

Long Term Bank Failure Prediction using Fuzzy Refinement-based Transductive Transfer Learning

Vahid Behbood, Jie Lu and Guangquan Zhang
Decision Systems & E-Service Intelligence Research Laboratory
Centre for Quantum Computing and Intelligent System
School of Software, Faculty of Engineering and Information Technology
University of Technology Sydney
POBOX123, Broadway, NSW 2007, Australia
vahid.behbood, jie.lu & guangquan.zhang@uts.edu.au}

Abstract- machine learning algorithms, which have been considered as robust methods in different computational fields, assume that the training and test data are drawn from the same distribution. This assumption may be violated in many real world applications like bank failure prediction because training and test data may come from different time periods or domains. An efficient novel algorithm known as Fuzzy Refinement (FR) is proposed in this paper to solve this problem and improve the performance. The algorithm utilizes the fuzzy system and similarity concept to modify the instances' labels in target domain which was initially predicted by shift-unaware Fuzzy Neural Network (FNN) proposed by [1]. The experiments are performed using bank failure financial data of United States to evaluate the algorithm performance. The results address a significant improvement in the predictive accuracy of FNN due to applying the proposed algorithm.

Keywords- Transfer Learning; Fuzzy Neural Network; Fuzzy Sets; Long Term Prediction; Bank Failure Prediction

1. INTRODUCTION

Although machine learning technologies have gained remarkable attention among researches in different computational fields including classification, clustering and prediction, most of them work very well under a common assumption that the training data and test data has the same future space and same distribution. So once the distribution or feature space changes, the models need to be rebuild and retrained from the scratch using new collected training data. In many real world applications, particularly finance business, recollecting new training data and retraining the model is very expensive or practically impossible. For example, more and more labeled financial data become out of date and may not follow the same distribution as time goes by and thus past labeled data cannot be used to reliably predict financial situation of an organization. It would be so useful if the necessary of the recollecting labeled data can be reduced and the data from different time periods or domains can be utilized to assist the current learning task such as failure prediction. In this situation a new field of study has been emerged recently which is called *Transfer Learning*.

Transfer learning, which is different from the traditional machine learning and semi-supervised algorithms [2-4], considers that the domains and tasks of training and testing

data can be different [5]. The study of transfer learning has been inspired by this fact that human beings can utilize the knowledge, which learned previously, to solve upcoming similar but not identical problems much faster and even better. The study of Transfer learning has begun since 1995 in different names: learning to learn, life-long learning, meta learning, multi-task learning [6]. According to the definition of Transfer learning which will be explained in the next Section, this techniques can be categorized into different settings.

However there are many researches in this field, most of them have used statistical models and there are a few studies have used soft computing techniques. In this paper a novel Transductive Transfer learning algorithm is proposed for fuzzy neural network using fuzzy sets concept which has been proved to be a successful soft computing techniques in machine learning. Likewise most of existence methods aim to refine decision boundary and models while the proposed algorithm has developed a method namely bridge refinement [7] to modify the prediction labels in target domain and focus on currently given test data. This ability makes the algorithm more practical and independent from the prediction method.

The rest of paper is organized as follow. In Section 2, some preliminaries concepts including the definition of Transductive Transfer learning and related works are given. Section 3 proposes the Fuzzy Refinement (FR) algorithm and Section 4 describes the experimental illustration and results comparing Fuzzy Neural Network (FNN) with FNN-FR (FNN using Fuzzy Refinement). Section 5 concludes this paper and discusses future researches.

2. PRELIMINARIES AND RELATED WORKS

In this section the definition of transfer learning and particularly Transductive Transfer learning is introduced and then the related works are presented. Also the FNN, which is proposed by [1], is briefly explained at the end.

A. Definition 1 (Domain)

A domain consists of two components: 1) feature space X and 2) marginal probability distribution $P(X)$ where $X = \{x_1, \dots, x_n\}$, Domain is denoted by $D = \{X, P(X)\}$.

B. Definition 2 (Task)

A task consists of two components: 1) a label space $Y = \{y_1, \dots, y_m\}$ and an objective predictive function $f(\cdot)$ which is not observed and to be learned by pairs $\{x_i, y_i\}$. Task is denoted by $T = \{Y, f(\cdot)\}$.

C. Definition 3 (Transfer learning)

Given a source domain D_s and learning task T_s , a target domain D_t and learning task T_t , transfer learning aims to help improve the learning of the target predictive function $f_t(\cdot)$ in D_t using the knowledge in D_s and T_s , where $D_s \neq D_t$ or $T_s \neq T_t$.

D. Definition 4 (Transductive Transfer learning)

A category of transfer learning in which $T_s = T_t$ and $D_s \neq D_t$ which implies that either $X_t \neq X_s$ or $P_t(x) \neq P_s(x)$, in this situation no labeled data is available in the target domain while a lot of labeled data in the source domain are available.

E. Related works

Transductive Transfer learning was first proposed by [8]. It mentioned that the source and target tasks are the same, although the domains are different due to their distribution. Almost all studies in Transductive Transfer learning can be categorized into two groups[6]:

1) *Transferring the knowledge of instances*: this approach is motivated by the importance of samples and it tries to find an optimum weight for each instance to learn a precise model for target domain. There are some papers in this category which can be found in a recently published book [9] by Quionero-Candela et al.

2) *Transferring the knowledge of feature representation*: this approach focuses on feature space and attempts to extract and/or convert some relevant features which reduce the difference between the domains. Blitzer et al [10-12] proposed SCL algorithm to define a pivot features on the target domain from both domains and then uses unlabeled instances from target do create the classification model. Dai et al. [13] proposed a co-clustering based algorithm to propagate the label information across domains. In [7] Xing et al. introduced a novel algorithm known as bridge refinement to modify the predicted labels of instances from target domain. This study, which is the base idea of this paper, utilizes mixture distribution of the training and test data as a bridge to transfer feature distribution from source domain to target one. Xue et al. [14] presented a cross-domain text classification known as TPLSA to integrate target instances and source instances into a unified probabilistic model.

F. Fuzzy Neural Network

Assume that $X = [x_1, x_2, \dots, x_n]$ and $Y = [y_1, y_2, \dots, y_m]$ represent the vector of inputs and outputs respectively to and from the network. In addition, assume the vector $D = [d_1, d_2, \dots, d_m]$ represents the desired (actual) outputs required during the learning phase.

The proposed network has five layers of nodes: (1) Input layer consisting of input nodes $It_i, i \in \{1, 2, \dots, n\}$, which have a single input x_i . (2) Input cluster layer including cluster nodes

$ITerm_{ij}, i \in \{1, 2, \dots, n\} \& j \in \{1, \dots, k_i\}$ which represent linguistic terms for each input. For instance, $ITerm_{23}$ represents the third linguistic term of input x_2 . (3) Rule layer consisting of rule nodes $R_h, h \in \{1, 2, \dots, l; l = \sum_{i=1}^n k_i \times \sum_{j=1}^m k_j\}$, which are representative of each rule and connect only one labels of each input ($x_i; i \in \{1, 2, \dots, n\}$) to one labels of each output ($y_i; i \in \{1, 2, \dots, m\}$). (4) Output cluster layer including cluster nodes $OTerm_{ij}, i \in \{1, 2, \dots, m\} \& j \in \{1, \dots, k_i\}$ which represent linguistic terms for each output. For instance, $OTerm_{35}$ represents the fifth linguistic term of output y_3 . (5) Output layer consisting of output nodes $Ot_i, i \in \{1, 2, \dots, m\}$, which have single output y_i . Figure 1 shows the structure of the proposed FNN.

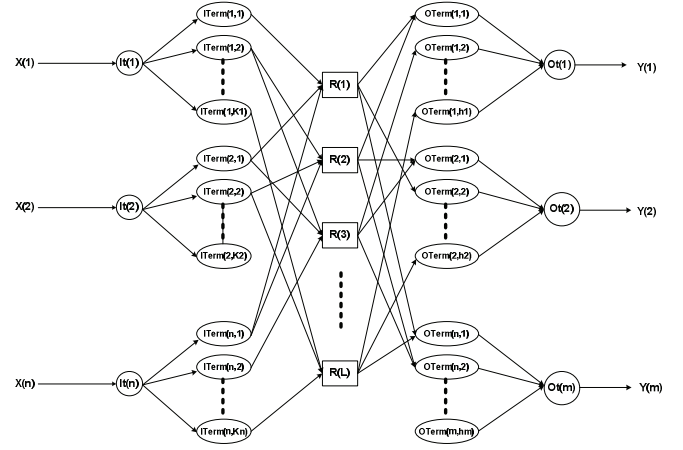


Fig1. FNN structure

The rule generation algorithm is developed by formulating an algorithm which is based on the structure of our proposed FNN. In this algorithm, ISP_h is the terms for all input linguistic terms (layer 2 nodes) that contribute to antecedent of rule node R_h and OSP_h refers to all output linguistic terms (layer 4 nodes) that form the consequent of rule node R_h :

$$ISP_h = \{ITerm_{1P_1}, ITerm_{2P_2}, \dots, ITerm_{nP_n}\} \quad \text{and} \\ OSP_h = \{OTerm_{1Q_1}, OTerm_{2Q_2}, \dots, OTerm_{mQ_m}\}$$

if and only if

$$R_h: \text{If } x_1 \text{ is } ITerm_{1P_1} \text{ and } x_2 \text{ is } ITerm_{2P_2} \text{ and } \dots \text{ and } x_n \text{ is } ITerm_{nP_n} \text{ then } y_1 \text{ is } OTerm_{1Q_1} \text{ and } y_2 \text{ is } OTerm_{2Q_2} \text{ and } \dots \text{ and } y_m \text{ is } OTerm_{mQ_m} \quad (1)$$

The rule generation algorithm has three main steps as follows: Algorithm 1:

Step 1: Firing input and desired vectors

The input vector $X(T) = [x_1(T), \dots, x_n(T)]$ and the desired vector $D(T) = [d_1(T), \dots, d_m(T)]$ are feed to the network through layer 1 and layer 5 respectively at T-th

epoch. According input and output linguistic terms (fuzzy sets) membership function, the membership value of inputs and outputs in each term is evaluated and then the linguistic terms of each input and output with maximum membership value is selected:

$$\begin{aligned} MITerm(T) = \\ \{ITerm_{1b_1}, ITerm_{2b_2}, \dots, ITerm_{nb_n} | \mu_{ITerm_{ib_i}}(x_i(T)) = \\ \text{Max} \left\{ \mu_{ITerm_{ij}}(x_i(T)), j \in \{1, \dots, k_i\} \right\} \end{aligned} \quad (2)$$

$$\begin{aligned} MOTerm(T) = \\ \{OTerm_{1b_1}, OTerm_{2b_2}, \dots, OTerm_{mb_m} | \mu_{OTerm_{ib_i}}(d_i(T)) = \\ \text{Max} \left\{ \mu_{OTerm_{ij}}(d_i(T)), j \in \{1, \dots, k_i\} \right\} \end{aligned} \quad (3)$$

Step 2: Indicating nominated rule

In this step, the rule which has to be nominated for updating is selected in each epoch. The rule which satisfies (4) in epoch T-th is qualified for an update:

$$\begin{aligned} ISP_h = MITerm(T) \text{ and } OSP_h = MOTerm(T) \\ h \in \{1, 2, \dots, l\}, l = \sum_{i=1}^n k_i \times \sum_{i=1}^m k_i \end{aligned} \quad (4)$$

Step 3: Rule updating

To update the rule weight in epoch T-th, equation (5) is applied:

$$\begin{aligned} W_h(T) = W_h(T-1) + (M_{ISP_h}(T) \times M_{OSP_h}(T)), \\ W_h(0) = 0, h \in \{1, 2, \dots, l\} \end{aligned} \quad (5)$$

where $M_{ISP_h}(T) = \mu_{ITerm_{1b_1}} \times \mu_{ITerm_{2b_2}} \times \dots \times \mu_{ITerm_{nb_n}}$ and $M_{OSP_h}(T) = \mu_{OTerm_{1b_1}} \times \mu_{OTerm_{2b_2}} \times \dots \times \mu_{OTerm_{mb_m}}$.

To prune the weak rules at the end of the rule generation phase, a predefined threshold parameter *Thresh* is considered and then each rule satisfying (6) will be pruned at the end of the training step.

$$l \times \left(\frac{W_h}{\sum_{k=1}^l W_k} \right) < \text{Thresh} \quad (6)$$

Through using this algorithm we can obtain an initial consistent and compact fuzzy rule base which needs to be modified and updated to gain more accurate results. In the next section, a learning algorithm is proposed to optimize the rule base.

The adaptive inference-based learning algorithm aims to update the fuzzy rule base, which is obtained by rule generation algorithm, to improve the prediction accuracy. This algorithm tries to adapt the parametric t-norm of inference system to reduce the significance of rules causing errors and consequently augment accuracy.

Assume that $X(T) = [x_1(T), \dots, x_n(T)]$ and $D(T) = [d_1(T), \dots, d_m(T)]$ are input and desired output vectors in T-th epoch of training respectively and there is a fuzzy rule base

that includes l rules and each rule, R_i (i^{th} rule) has the form as shown in (7).

$$\begin{aligned} R_i: \text{If } x_1 \text{ is } ITerm_{1j_1} \dots \text{and } x_n \text{ is } ITerm_{nj_n} \\ \text{Then } y_1 \text{ is } OTerm_{1h_1} \dots \text{and } y_m \text{ is } OTerm_{mh_m} \end{aligned} \quad (7)$$

The proposed **adaptive inference-based learning algorithm** has seven steps as follows:

Algorithm 2:

Step 1: Fuzzyfying

Input nodes act as singleton fuzzifier that fuzzyfy the crisp-valued inputs. So, the activation function of each node in layer 1 is defined as:

$$\begin{aligned} Z_i^{(1)}(T) = f^{(1)}(x_i(T)) = \mu_{\tilde{x}_i}(\tilde{x}_i(T)) = \\ \begin{cases} 1, & \text{if } \tilde{x}_i(T) = x_i(T) \\ 0, & \text{Otherwise} \end{cases} \quad i \in \{1, 2, \dots, n\} \end{aligned} \quad (8)$$

where $\tilde{x}_i(T)$ is the fuzzyfied equivalent of crisp input $x_i(T)$ and $Z_i^{(1)}(T)$ is the output of node $It_i, i \in \{1, 2, \dots, n\}$ at T-th epoch.

Step 2: Antecedent matching

To perform antecedent matching of fuzzyfied inputs against linguistic terms, input cluster nodes compute the similarity measure which is the input membership value in each term. The activation function of each node in layer 2 is defined as:

$$\begin{aligned} Z_{ij}^{(2)}(T) = f^{(2)}(Z_i^{(1)}(T)) = \mu_{ITerm_{ij}}(x_i(T)) = \\ \begin{cases} 0, & \text{if } x_i(T) \leq l_{ij} \\ \frac{x_i - l_{ij}}{u_{ij} - l_{ij}}, & \text{if } l_{ij} \leq x_i \leq u_{ij} \\ 1, & \text{if } u_{ij} \leq x_i \leq v_{ij} \\ \frac{r_{ij} - x_i}{r_{ij} - u_{ij}}, & \text{if } v_{ij} \leq x_i \leq r_{ij} \end{cases} \\ i \in \{1, 2, \dots, n\} \quad j \in \{1, 2, \dots, k_i\} \end{aligned} \quad (9)$$

where $Z_{ij}^{(2)}(T)$ is the output of node $ITerm_{ij}$ at T-th epoch.

Step 3: Rule fulfillment

To calculate the strength of activation of an antecedent for all rules with input vector $X(T)$, rule nodes compute the degree of rule fulfillment as their outputs. The higher the degree of fulfillment, the greater is the compatibility of the input to the antecedent of the rule. Hence, the activation function of each node, $R_h; h \in \{1, 2, \dots, l\}$, in layer 3 is defined as:

$$Z_{R_h}^{(3)}(T) = f^{(3)}(Z_{(ij)_h}^{(2)}(T), i \in \{1, 2, \dots, n\}) =$$

$$T \left(Z_{(ij)_h}^{(2)}(T), i \in \{1, 2, \dots, n\} \right) \quad (10)$$

where $Z_{(ij)_h}^{(2)}(T)$ is the output of the j -th term of the i -th input that is connected to the rule R_h and T is the conjunction operator. In order to adapt the inference system, this operator is considered as parameterized t-norm. Some researchers [15, 16] have shown that a tuning parameter can significantly improve the accuracy of linguistic fuzzy systems. Therefore, Dubios t-norm shown in (13) is used as conjunction operator in this step because it provides better accuracy than other parametric t-norms [15].

$$T_{Dubios}(x, y, \alpha) = \frac{x \times y}{\max(x, y, \alpha)} \quad (11)$$

Dubios t-norm operates as the minimum and an algebraic product with $\alpha = 0$ and $\alpha = 1$ respectively. The activation function is represented as:

$$\begin{aligned} Z_{R_h}^{(3)}(T) &= f^{(3)} \left(Z_{(ij)_h}^{(2)}(T), i \in \{1, 2, \dots, n\} \right) = \\ &T_{Dubios} \left(Z_{(ij)_h}^{(2)}(T), i \in \{1, 2, \dots, n\}, \alpha_h(T) \right) = \\ &\left(\prod_{i=1}^n Z_{(ij)_h}^{(2)}(T) \right) / \text{Max} \{ Z_{(ij)_h}^{(2)}(T), i \in \{1, 2, \dots, n\}, \alpha_h(T) \} \end{aligned} \quad (12)$$

where $Z_{R_h}^{(3)}(T)$ is the output of node R_h at T -th epoch. In addition, $\alpha(0) = (\alpha_1(0), \alpha_2(0), \dots, \alpha_l(0))$ is initialized as 0 at the beginning of the learning algorithm, and then, during the feedback learning phase, each α_h may be increased separately.

Step 4: Consequent derivation

In this step, an aggregator operator is used to derive the consequent of fuzzy rules. Max operator which is considered as s-norm uses rule weights and rule outputs to compute the inferred output according to each rule in the rule base. Therefore, the activation function for each node in layer 4 becomes:

$$\begin{aligned} Z_{ij}^{(4)}(T) &= f_{ij}^{(4)}(Z_{R_h}^{(3)}(T), h \in \{1, \dots, l\}, W_h) = \\ &\text{Max}(\prod_{h=1}^l W_h \times Z_{R_h}^{(3)}(T)) \end{aligned} \quad (13)$$

where $Z_{R_h}^{(3)}(T)$ is the output of the h -th rule node in layer 3 that is connected to $OTerm_{ij}$ as its consequent.

Step 5: Deffuzifying

To deffuzify the derived fuzzy outputs, the weighted center of averaging (COA) technique [17] is used. So, the activation function in layer 5 is as follows:

$$\begin{aligned} y_i(T) &= Z_i^{(5)}(t) = f_i^{(5)}(z_{ij}^{(4)}(T), j = (1, 2, \dots, h_j)) = \\ &\frac{\sum_{j=1}^{h_j} (Z_{ij}^{(4)}(T) \times \tilde{m}_{ij})}{\sum_{j=1}^{h_j} Z_{ij}^{(4)}(T)} \end{aligned} \quad (14)$$

where $\tilde{m}_{ij} = \frac{u_{ij} + v_{ij}}{2}$ and $y_i(T)$ is the output of i -th node OT_i in layer 5.

Step 6: Error evaluation

During the feed forward phase, the input vector $X(T)$ is presented to the network and the output vector $Y(T)$ results in T -th epoch. At the beginning of the feed backward pass, $Y(T)$ is compared against the desired output vector $D(T)$ and the error is used to modify the vector $\alpha(T)$ in T -th epoch. Regarding the framework of imbalanced data-sets, the geometric mean presented in (2) should be used to calculate the error for all outputs. The following equation denotes the error in the t -th epoch.

$$\begin{aligned} \text{Error}(T) &= (1 - G(T)) = \\ &\left(1 - \sqrt{\frac{TP(T)}{TP(T) \times FN(T)} \times \frac{TN(T)}{FP(T) \times TN(T)}} \right) \end{aligned} \quad (15)$$

where $G(T)$ is the accuracy of model till T -th epoch of training.

Step 7: Modifying t-norm

All links in the proposed network do not have a particular weight which affects the error in the feed backward phase. The first issue to be taken into consideration in order to achieve better accuracy is the significance of antecedents that make inaccuracy should be reduced. Dobius t-norm performs like minimum t-norm if the following equation (15) becomes satisfied and Dobius t-norm approaches to product t-norm, which reduces the significance of the antecedent causing inaccuracy, by increasing $\alpha_h(T)$. Hence, better prediction can be achieved by increasing $\alpha_h(T)$ of rule R_h which participates in producing $\text{Error}(T)$.

$$Z_{(ij)_h}^{(2)}(T) \geq \alpha_h(T) \quad (16)$$

To modify the parameter $\alpha_h(T)$ in each iteration (17) is used.

$$\alpha_h(T + 1) = \alpha_h(T) - \sigma \times (\text{Error}(T) \times W_h) \quad (17)$$

where σ is a positive experimental consistent less than 1.

3. FUZZY REFINEMENT ALGORITHM

The bridge refinement algorithm[7] is motivated by PageRank algorithm [18] and it assumes that the conditional probability of a specified label C given an instance d , does not vary among different distributions: $P_{D_s}(C|d) = P_{D_{sut}}(C|d) = P_{D_t}(C|d)$ although the marginal probability of instance d , $P(d)$, varies. The reason is based on the fact that if an identical instance appears in the target and source domain, the predicted label should be the same. In further, the more similar instances are in the target domain, the more the probability is they have the same label. This situation forms a mutual reinforcement relationship between instances in target domain

and source domain and can be used to correct the predicted labels. Not only is this assumption considered in our research but also a complementary idea of that is applied. We assume that more different instances in the target domain, the less the probability is they have the same label.

Given $\widetilde{F}^s = \{\widetilde{f}_1^s, \dots, \widetilde{f}_m^s\}$ and $\widetilde{F}^t = \{\widetilde{f}_1^t, \dots, \widetilde{f}_m^t\}$ are the fuzzy feature sets for source and target domain respectively. The number of these features is the same for both target and source domains but the membership function of these fuzzy sets are different. This assumption implies a Transductive Transfer learning problem in which the feature space is the same but the distribution is different. Given $X^s = \{x_1^s, \dots, x_{n_s}^s\}$ and $X^t = \{x_1^t, \dots, x_{n_t}^t\}$ are the instances of source and target domain respectively. Given $\widetilde{Y} = \{\widetilde{y}_1, \dots, \widetilde{y}_l\}$ is the predictive fuzzy label set which is the same for both domains. Given $G(\cdot)$ is a predictive model, which is fuzzy neural network (FNN) proposed by [1], such that $G(x_i) = \{\mu_{\widetilde{y}_1}(x_i), \dots, \mu_{\widetilde{y}_l}(x_i)\}$ is the vector of membership value of x_i that belongs to each label. The parameter k is the number of the most and least similar instances which is applied to refine the predicted labels and α is used to specify the degree of refinement in each iteration. The steps of Fuzzy Refinement (FR) algorithm are described as follow.

[Fuzzy Refinement algorithm]

Input: $F^s, F^t, X^s, X^t, \widetilde{Y}, G(\cdot), k$ and α .

Output: Label matrix for instances of target domain, MR.

[Begin]

Step 1: At the first step the singleton fuzzifier is used as follow to fuzzyfy the crisp-valued of instances from both domains.

$$\mu_{\widetilde{x}_i}(\widetilde{x}_i(T)) = \begin{cases} 1, & \text{if } \widetilde{x}_i(T) = x_i \\ 0, & \text{Otherwise} \end{cases} \quad i \in \{1, 2, \dots, n\} \quad (18)$$

where $\widetilde{x}_i$ is the fuzzyfied equivalent of crisp input x_i , $i \in \{1, 2, \dots, n\}$.

Step 2: In this step the similarity matrix is calculated by following equation:

$$M_{n_s \times n_t} = \{m_{(i,j)} | m_{(i,j)} = 1 - \sqrt{\sum_{k=1}^m ((\mu_{\widetilde{f}_k^s}(x_i^s) - \mu_{\widetilde{f}_k^t}(x_j^t))^2)} \quad (19)$$

Step 3: In this step the pseudo label matrix for target domain instances and label matrix for source domain instances are calculated by FNN as follow.

$$G_{n_s \times l}^s = \{g_{(i,j)}^s | g_{(i,j)}^s = FNN(x_i^s) = \mu_{\widetilde{y}_j}(x_i^s)\} \quad (20)$$

$$G_{n_t \times l}^t = \{g_{(i,j)}^t | g_{(i,j)}^t = FNN(x_i^t) = \mu_{\widetilde{y}_j}(x_i^t)\} \quad (21)$$

Step 4: In this step for each target instance, k most similar and k least similar instances from source domain are found. N^i and V^i includes the k most similar and k least similar instances from source domain to target instance i .

for $i = 1$ to n_t

$$N_{1 \times k}^i = \{n_1^i, \dots, n_k^i\} = \{n_j^i = x_p^s | m(p, i) = \text{Max} \{M(\cdot, i)\}, j = 1, \dots, k\} \quad (22)$$

$$V_{1 \times k}^i = \{v_1^i, \dots, v_k^i\} = \{v_j^i = x_p^s | m(p, i) = \text{Min} \{M(\cdot, i)\}, j = 1, \dots, k\} \quad (23)$$

Next i

Step 5: The pseudo label for each instance from target domain is refined in this step according to the following equation. $\widetilde{MR}_{n_t \times l}$ is the fuzzy label matrix for instances of target domain.

For $i = 1$ to n_t

For $j = 1$ to l

Do

$$MR_{(i,j)}(t) = (1 - \alpha)g_{(i,j)}^t + \alpha \left(\frac{\sum_{p=1}^k \mu_{\widetilde{y}_j}(n_p^i) - \sum_{p=1}^k \mu_{\widetilde{y}_j}(v_p^i)}{k} \right) (MR_{(i,j)}(t - 1)) \quad (24)$$

Until MR converges

Next j

Next i

[End]

As can be seen from the equation (24) the refinement is based on the fact that the label of most similar and least similar instances to the target instance is used to modify the pseudo label of target instance which was initialized by FNN. As result of this algorithm a fuzzy label matrix for all instances of target domain is achieved. Each row of this matrix indicates the membership value of one instance to all label classes. To find the final label for each instance namely x_i^t , the following equation can be used.

$$MR = \arg \max_j \{\widetilde{MR}_{i,j} | j = 1, \dots, l\} \quad (25)$$

It should be mentioned that the above algorithm is implemented by Matlab which has the ability of matrix language programming.

4. EXPERIMENTS

In this section we would like to validate the proposed algorithm using real world financial data considering different scenarios. The task in this experiment is bank failure prediction and the prediction label has two classes: Failed; Survived. We perform the experiments to examine the algorithm whether it can transfer knowledge from different time periods.

A. Data Sets

The data set and financial variables are extracted from Call Report Data, which is downloaded from the website of Federal Reserve Bank of Chicago (<http://www.chicagofed.org>) and the status of each bank is identified according Federal Financial Institutions Examination Council (<http://www.ffiec.gov/nicpubweb/nicweb/NicHome.aspx>). The dataset, which is showed in Table 1, includes the observation period of the Survived banks of 21 years from Jun 1980 to Dec 2000 and based on the history of each bank in FFIEC.

There are 548 Failed banks and 2555 Survived ones which are presented by [19, 20]. Although, Tung et al. [19] used nine financial features, according to their statistical significance and correlation, it is observed that the model with three features has less created rules, less computational load and more prediction accuracy. Each feature is ranked based on the importance of a feature as a result of a future selection process and three features with the highest grade are selected [20]. The definition of all features and their expected impacts on bank failure are described in Table 2. The proposed approach is run by nine inputs and three inputs separately and the results are then compared.

TABLE 1
DATA SETS

Year	Total Number banks	Number of Survived banks	Number of Failed banks
1990	2156	1843(85.48%)	313 (14.52%)
1995	2539	2192(86.34%)	347(13.66%)
1998	2943	2585(87.84%)	358(12.16%)
2000	3103	2555(82.34%)	548(17.66%)

Number of available records in second data set for each scenario

TABLE 2
VARIABLES

Financial Variable
1) *CAPADE: average total equity capital /average total assets
2) OLAQLY: average (accumulated)loan loss allowance/average total loans and leases, gross
3) PROBLO: Loans 90+days late/average total loans & leases, gross
4) *PLAQLY: Loan loss provision /average total loans and leases, gross
5) NIEOIN: noninterest expense/operating income
6) NINMAR: (total interest income – interest expense)/average total asset
7) *ROE: (net income after tax +applicable income taxes)/average total equity capital
8) LIQUID: (average cash + average federal funds sold)/ (average total deposit + average fed funds purchased + average banks' liability on acceptance + average other liabilities)
9) GROWLA: (total loans & leases (t) – total loans & leases (t-1))/ total loans & leases (t-1)

Definition of variables (three selected variables are marked with *).

B. Research design and preprocessing

The domain instances (training data) are selected from data until year 1990 and the target instances (test data) are selected from records of year 1995, 1998 and 2000, 5, 8 and 10 years after 1990 (see Table 1). We compare the results of FNN using proposed algorithm denoted as FNN-FR with FNN to find out the improvement gained due to fuzzy refinement. Likewise the evaluation is performed using two categories of feature sets: nine variables; three variables. In conclusion the experiments have been performed based on 12 scenarios.

To reduce the influence of imbalance data-sets problem, the SMOTE technique is applied to train data sets. The number of Failed banks increases to the number of Survived ones to achieve balanced data set which improves the accuracy of prediction without losing important information.

In each scenario, the training data set splits into two pools: (1) Failed banks denoted with output 1; (2) Survived banks denoted with output 0. There are five cross-validation groups which include instances of both pools randomly to form the training set. The training set of the five groups are mutually exclusive. The FNN and FNN-FR are trained using training data sets and then evaluated by the testing data sets. The accuracy of the each experiment, which is the mean value of cross-validation groups' accuracy, is calculated using following equation.

$$GM = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{FP+TN}} \quad (9)$$

where $\frac{TP}{TP+FN}$ is called *sensitivity* and $\frac{TN}{FP+TN}$ is called *specificity*, measure the effectiveness of the prediction algorithm in any class. The proposed metric (9) is a geometric mean of *sensitivity* and *specificity*, because both of them are expected to be high simultaneously

C. Experiment results analysis

In this section the results which are gained from the experiment is reported. To ensure that the Fuzzy Refinement (FR) algorithm for Transfer learning makes significant improvement, the performance of FNN applying FR (FNN-FR) is compared with FNN which generates unrefined results.

As a result of this comparison, the Figure 2 represents the difference of FNN and FNN-FR performance using three financial features. In all data sets the FNN-FR outperforms the FNN. The relative increase is achieved by 10%, 14% and 15% on 1995, 1998 and 2000 data sets. Surprisingly it shows that the influence of proposed algorithm become more significant once the period of prediction become farther and so the difference between target domain and source domain become greater. Figure 3 shows the results of FNN and FNN-FR using nine financial features. The relative growth is 10%, 20% and 25% on 1995, 1998 and 2000 data sets. This growth in accuracy, which is gained due to applying proposed algorithm, is more than previous experiments. The reason may be that the proposed algorithm works better on larger feature space and more number of features. The summary of all 12 scenarios results are shown in the Table 3.

To clarify this improvement clearly the t-test is applied on the level of confidence of 0.05 on 12 scenarios. As it is showed in Table 4, the null hypothesis of the equality of average accuracy is rejected for FNN-FR. So it can be concluded that the proposed algorithm can significantly improve the predictive accuracy of FNN for long term prediction.

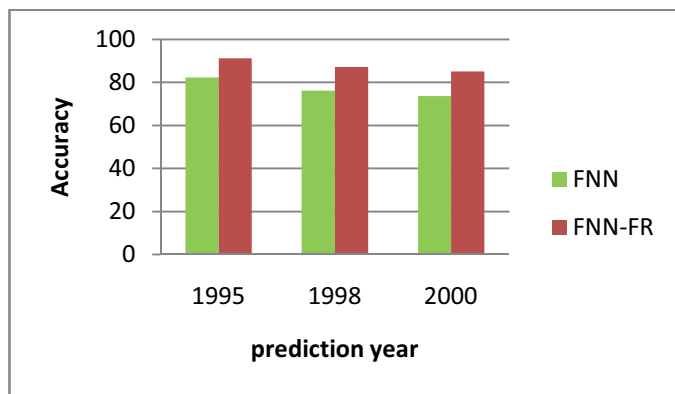


Fig. 2 Comparison FNN and FNN-FR accuracy using three variables

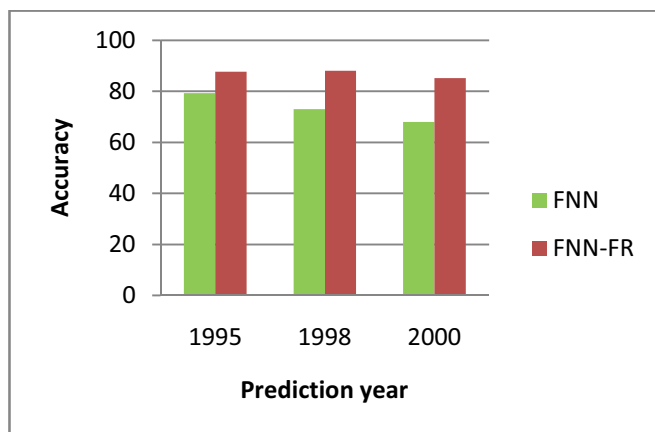


Fig. 3 Comparison FNN and FNN-FR accuracy using nine features.

TABLE 3
RESULTS

		12 Scenarios		
Method	N.O feature	1995	1998	2000
FNN	3	82.36	76.15	73.29
	9	79.25	72.98	68.03
FNN-FR	3	91.15	87.22	85.07
	9	87.66	88.10	85.17

The accuracy of FNN and FNN-FR

TABLE 4
T-TEST RESULT

Algorithm	Mean Value	Hypothesis	t	DF	P-Value
FNN	75.4167	FNN vs. FNN-FR	-8.323	5	0.00
FNN_FR	87.3950				

The t-test result for comparison of FNN and FNN-FR

D. Parameter sensitivity

The proposed algorithm has two parameters, k and α , which need to be set in performing experiments. In this section we investigate the influence of these parameters on the

performance of the Fuzzy refinement algorithm. To do this the performance of the algorithm is examined using different value of these parameters on 6 scenarios. The accuracy of the algorithm for different value of k is shown in the Figure 4. It shows that the performance is not greatly sensitive to k as long as k is large enough. According to this figure the value of 80 is the best value for k which is chosen in this research. Figure 5 shows the accuracy by applying different values of α . It shows that 0.66 is the best value for α which is selected in this paper.

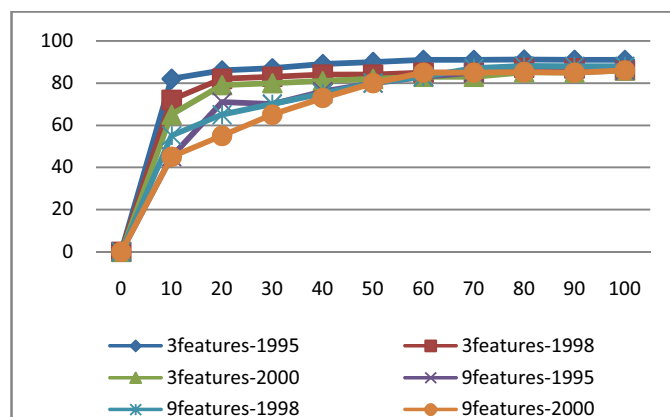


Figure 4. The accuracy of FNN-FR using different value of K

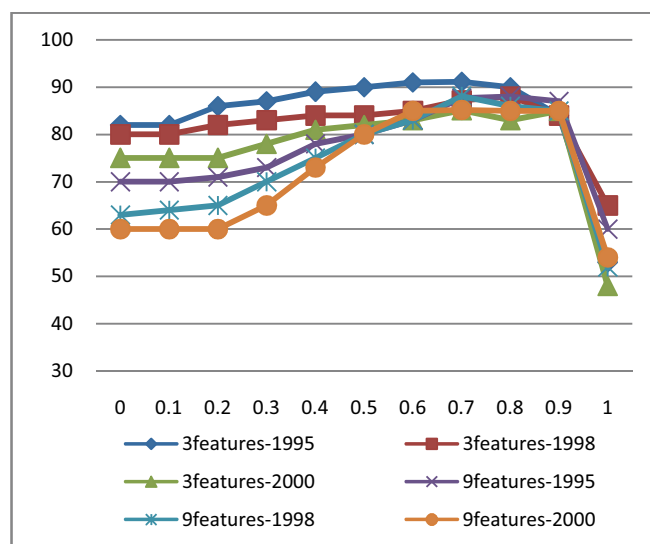


Figure 5. The accuracy of FNN-FR using different value of α

5. CONCLUSION AND FUTURE STUDY

In this paper Fuzzy Refinement algorithm is proposed to solve the Transductive Transfer Learning problem. The algorithm concentrates on the modification the instances' labels in target domain and uses the fuzzy concepts to improve the predictive accuracy. The FNN which is proposed by [1] is considered as prediction model to determine the initial labels for target instances. According to the empirical results, the proposed algorithm brings about remarkable improvement in performance. It demonstrates a

significant increase in predictive accuracy when the algorithm is applied particularly for farther time period ahead prediction

The approach which is mentioned in [7] to apply two refinement steps can be applied in the algorithm as future research. Also developing an algorithm based the proposed algorithm which can extract the relevant features to reduce the difference between domains is desirable for future work. Finally evaluating the proposed algorithm in the situation that there are some labeled data in target domain can be an interesting study for future as well.

RERERENCES

- [1] V. Behbood, J. Lu, and G. Zhang, "Adaptive Inference-based learning and rule generation algorithms in fuzzy neural network for failure prediction," in *2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering (ISKE 2010)* China, 2010, pp. 33-38.
- [2] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text Classification from Labeled and Unlabeled Documents using EM," *Mach. Learn.*, vol. 39, pp. 103-134, 2000.
- [3] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory* Madison, Wisconsin, United States: ACM, 1998.
- [4] T. Joachims, "Transductive Inference for Text Classification using Support Vector Machines," in *Proceedings of the Sixteenth International Conference on Machine Learning*: Morgan Kaufmann Publishers Inc., 1999.
- [5] F. Gabriel Pui Cheong, J. X. Yu, L. Hongjun, and P. S. Yu, "Text classification without negative examples revisit," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 18, pp. 6-20, 2006.
- [6] P. Sinno Jialin and Y. Qiang, "A Survey on Transfer Learning," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, pp. 1345-1359, 2010.
- [7] D. Xing, W. Dai, G.-R. Xue, and Y. Yu, "Bridged Refinement for Transfer Learning," in *Knowledge Discovery in Databases: PKDD 2007*. vol. 4702: Springer Berlin / Heidelberg, 2007, pp. 324-335.
- [8] A. Arnold, R. Nallapati, and W. W. Cohen, "A Comparative Study of Methods for Transductive Transfer Learning," in *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on*, 2007, pp. 77-82.
- [9] J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence, *Dataset Shift in Machine Learning*: The MIT Press, 2009.
- [10] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* Sydney, Australia: Association for Computational Linguistics, 2006.
- [11] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification," in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, 2007, pp. 440-447.
- [12] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, "Learning Bounds for Domain Adaptation," in *Advances in Neural Information Processing Systems*, 2007.
- [13] W. Dai, G.-R. Xue, Q. Yang, and Y. Yu, "Co-clustering based classification for out-of-domain documents," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* San Jose, California, USA: ACM, 2007.
- [14] G.-R. Xue, W. Dai, Q. Yang, and Y. Yu, "Topic-bridged PLSA for cross-domain text classification," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* Singapore, Singapore: ACM, 2008.
- [15] J. Alcalá-Fdez, F. Herrera, F. Márquez, and A. Peregrín, "Increasing fuzzy rules cooperation based on evolutionary adaptive inference systems," *International Journal of Intelligent Systems*, vol. 22, pp. 1035-1064, 2007.
- [16] F. A. Marquez, A. Peregrin, and F. Herrera, "Cooperative Evolutionary Learning of Linguistic Fuzzy Rules and Parametric Aggregation Connectors for Mamdani Fuzzy Systems," *Fuzzy Systems, IEEE Transactions on*, vol. 15, pp. 1162-1178, 2007.
- [17] C.-T. Lin and C. S. G. Lee, *Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems*: Prentice-Hall, Inc., 1996.
- [18] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," in *7th International World Wide Web Conference Australia*, 1998, pp. 161-172.
- [19] W. L. Tung, C. Quek, and P. Cheng, "GenSo-EWS: a novel neural-fuzzy based early warning system for predicting bank failures," *Neural Networks*, vol. 17, pp. 567-587, 2004.
- [20] G. S. Ng, C. Quek, and H. Jiang, "FCMAC-EWS: A bank failure early warning system based on a novel localized pattern learning and semantically associative fuzzy neural network," *Expert Systems with Applications*, vol. 34, pp. 989-1003, 2008.