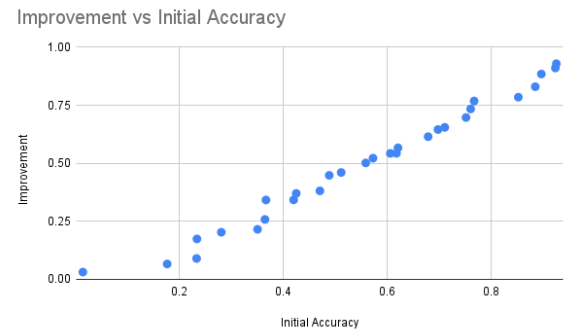


(l) Parkinson dataset

Figure 2: Results of 1st experimental setup (ELM used as base classifier) (cont.)



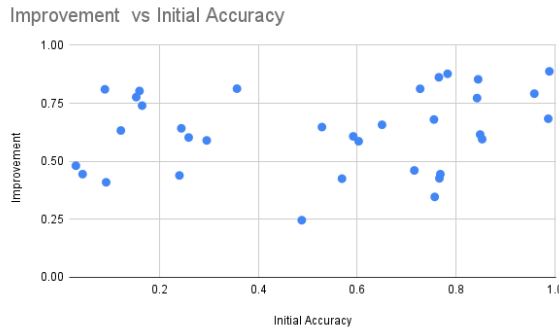
(m) YALE dataset



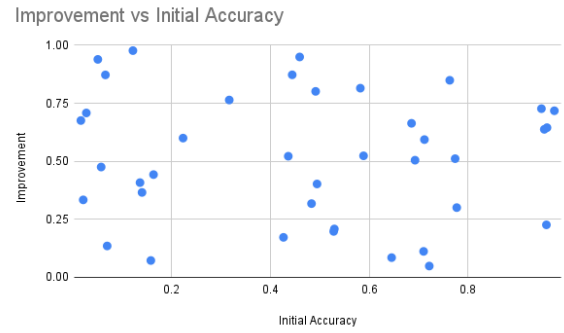
(n) ORL dataset

Figure 2: Results of 1st experimental setup (ELM used as base classifier) (cont.)

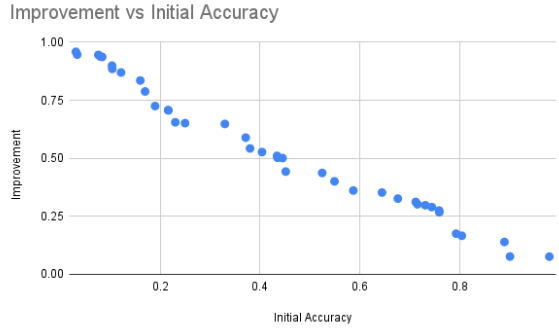
In the second experiment, we use backpropagation based neural network as a classifier in order to demonstrate the effectiveness of our proposed fuzziness based semi-supervised learning algorithm. The experimental results are depicted in figures 3(a − n), where, the x-axis denotes the initial accuracy of base classifier and the y-axis denotes the improvements over the initial accuracy. From the results, it is obvious that, if we calibrate the initial accuracy of the base classifier across a broad range (e.g., 50% to 100%) to find out the maximum improvement of classifier accuracy over the initial accuracy then we find it when the initial accuracy is around 75%. For example, in the Figure 3(b), we can see that when the initial accuracy is 0.753581662 then we got the maximum improvement of accuracy which is 0.042013 and all the datasets under this experiment show almost the same results.



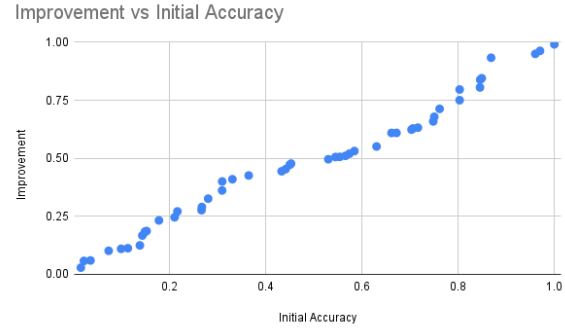
(a) Blood Transfusion Service Center Dataset



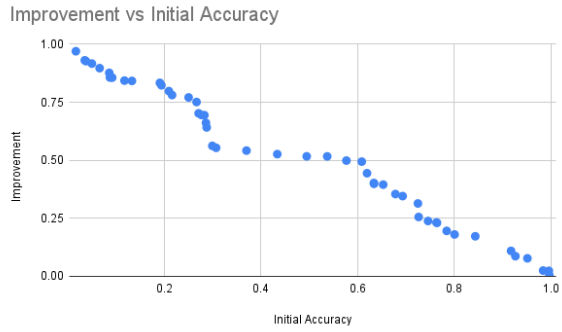
(b) Indian Liver Patient Dataset



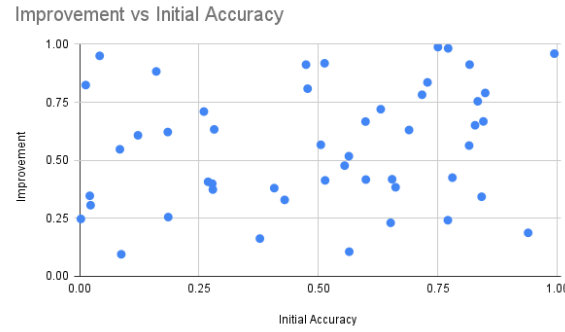
(c) Phishing Dataset



(d) Pima-indians-diabetes

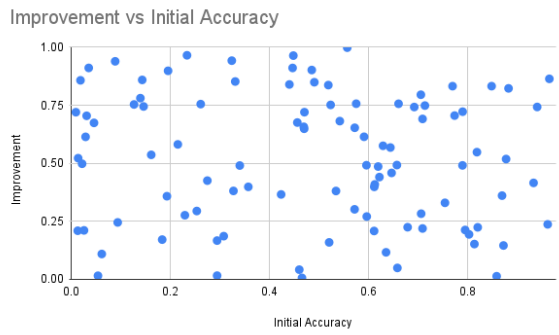


(e) HIGGS-30000 dataset

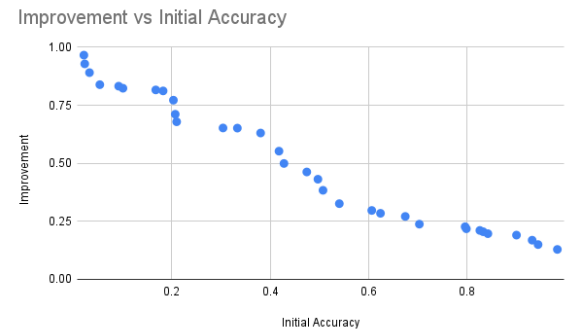


(f) Letter recognition

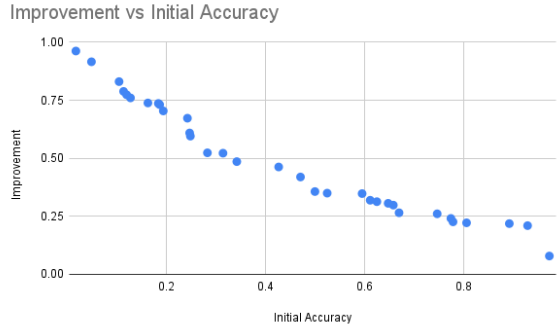
Figure 3: Results of 2nd experimental setup (NN used as base classifier)



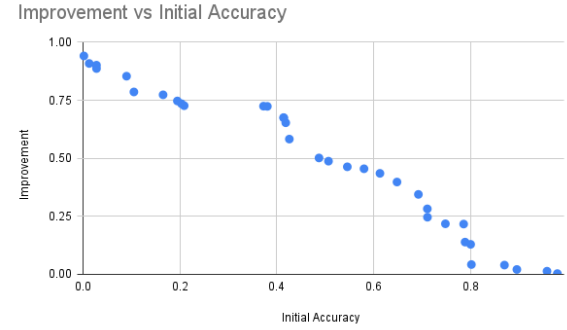
(g) magic04 dataset



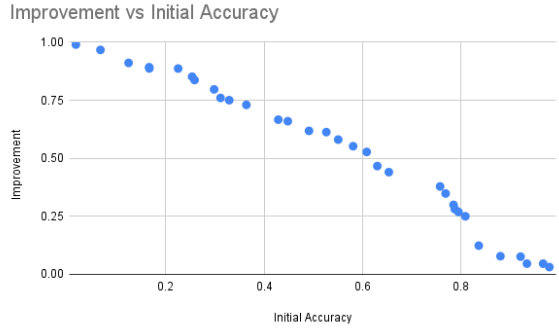
(h) waveform dataset



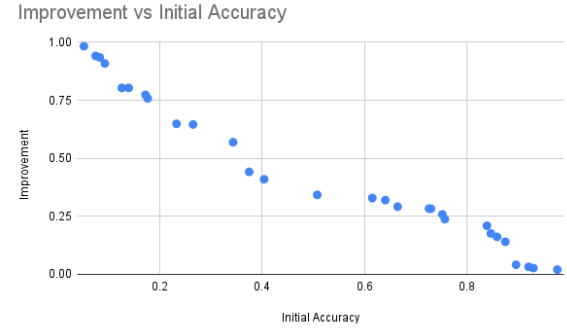
(i) Vehicle dataset



(j) Ecoli

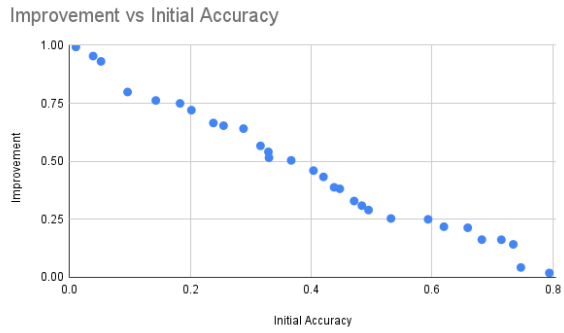


(k) Sonar dataset

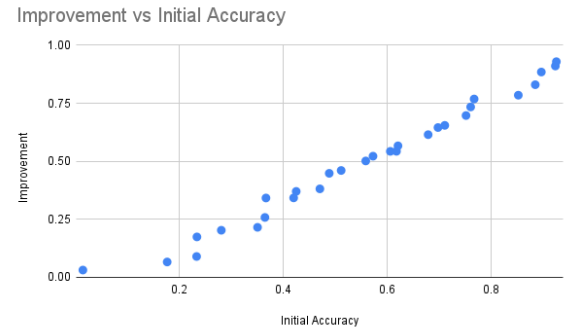


(l) Parkinson dataset

Figure 3: Results of 2nd experimental setup (NN used as base classifier) (cont.)



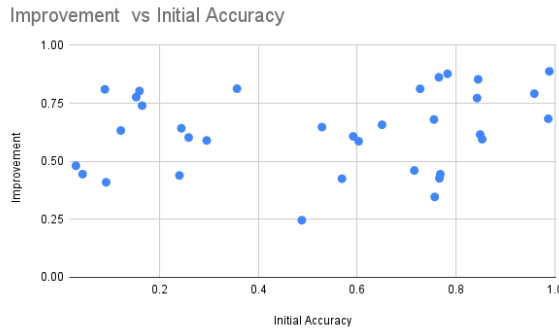
(m) YALE dataset



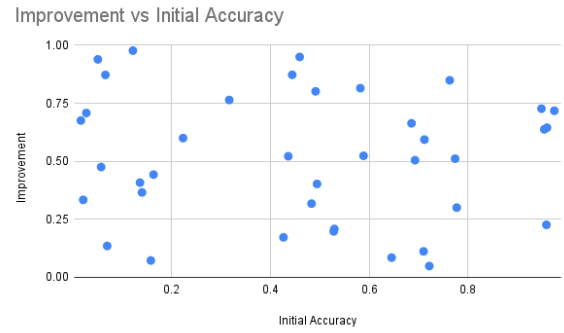
(n) ORL dataset

Figure 3: Results of 2nd experimental setup (NN used as base classifier) (cont.)

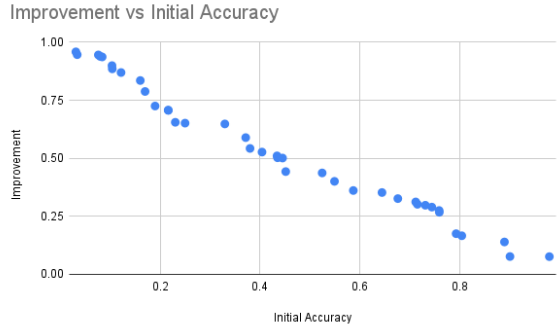
When there are a few labeled data and huge amount of unlabeled data we can use label propagation algorithm to give labels to unlabeled data. From the literature we found many works that tried to exploit the idea of label propagation algorithm. Among others Zhao et al. used label propagation in the following works [43, 44, 45]. Xiaojin et al. [39] showed that to train the label propagation model if the number of the labeled data is changed, accuracy is also changed accordingly. They demonstrated the more the number of labeled data is used the more the accuracy is improved. In our third experiment, where we use LP (Label Propagation)as initial classifier, during the training we changed the number of labeled data to calibrate the initial accuracy of the classifier so that we can observe for which initial accuracy maximum improvement of accuracy is obtained. The experimental results are represented in figure 4(a − n). Here, the x-axis represents the initial accuracy of the base classifier and the y-axis represents the improvements over the initial accuracy. For example, in the Figure 4(c), we can see that when the initial accuracy is 0.7061 then we got the maximum improvement of accuracy which is 0.07291 and all the datasets under this experiment exhibit almost the same results.



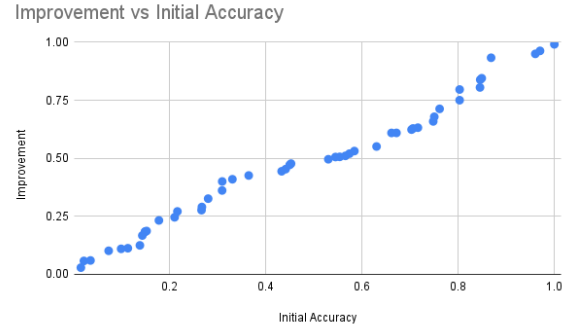
(a) Blood Transfusion Service Center Dataset



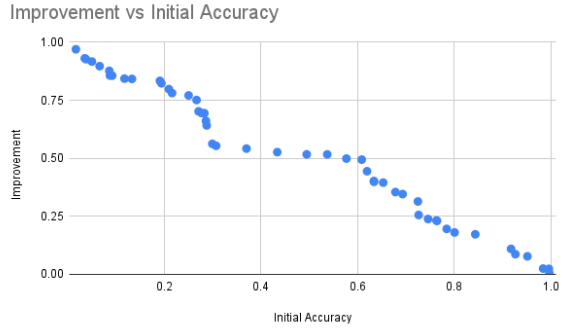
(b) Indian Liver Patient Dataset



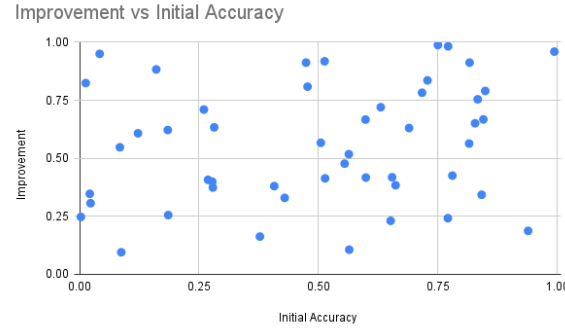
(c) Phishing Dataset



(d) Pima-indians-diabetes

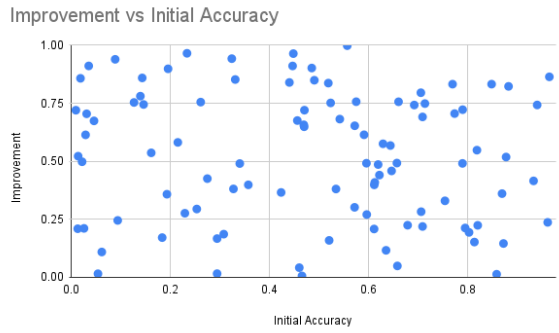


(e) HIGGS-30000 dataset

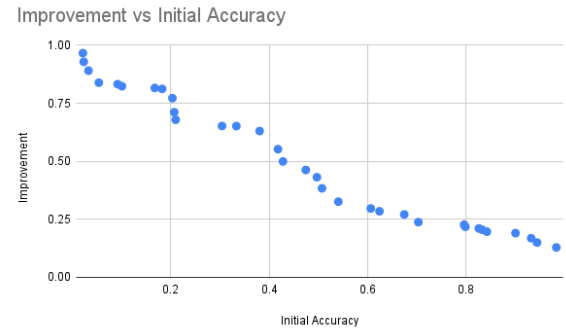


(f) Letter recognition

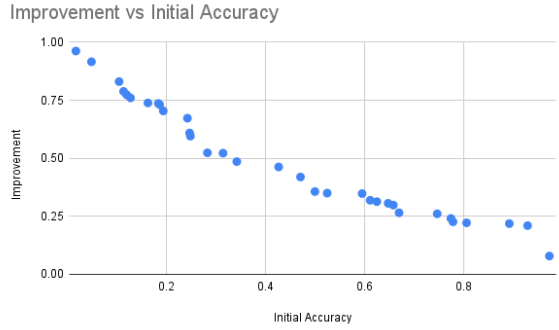
Figure 4: Results of 3rd experimental setup (LP used as base classifier)



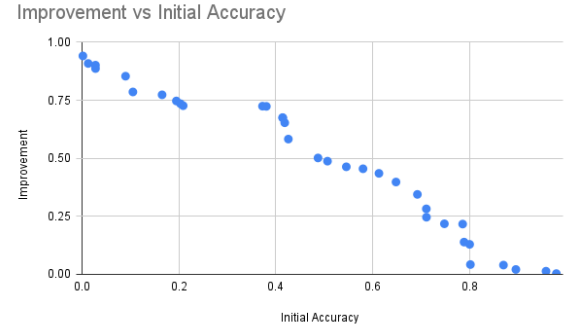
(g) magic04 dataset



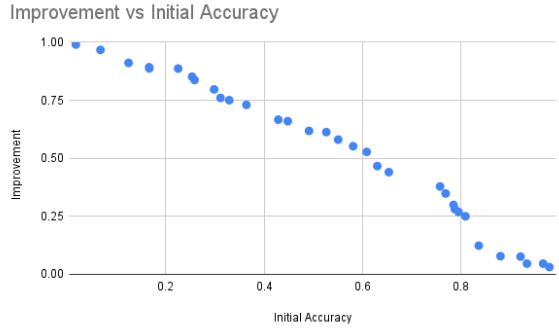
(h) waveform dataset



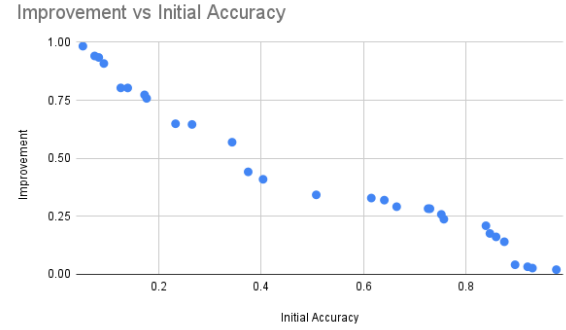
(i) Vehicle dataset



(j) Ecoli

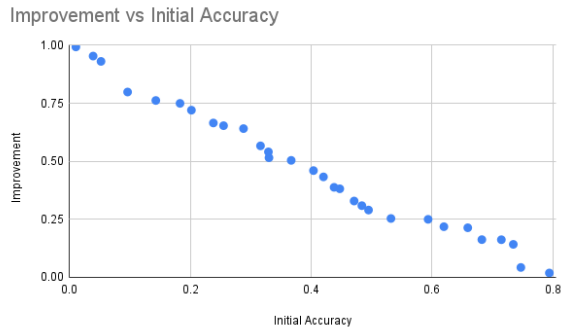


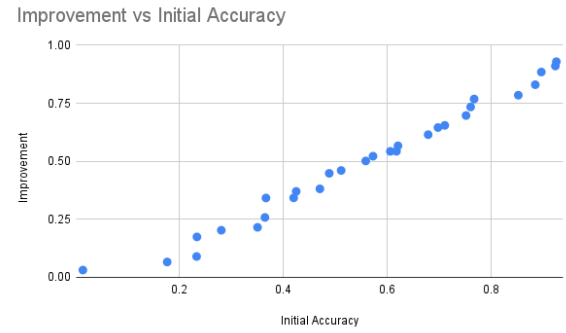
(k) Sonar dataset



(l) Parkinson dataset

Figure 4: Results of 3rd experimental setup (LP used as base classifier) (cont.)





(m) YALE dataset

(n) ORL dataset

Figure 4: Results of 3rd experimental setup (LP used as base classifier) (cont.)

Table 2, Table 3 and Table 4 represent the results of our proposed algorithm respectively for ELM, NN and LP used as initial classifiers. These tables report the maximum improvements of accuracy for every dataset when we add low fuzzy samples from testing dataset to training dataset. From the Tables 2−4 it is clear that maximum improvement is obtained when the initial accuracy is between 70% and 80%.

Table 2: ELM used as initial classifier

Dataset	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**	**Improvement of accuracy**
Blood Transfusion Service Center Dataset	0.770089286	0.80291971	0.03283
Indian Liver Patient Dataset	0.765721662	0.79859485	0.045013
Phishing Dataset	0.765721332	0.79818365	0.032462
Pima-indians-diabetes	0.776521739	0.80451957	0.027998
HIGGS-30000 dataset	0.690013405	0.70577686	0.015763
letter-recognition	0.703923077	0.72418966	0.020267
magic04 dataset	0.729111057	0.75085722	0.021746
waveform	0.78	0.7921628	0.012163
vehicle	0.788461538	0.80327869	0.014817
Ecoli	0.780597015	0.79310345	0.012506
sonar	0.765060241	0.8	0.03494
Parkinson	0.781282051	0.80248521	0.021203
YALE	0.776	0.7896628	0.013663
ORL	0.7625	0.83540462	0.072905

Table 3: NN used as initial classifier

Dataset	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**	**Improvement of accuracy**
Blood Transfusion Service Center Dataset	0.756785714	0.80291971	0.03283
Indian Liver Patient Dataset	0.753581662	0.79859485	0.045013
Phishing Dataset	0.774352651	0.79818365	0.032462
Pima-indians-diabetes	0.767391304	0.80451957	0.027998
HIGGS-30000 dataset	0.690013405	0.70577686	0.015763
letter-recognition	0.703923077	0.72418966	0.020267
magic04 dataset	0.729111057	0.75085722	0.021746
waveform	0.78	0.7921628	0.012163
vehicle	0.788461538	0.80327869	0.014817
Ecoli	0.780597015	0.79310345	0.012506
sonar	0.765060241	0.8	0.03494
Parkinson	0.737179487	0.80248521	0.021203
YALE	0.776	0.7896628	0.013663
ORL	0.7575	0.83540462	0.072905

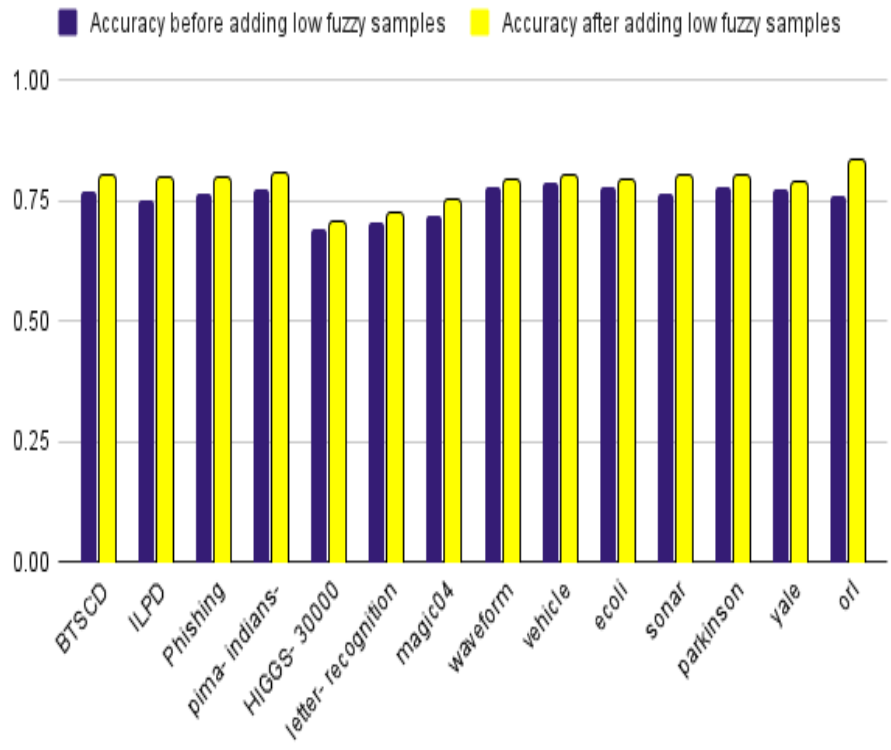
Table 4: LP used as initial classifier

Dataset	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**	**Improvement of accuracy**
Blood Transfusion Service Center Dataset	0.770089286	0.790123457	0.012531
Indian Liver Patient Dataset	0.765721662	0.79859485	0.045013
Phishing Dataset	0.765721332	0.79818365	0.032462
Pima-indians-diabetes	0.776521739	0.80451957	0.027998
HIGGS-30000 dataset	0.690013405	0.70577686	0.015763
letter-recognition	0.703923077	0.72418966	0.020267
magic04 dataset	0.729111057	0.75085722	0.021746
waveform	0.78	0.7921628	0.012163
vehicle	0.788461538	0.80327869	0.014817
Ecoli	0.780597015	0.79310345	0.012506
sonar	0.765060241	0.8	0.03494
Parkinson	0.781282051	0.80248521	0.021203
YALE	0.776	0.7896628	0.013663
ORL	0.703125	0.732080925	0.028956

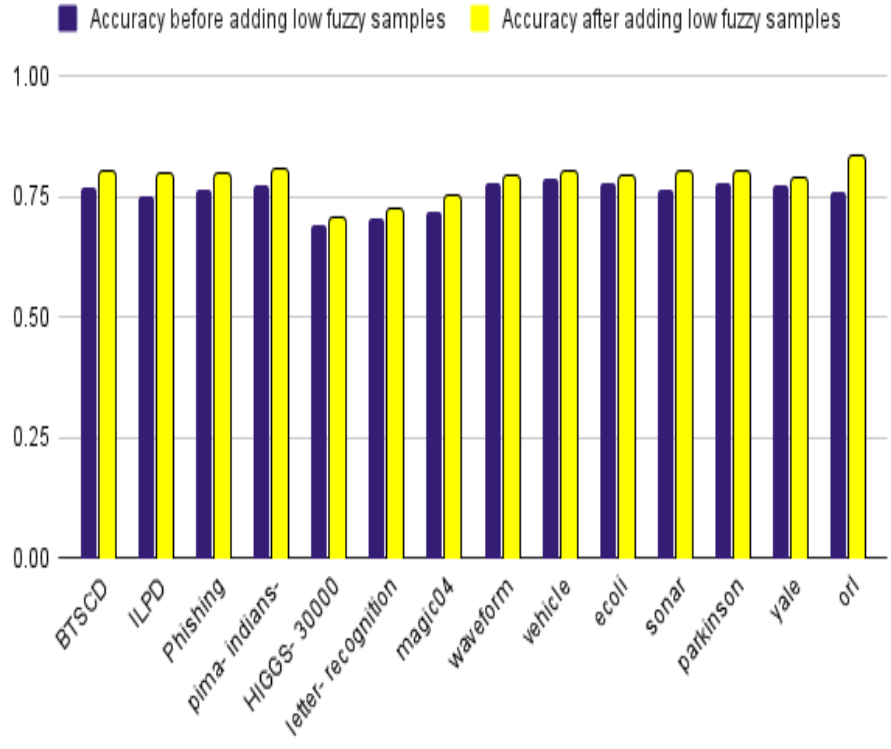
Table 5: Performance comparison of ELM, NN and LP as base classifier

	ELM as base Classifier		**NN as base Classifier**		**LP as base Classifier**	
Dataset	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**	**Accuracy before adding low fuzzy samples**	**Accuracy after adding low fuzzy samples**
Blood Transfusion Service Center Dataset	0.770089286	0.802919708	0.756785714	0.796569343	0.777591973	0.790123457
Indian Liver Patient Dataset	0.753581662	0.798594848	0.753581662	0.795594848	0.706008584	0.736633663
Phishing Dataset	0.765721332	0.798183653	0.774352651	0.813229062	0.7061	0.77901
pima-indians-diabetes	0.776521739	0.798183653	0.774352651	0.813229062	0.7061	0.77901
HIGGS-30000	0.690013405	0.798183653	0.774352651	0.813229062	0.7061	0.77901
letter-recognition	0.703923077	0.798183653	0.774352651	0.813229062	0.7061	0.77901
magic04 dataset	0.7203923077	0.798183653	0.774352651	0.813229062	0.7061	0.77901
waveform	0.78	0.798183653	0.774352651	0.813229062	0.7061	0.77901
vehicle	0.788461538	0.798183653	0.774352651	0.813229062	0.7061	0.77901
ecoli	0.780597015	0.798183653	0.774352651	0.813229062	0.7061	0.77901
sonar	0.765060241	0.798183653	0.774352651	0.813229062	0.7061	0.77901
parkinson	0.781282051	0.798183653	0.774352651	0.813229062	0.7061	0.77901
yale	0.776	0.789662797	0.726	0.738562797	0.718484848	0.742370629
ORL	0.7625	0.798183653	0.774352651	0.813229062	0.7061	0.77901

The Table 5, compares the performance of ELM, NN and LP respectively used as base classifier. It 370 is clear that for all the base classifiers used in our experiments, we obtain the maximum improvement of accuracy when the initial accuracy of the base classifier is between 70% and 80%.

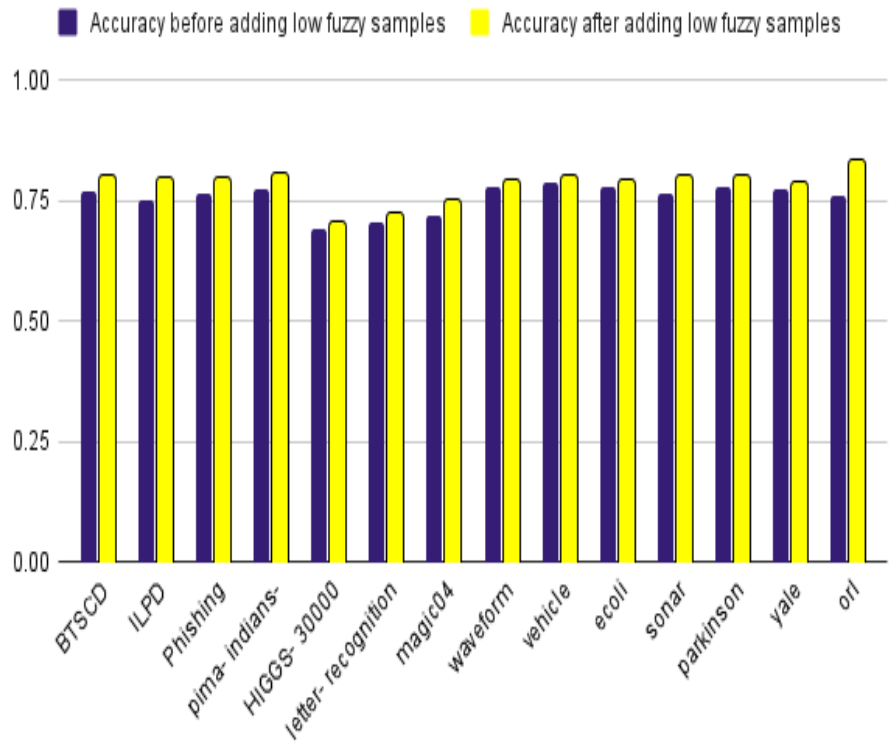


(a) ELM used as base classifier



(b) NN used as base classifier

Figure 5: Comparison among ELM, NN and LP used as base classifier



(c) LP used as base classifier

Figure 5: Comparison among ELM, NN and LP used as base classifier(cont.)

The Figure 5, graphically summarizes the results reported in Table 5 and compares the performance of our proposed algorithm used with different initial classifiers, such as ELM, NN and LP. From the Figure 5, it is clear that if we add low fuzzy samples from the testing dataset to the training dataset the accuracy is 375 improved and the improvement is maximum when the initial accuracy is between 70% and 80%.

6. Conclusions and future works

In this study, a new aspect of semi-supervised learning technique was explored to improve the performance of a classifier by using divide-and-conquer strategy. A small amount of labeled samples were used to build three initial classifiers, later those classifiers were used to classify huge amount of unlabeled samples and based on the degree of fuzziness those samples were categorized into low, medium and high fuzziness groups. Non-iterative single hidden layer neural network, back-propagation based neural network algorithms and label propagation algorithm were chosen as base classifiers because of their simplicity and efficiency. if the low fuzziness samples Extensive experiments were conducted for revealing the fact that for a trained classifier, if the low fuzziness samples from the testing dataset were added to the original training dataset and retrained the model then the training accuracy was substantially improved and this phenomenon was explained in the theory of learning from noisy data. It was also observed that, the improvement was largely dependent on the initial accuracy of the base

classifier and the improvement was found at its maximum when the initial accuracy was between 70% to 80%. One of our future works is to establish a robust mathematical model to explain why low-fuzziness samples have the enhanced impact on the learning performance.

7. Acknowledgement

The authors would like to thank Assoc. Prof. Walid Gomaa for his support during the completion of this paper. This work was supported in part by the National Natural Science Foundation of China (Grant 61772344 and Grant 61732011), in part by the Natural Science Foundation of SZU (Grant 827-000140, Grant 827-000230, and Grant 2017060), and in part by Guangdong Province 2014GKXM054.

References

References

[1] Abd El-Wahed, W. F. A multi-objective transportation problem under fuzziness. *Fuzzy sets and systems 117*, 1 (2001), 27–33.

[2] Alam, I. Removing the fuzziness from the fuzzy front-end of service innovations through customer interactions. *Industrial marketing management 35*, 4 (2006), 468–480.

[3] Ashfaq, R. A. R., Wang, X.-Z., Huang, J. Z., Abbas, H., and He, Y.-L. Fuzziness based semi-supervised learning approach for intrusion detection system. *Information sciences 378* (2017), 484–497.

[4] Banai, R. Fuzziness in geographical information systems: contributions from the analytic hierarchy process. *International Journal of Geographical Information Science 7*, 4 (1993), 315–329.

[5] Bezdek, J., et al. Fuzziness vs. probability-again (!?). *IEEE Transactions on Fuzzy systems 2*, 1 (1994), 1–3.

[6] Cardoso, J., and Camargo, H. *Fuzziness in Petri nets*, vol. 22. Springer Science & Business Media, 1998.

[7] De, S. K. On degree of fuzziness and fuzzy decision making. *Cybernetics and Systems 51*, 5 (2020), 600–614.

[8] Delgado, M., Vila, M. A., and Voxman, W. A fuzziness measure for fuzzy numbers: Applications. *Fuzzy sets and systems 94*, 2 (1998), 205–216.

[9] Dombi, J. A general class of fuzzy operators, the demorgan class of fuzzy operators and fuzziness measures induced by fuzzy operators. *Fuzzy sets and systems 8*, 2 (1982), 149–163.

[10] Fisher, P., Wood, J., and Cheng, T. Where is helvellyn? fuzziness of multi-scale landscape morphometry. *Transactions of the Institute of British Geographers 29*, 1 (2004), 106–128.

[11] Flores, O., Deniz, O., Soler-Lopez, M., and Orozco, M. Fuzziness and noise in nucleosomal architecture. *Nucleic acids research 42*, 8 (2014), 4934–4946.

[12] Fuhrmann, G. Fuzziness of concepts and concepts of fuzziness. *Synthese 75* (1988), 349–372.

[13] Fuxreiter, M. Fuzziness in protein interactions—a historical perspective. *Journal of molecular biology 430*, 16 (2018), 2278–2287.

[14] Gentili, P. L. The fuzziness of a chromogenic spirooxazine. *Dyes and Pigments 110* (2014), 235–248.

[15] Gentili, P. L. The fuzziness of the molecular world and its perspectives. *Molecules 23*, 8 (2018), 2074.

[16] Ghosh, A., Pal, N. R., and Pal, S. K. Self-organization for object extraction using a multilayer neural network and fuzziness mearsures.

[17] Hampton, J. A. Similarity-based categorization and fuzziness of natural categories. *Cognition 65*, 2-3 (1998), 137–165.

[18] Hawer, S., Schönmann, A., and Reinhart, G. Guideline for the classification and modelling of uncertainty and fuzziness. *Procedia Cirp 67* (2018), 52–57.

[19] Higashi, M., and Klir, G. J. On measures of fuzziness and fuzzy complements.

[20] Huang, L.-K., and Wang, M.-J. J. Image thresholding by minimizing the measures of fuzziness. *Pattern recognition 28*, 1 (1995), 41–51.

[21] Hudec, M. Fuzziness in information systems. *Switzerland (CHE): Springer Nature* (2016).

[22] Kacprzyk, J., Nurmi, H., and Fedrizzi, M. *Consensus under fuzziness*, vol. 10. Springer Science & Business Media, 2012.

[23] Kacprzyk, J., and Onisawa, T. *Reliability and safety analyses under fuzziness*. Springer, 1995.

[24] Kahraman, C., and Oztaysi, B. Supply chain management under fuzziness. *Recent Developments and Techniques 313* (2014), 1860–0808.

[25] Klir, G. J. Where do we stand on measures of uncertainty, ambiguity, fuzziness, and the like? *Fuzzy sets and systems 24*, 2 (1987), 141–160.

[26] Knopfmacher, J. On measures of fuzziness. *Journal of Mathematical Analysis and Applications 49*, 3 (1975), 529–534.

[27] Kordi, M., and Brandt, S. A. Effects of increasing fuzziness on analytic hierarchy process for spatial multicriteria decision analysis. *Computers, Environment and Urban Systems 36*, 1 (2012), 43–53.

[28] Kosko, B. Fuzziness vs. probability. *International Journal of General System 17*, 2-3 (1990), 211–240.

[29] Lerner, A. W., and Wanat, J. Fuzziness and bureaucracy. *Public administration review* (1983), 500–509.

[30] Liang, J., Chin, K.-S., Dang, C., and Yam, R. C. A new method for measuring uncertainty and fuzziness in rough set theory. *International Journal of General Systems 31*, 4 (2002), 331–342.

[31] Luhandjula, M. K. Linear programming under randomness and fuzziness. *Fuzzy Sets and Systems 10*, 1-3 (1983), 45–55.

[32] Miskei, M., Gregus, A., Sharma, R., Duro, N., Zsolyomi, F., and Fuxreiter, M. Fuzziness enables context dependence of protein interactions. *Febs Letters 591*, 17 (2017), 2682–2695.

[33] Montero, J., López, V., and Gómez, D. The role of fuzziness in decision making. *Fuzzy Logic: A Spectrum of Theoretical & Practical Issues* (2007), 337–349.

[34] Osman, M., Abo-Sinna, M. A., Amer, A. H., and Emam, O. A multi-level non-linear multi-objective decision-making under fuzziness. *Applied Mathematics and Computation 153*, 1 (2004), 239–252.

[35] Ozkan, I., and Turksen, I. Upper and lower values for the level of fuzziness in fcm. *Fuzzy Logic: A Spectrum of Theoretical & Practical Issues* (2007), 99–112.

[36] Pal, S. K., King, R. A., and Hashim, A. Automatic grey level thresholding through index of fuzziness and entropy. *Pattern Recognition Letters 1*, 3 (1983), 141–146.

[37] Patwary, M. J., Wang, X.-Z., and Yan, D. Impact of fuzziness measures on the performance of semi-supervised learning. *International Journal of Fuzzy Systems 21* (2019), 1430–1442.

[38] Ralescu, A. L., and Ralescu, D. A. Probability and fuzziness. *Information Sciences 34*, 2 (1984), 85–92.

[39] Reinertsen, D. G. Taking the fuzziness out of the fuzzy front end. *Research-Technology Management 42*, 6 (1999), 25–31.

[40] Saaty, T. L. Measuring the fuzziness of sets.

[41] Saaty, T. L. The analytic hierarchy process: A new approach to deal with fuzziness in architecture. *Architectural Science Review 25*, 3 (1982), 64–69.

[42] Sahu, A. K., Sahu, N. K., and Sahu, A. K. Fuzziness: a mathematical tool. In *Theoretical and Practical Advancements for Fuzzy System Integration*. IGI Global, 2017, pp. 1–30.

[43] Słowiński, R., and Hapke, M. Scheduling under fuzziness. *Physica, Heidelberg, Germany* (2000).

[44] Sowa, J. F. What is the source of fuzziness? *On Fuzziness: A Homage to Lotfi A. Zadeh–Volume 2* (2013), 645–652.

[45] Wang, P. The interpretation of fuzziness. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 26*, 2 (1996), 321–326.

[46] Wang, X., Ruan, D., and Kerre, E. E. *Mathematics of fuzziness—Basic issues*, vol. 245. Springer, 2009.

[47] Williams, C., et al. Fuzziness in legal english: what shall we do with'shall'? In *Legal language and the search for clarity*, vol. 1. Peter Lang, 2006, pp. 237–263.

[48] Worrall, J. L. Public perceptions of police efficacy and image: The "fuzziness" of support for the police. *American Journal of Criminal Justice 24* (1999), 47–66.

[49] Yager, R. R. On the measure of fuzziness and negation. ii. lattices. *Information and control 44*, 3 (1980), 236–260.

[50] Zhang, Q. Fuzziness-vagueness-generality-ambiguity. *Journal of pragmatics 29*, 1 (1998), 13–31.