I.J. Information Engineering and Electronic Business, **2013, 1, 40-49**
Published Online May 2013 in MECS (http://www.mecs-press.org/)

Object Oriented Software Usability Estimate with Adaptive Neuro Fuzzy, Fuzzy Svm

Mohammad Saber Iraji, Reyhane mosaddegh
Department of Computer Engineering and Information Technology, Payame Noor University, I.R. of Iran
Iraji.ms@gmail.com, Reyhane.mosaddegh@yahoo.com

Abstract— In this paper, we present many intelligent models to estimate the usability of object oriented software. In our proposed system, fuzzy svm has less errors and system worked more accurate and appropriative than prior methods.

Index Terms— Usability, object oriented, fuzzy svm, software

I.INTRODUCTION

Today, in order to estimate the quality of software, software industry experience and training data is important.

Usability is regarded as an important quality of successful factor for developing the successful interactive software system. It is also a key quality factor in the development of software applications[1].

The ISO 9241-11 [2] defines usability as “the context which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”. Subsequently, ISO/IEC 9126-1 [3] categorized usability a part stating internal and external software quality, defining it as “the capability of software product to be understood, learned, used and attracted the user under specified conditions ”.

According to E. Chang and T. S Dillon [4], the literature has provided different approaches for usability evaluation which are:

a) Empirical testing [5]
b) Inspection [6]
c) Comparative usability measures [7]
d) Formal complexity-based measures [8]
e) Metrics for usability standards in computing [9]

Research studies determined that many factors influence the usability of the software and these factors contribute into making user performance, user preference, and user interface [5, 7, 9, 10].

Moutinho, Antonio morais in 2008 present a methodology for evaluation the usability of the software for industrial automation using artificial neural networks and consider profile of the software, user profiles, efficiency, effectiveness and user satisfaction as factors affecting reusability.

Sanjay kumar dubey, others present usability evaluation of object oriented software system using fuzzy logic, fuzzy AHP[11].

Software estimation models combining algorithmic models with machine learning approaches, such as neural networks and fuzzy logic, have been viewed with skepticism by the majority of software managers [12]. Briefly, neural networks techniques are based on the principle of learning from historical data, whereas fuzzy logic is a method that used to make rational decisions in an environment of uncertainty and vagueness. However, fuzzy logic alone does not enable learning from the historical database of the software projects. Once the concept of fuzzy logic is incorporated into the neural networks, the result is a Neuro-fuzzy system that combines the advantages of both techniques [13].

However, our proposed Neuro-fuzzy model goes even further: it is a unique combination of neural networks and fuzzy logic. Specifically, we obtained a suite of fuzzy sets to represent human judgment, and used a neural network to learn from a comprehensive historical database of software projects. A Neuro-Fuzzy use case size Points Calibration model that incorporates the learning ability from neural network and the ability to capture human knowledge from fuzzy logic is proposed and further validated in this paper.

The paper is organized in six sections. After introducing Section I, Section II which also introduces the related works of usability estimation. Section II continues with explanations of support vector machine in section III, IV. In Section IV, V is proposed usability estimate with adaptive Neuro fuzzy ,fuzzy svm . It continues with discussions on the architecture of hybrid learning and fuzzy model validation, the error of observations for training data sets. Section VI presents the conclusions of the research. The paper ends with a list of references.

II. SOFTWARE USABILITY ESTIMATE ALGORITHMS

In [1] proposed multiple sub-factors for the different factors of usability. The multiple factors and sub-factors are defined as follows:

i) Effectiveness (A1): It refers to the capability of the software which users achieve specified goals. It contains the following sub-factors:

a) Accuracy: It evaluates whether the system, after implementing aspects, is giving accurate results when used under specified condition.

b) Speed: It evaluates how quickly a task is performed.

c) Consistency: It allows a user to easily generalize his understanding of different modules of a system.

d) Understandability: It describes the capability of software to enable users to understand the appropriateness of software and its use for particular tasks and conditions of use.

e) Quality of Outcome: It evaluates the quality of the interaction understanding or learning of information in the interface.

ii) Efficiency (A2): It refers to the characteristics of the product that gives best results with use of minimum resources. It contains the following sub-factors:

a) Scalability: It is the ability of a system, network, or process, to handle growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth.

b) Operability: It describes the capability of software to enable users to operate and control it.

c) Compatibility: It indicates that a product can work with or is equivalent to another, better-known product.

d) Time Efficiency: It describes the capability of software to provide appropriate responses, processing time and throughput rates when performing its function under required conditions.

e) Resource Efficiency: It describes the capability of software to use appropriate resources in time when the software implements its function in required conditions.

iii) Satisfaction (A3): It refers to the fulfillment of all requirements by the product as specified by the user. It contains the following sub-factors:

a) Preference:-It measures the satisfaction as an interface using users prefer.

b) Pleasant:-It indicates the capability of the software component to be attractive to the user.

c) Ease of use: It refers to the capability of the software that it can be used easily by the user.

iv) Learn ability (A4): It is the capability of the software product to enable the user to learn its application. It contains the following sub-factors:

a) Wizard or User Guidance: It act as the guide which help the user to understand about the software.

b) Memo ability: It refers to the capability of the software that it is easy to remember.

c) Simplicity: It indicates the capability of the software component to be simple to the user.

d) Self-Descriptiveness: It implies to the useful explanation of the software program design.

We consider factors of Effectiveness (A1), Efficiency (A2), Satisfaction (A3), Learn ability (A4) and use adaptive fuzzy neural network and svm for evaluation of object oriented software.

Our Neuro-Fuzzy and svm approach presented in this research is a novel combination of the above three approaches. It defines a suite of fuzzy sets to represent human judgment Exactly, and uses neural network to learn the calibrated parameters from the historical project database. The equation from statistical analysis is fed into neural network learning . The calibrated parameters from neural network are then utilized in fuzzy sets and the users can specify the upper and lower bounds from their human judgment.

III. SVM

The Support Vector Machines (SVMs), developed by Vapnik and co-workers, have been very successful in pattern classification and function estimation problems for crisp data. A comprehensive tutorial on SVM classifier has been published by Burges . In regression and time series prediction applications, excellent performances were also obtained . Lin et al. and Huang et al first introduced the use of fuzzy set theory for SVM classification problems. Whereas Chiang et al. applied the SVM theory for the fuzzy rules based modeling [14]

Given a training set of instance-label pairs (xi, yi), i = 1,…, l where xi ϵ Rn and y ϵ{1,-1}l, the support vector machines (SVM) (Boser et al., 1992; Cortes and Vapnik, 1995) require the solution of the following optimization problem:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi i$$

subject to

$$yi(wT\Phi(xi) + b) \geq 1 \_\xi i; \quad (1)$$

$$\xi i \geq 0:$$

Here training vectors xi are mapped into a higher (maybe infinite) dimensional space by the function ø. SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term.Furthermore, K(xi,xj) ≡ ø(xi)T ø(xj) is called the kernel function.Though new kernels are being proposed by researchers, beginners may find in SVM books the following four basic kernels:

linear: K(xi,xj) = xi T xj .

Polynomial: K(xi,xj) = (γxiTxj + r)d, γ > 0.

Radial basis function(RBF):K(xi,xj) = exp(-γ||xi - xj||²),γ > 0.

Sigmoid: K(xi,xj) = tanh(γxi T xj + r).

Here, γ , r, and d are kernel parameters[15].

3.1 Linear case[16]

Consider a set of l vectors {xi}, xi € Rn , 1 ≤ i ≤ l, representing input samples and the set of labels {yi}, yi € { ±1}, that divide input samples into two classes, positive and negative. An indicator vector y is defined as:

yi= 1 if xi in class 1 and

yi = -1 if xi in class 2 (2)

and decision function is:

d (x) = sign(wT x +b) (3)

where w is the weight vector, and b is the bias.

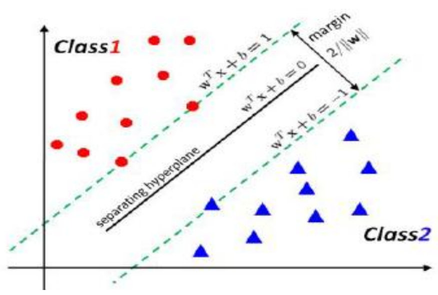


Figure 1.Linear SVM using maximum margin[16]

If the two classes are linearly separable, there exists a separating hyperplane (*w, b*) defining the function

$$f(x) = < w \cdot x > +b, \quad (4)$$

and *sgn*(*f*(*x*)) shows on which side of the hyperplane *x* rests, in other words - the class of *x*. Vector *w* of the separating hyperplane can be expressed as a linear combination of *xi* (often called a dual representation of *w*) with weights α_i:

$$w = \Sigma_{1\le i\le l}\alpha_i y_i x_i. \quad (5)$$

The dual representation of the decision function *f*(*x*) is then:

$$f(x) = \Sigma_{1\le i\le l}\, \alpha_i yi < x_i \cdot x > +b. \quad (6)$$

Training a linear SVM means finding the *embedding strengths*{α_i} and offset *b* such that hyperplane (*w*, *b*) separates positive samples from negatives ones with a maximal margin. Notice that not all input vectors {x_i} might be used in the dual representation of *w*; those vectors x_i that have weight $\alpha_i > 0$ and form *w* are called *support vectors*.

3.2 Non-linear case

In real-life problems it is rarely the case that positive and negative samples are linearly separable. Non-linear support vector classifiers map input space *X* into a feature space *F* via a usually non-linear map

Ø: $X \rightarrow F$, $x \rightarrow Ø(x)$ and solve the linear separation problem in the feature space by finding weights α_i of the dual expression of the separating hyperplane's vector *w*:

$$w = \Sigma_{1\le i\le l}\, \alpha_i y_i \Phi(x_i), \quad (7)$$

while the decision function *f*(*x*) takes the form

$$f(x) = \Sigma_{1\le i\le l}\, \alpha_i y_i < \Phi(x_i) \cdot \Phi(x) > +b. \quad (8)$$

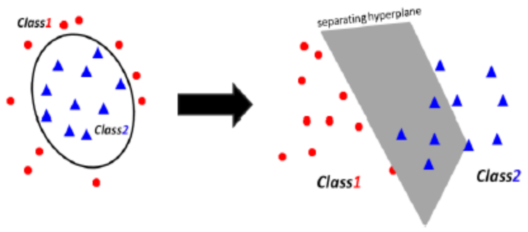


Figure 2.Nonlinear SVM:Mapping data from input space (left), into the higher dimentional feature space (right)[16]

Usually *F* is a high-dimensional space where images of training samples are highly separable, but working directly in such a space would be computationally expensive. However we can choose a space *F* which is induced by kernel *K*, defined by a kernel function *K*(*x, y*) that computes the dot product in *F*, $K(x, y) = < \Phi(x) \cdot \Phi(y) >$. The decision function (6) can then be computed byjust using the kernel function and it can also be shown that finding the maximum margin separating hyperplaneis equivalent to solving the following optimization problem:

$\max[\Sigma 1\le i\le l\alpha_i - (1/2)\, \Sigma 1\le i\le l\alpha_i\alpha_j y_i y_j K(x_i,x_j)]$,subjected to

$$0 \le \alpha_i \le C,\ 1\le i \le l,\ \Sigma 1\le i\le l\alpha_i y_i = 0, \quad (9)$$

Where positive *C* is a parameter showing the trade-off between margin maximization and training error minimization. Thus knowing the kernel function *K* we avoid working directly in feature space *F*. After solving (7), offset *b* can be chosen so that the margins between the hyperplane and the two classes of sample images are equal. We then have our decision function

$$sgn(f(x)) = sgn[\Sigma 1\le i\le l a_i y_i K(x_i,x) + b]\ . \quad (10)$$

Commonly used kernels include polynomial kernels $=K(x, y) = (x + y)^d$ and the Gaussian kernel $K(x, y) = exp(-||x-y||^2/\sigma^2)$.

IV. Fuzzy svm[19]

4.1 Fuzzy linear svm

4.1.1 Conventional Pairwise Classification

Since the extension to nonlinear decision functions is straightforward,to simplify discussions,we consider linear decision functions. Let the decision function for class *i* against class *j*, with the maximum margin,be

$$D_{ij}(X)=W_{ij}^t X+b_{ij}, \quad (11)$$

where w_{ij} is the *m*-dimensional vector, b_{ij} is a scalar, and

$$D_{ij}(x) = -D_{ji}(x).$$

For the input vector x we calculate

$$D_i(X) = \sum_{j\neq i, j=1} sign\left(D_{ij}(X)\right), \quad (12)$$

Where

$$\text{Sign}(x) = \begin{cases}1 & x > 0\\ 0 & x \ge 0\end{cases} \quad (13)$$

and classify **x** into the class

$$\arg\ \max_{i=1,\ldots,n} D_i(X) \quad (14)$$

If (14) is satisfied for plural *i*'s, **x** is unclassifiable. In the shaded region in Figure. 3, $D_i(\mathbf{x}) = 1(i = 1, 2, \text{and } 3)$. Thus, the shaded region is unclassifiable.

4.1.2 Introduction of Membership Functions[19]

Similar to the one-to-($n - 1$) formulation,we introduce the membership functions to resolve unclassifiable regions while realizing the same classification results with that of the conventional pairwise classification. To do this, for the

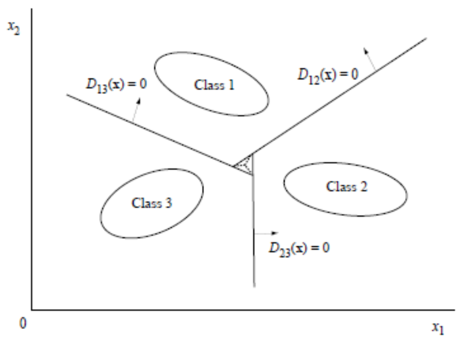


Figure 3: Extended generalization regions

Optimal separating hyperplane $D_{ij}(\mathbf{x}) = 0 (i \neq j)$ we define one-dimensional membership functions $m_{ij}(\mathbf{x})$ on the directions orthogonal to $D_{ij}(\mathbf{x}) = 0$ as follows:

$$m_{ij}(X) = \begin{cases} 1 \text{ for } D_{ij}(x) \geq 1 \\ d_{ij}(X) \quad \text{otherwise.} \end{cases} \tag{15}$$

Using $m_{ij}(\mathbf{x})$ $(j \neq i, j = 1, \dots, n)$, we define the class i membership function of $\mathbf{x}$ using the minimum operator:

$$m_i(\mathbf{x}) = \min_{j=1,\dots,n} m_{ij}(X). \tag{16}$$

Equation (11) is equivalent to

$$M_i(X) = \min\,(1, \min_{j \neq i, j=1,\dots,n} D_{ij}(X) \qquad). \tag{17}$$

The shape of the membership function is shown to be a truncated polyhedral pyramid. Since $m_i(\mathbf{x}) = 1$ holds for only one class, (17) reduces to

$$m_i(X) = \min_{j \neq i, j=1,\dots,n} \; D_{ij}(\mathbf{x}). \tag{18}$$

Now an unknown datum $\mathbf{x}$ is classified into the class

$$\arg\max_{i=1,\dots,n} \; m_i(\mathbf{x}). \tag{19}$$

Thus, the unclassified region is resolved as shown in Figure 3.

4.2 Fuzzy SVM Regression(*Non_linear*)[14]

In many real-world applications, available information is often uncertain, imprecise and incomplete and thus, usually is represented by fuzzy sets or a generalization of interval data. For handling those fuzzy data, fuzzy regression analysis is an important tool and has been successfully applied in different applications. In this section, we applied the fuzzy set theory for the SVM regression model. The fuzzifized parameters within SVM regression will make it more elastic.

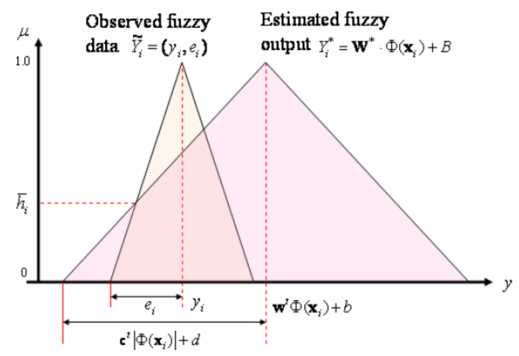
First, we deal with fuzzy desired output in the regression task. The given output data, denoted by $\tilde{Y}_i=(y_i,e_i)$, are symmetric triangular fuzzy numbers,where y_i is a center and e_i is a width. The membership function of $\tilde{Y}_i$ is given by

$$\mu_{\tilde{Y}_i}(Y) = 1 - \frac{|y-yi|}{ei}. \tag{20}$$

Then, for handling those fuzzifized training data, the components in weight vector and bias term using in the SVM regression model are also set to be fuzzy numbers.Give fuzzy weight vector W={w, c} and fuzzy bias term $B=\{b, d\}$, W={w, c} is the fuzzy weight vector, where each components within it are fuzzy numbers. It was denoted in the vector form of w=$[w_1, \dots, w_n]^t$, and c= $[c_1, \dots, c_n]^t$, which means “approximation w”, described by the center w and the width c. Similarly, $B=\{b,d\}$ is the fuzzy bias term, which means “approximation b”, described by the center b and the width d. The fuzzy parameters studied in this work are restricted to a class of “triangular” membership functions. The fuzzy function

$$Y = W_1\phi(X)_1 + W_n\phi(X)_n + B = W.\phi(X) + B, \tag{21}$$

is defined by the following membership function:



μ
1.0
Observed fuzzy data $\tilde{Y}_i = (y_i, e_i)$
Estimated fuzzy output $Y_i^* = \mathbf{W}^* \cdot \Phi(\mathbf{x}_i) + B$
$\bar{h}_i$
0
y
e_i
y_i
$\mathbf{w}^t\Phi(\mathbf{x}_i)+b$
$\mathbf{c}^t|\Phi(\mathbf{x}_i)|+d$

Figure. 4. Degree of fitting of Y_i^* to given fuzzy data $\tilde{Y}_i$

where $|\phi(x)|=[|\phi(x)_1|, \dots, [|\phi(x)_n|]^t$, n is the dimension of feature space, and $\mu_Y(y)$. 0 when $c^t\,|\,\Phi(x)\,|.\,d \leq |y\,(w^t\Phi(x).\,b)|$. To formulate a fuzzy regression model, the followings are assumed to hold.

(1)The data in the feature space can be represented by a fuzzy linear model:$Y_i^*=W^*.\,\Phi(xi\,)\,B^*$. Given xi ,Y_i^* can be obtained as

$$\mu^*_{Y_i}(y) = 1 - \frac{|y-(W^t\,\phi\,(X_i)+b)|}{c^t|\phi\,(X_i)|+d}$$

(2) The degree of the fitting of the estimated fuzzy linear model $Y_i^* = W^*\,\Phi(xi\,)\,B^*$ to the given data $\tilde{Y}I = (yi\,,\,ei)$ is measured by the following index $\bar{h}_i$ which maximizes h subject to $\tilde{Y}_i^h \subset Y_i^{*h}$ where

$$\tilde{Y}_i^h = \{y \mid \mu_{\tilde{Y}_i}(y) \geq h\}$$

$$Y_i^{*h} = \{y \mid \mu_{Y_i^*}(y) \geq h\}$$

which are h-level sets. This index $\bar{h}_i$ is illustrated in Figure. 4. The key is that the observed fuzzy data, adjusted for the h-certain factor, is contained within the estimated fuzzy output, adjusted for the h-certain factor. The degree of fitting of the linear model to all data $\tilde{Y}_1, \dots, \tilde{Y}_N$ is defined by $\min_j |\bar{h}_j|$.

(3) The vagueness of the fuzzy linear model is defined by $\sum_j c_j + d$.

(4) The capacity control term of the fuzzy regression model is defined by the Euclidean norm ||W||2.

The regression problem is explained as obtaining the fuzzy weight vector W* ={w, c} and the fuzzy bias term B* ={b, d}, such that the fitting degree between estimated output $\mu_{Y_i^*}$ (y) and desired output $\mu_{\tilde{Y}_i}(y)$ is more than a given constant H for all i, where H is chosen as the degree of the fitting of the fuzzy linear model by the decision-maker. The value $\bar{h}_i$ can be obtained as

$$\bar{h}\text{i} = 1\text{-} \frac{|y_i-(w^t\ \Phi(x_i)+b)|}{(c^t\ |\phi(x_i)|+d\}-e_i}, \tag{22}$$

which is equal to the early work by Tanaka . Our regression task here is therefore to minimize

$$J = \frac{1}{2}\|W\|^2 + C(c_j + d) \tag{23}$$

subject to $\bar{h}_i \geq H$ for all i=1,..,N, where $\|W\|^2$ is the term which characterizes the model complexity, minimize of $\|W\|^2$ can be understood in the context of regularization operators. $\Sigma c_j\ d$ is the term which characterizes the vagueness of the model. The more vagueness in the fuzzy linear regression model means the more inexactness in the regression result. *C* is a trade off parameter chosen by the decision-maker. The value of *H* determines the low bound for the degree of the fitting of the fuzzy linear model, and $\bar{h}_i$ is the degree of fitting of the estimated fuzzy linear model Y_i^*= w^* .$\phi(X)$ + B^* to the given fuzzy desired output data $\tilde{Y}_i$ = (y_i,e_i) More specifically, according to Eq. (8), our problem is to find out the fuzzy weight vector W^* ={w, c} and fuzzy bias term *B** ={*b*, *d*}, which are the solution of the following quadratic programming problem:

$$\min_{w,c,b,d} J = \frac{1}{2}\|W\|^2 + C\left(\sum_j c_j + d\right)$$

subject to

$$(w^t\,\Phi(x_i)\ b)\ (1H)\ (c^t|\Phi(x_i)|\ d) \geq\geq y_i\ (1H)e_i$$

$$-(W^i\Phi(X_i)+b)+(1-H)(c^t|\Phi(x_i)|+d)\geq -y_i(\tilde{1}H)e_i$$

for i=1,…,N, $d \geq 0$, and $c\ j \geq 0$, for j=1,…,n.

The SVM regression model seeks a linear function in the feature space that has at most εdeviation from the actually obtained targets yi for all the training data, whereas fuzzy SVM regression model seeks a fuzzy linear function with fuzzy parameters in the feature space that has at last H fitting degree from the fuzzy desired targets Ỹi for all the training data.

4.3 Neuro Fuzzy svm

4.3.1 Neuro Fuzzy svm structure[18]

A basic requirement for a chosen kernel K(x, xi) is to satisfy Mercer's theorem . In order to be able to be analyzed in a series with positive coefficients:

$$K(X,X_i) = \sum_{i=1}^{\infty} \lambda_i \phi i\ (X_i)\phi(X_i) \tag{24}$$

Since the kernel has been constructed in the way presented above, the permutation and the ability to be analyzed in a series is a priori secured.However,this can be proved if the following relation stands:

$$\int_a^b \int_a^b K\ (X,x')\psi(x)\Psi(x')dxdx' \geq 0$$

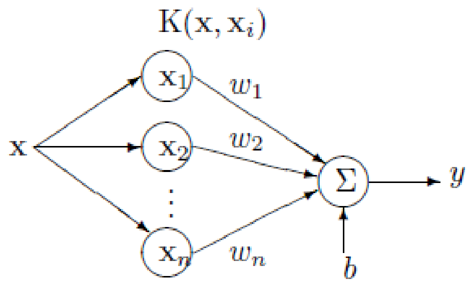


Figure 5: The fuzzy support vector machine network

K is the kernel of the transformation, xi is a support vector,x is an input vector, wi a weight, b the bias and y the output of the network.

For every ψ(x).It is: $\int_a^b \int_a^b K\,(X,X')\psi(x)\psi(x')dxdx'$ = ababi=1nAiX.AX′ψxψx′dxdx′=i=1nIj=I

Since Ij= $\int_a^b \int_a^b A_j\ (x)\,.\,\psi(x)dxA_j(x')\psi(x')dx'$ = $\int_a^b A_j\ (x)\psi(x)dx.\int_a^b A_j\ (x')\psi(x')dx' \geq 0$ It becomes obvious that I ¸ 0 as a sum of non-negative quantities.Thus, Mercer's theorem is satisfied not only from the chosen kernel, but from every kernel that can be written in the form of an inner product.

4.3.2 Learning/Network's Construction

Each neuron in the hidden layer can be viewed as a result of a fuzzy "IF-THEN" rule. This means that knowledge can be easily extracted from a fuzzy SVM network. The network has an input layer, a single hidden layer and an output layer and can be seen on figure 5.

Each support vector represents a fuzzy rule. Let x =[x1, x2,… xm] an input vector, xi = [xi1, xi2,…, xim] a support vector and ψi its response. A fuzzy rule can be stated as:

IF x1 is around xi1 AND x2 is around xi2 AND

: : : xm is around xim THEN x belongs to ψi

V STUDY OBJECTIVES AND METHOD

The objectives of this study are:

- calibration of the use case point factor weight values Exactly to fuzzy further enhanced improvements in the software effort estimation process
- Artificial neural network approach to calibrate the function point weight values provides improvement in the software size estimation process.

Our Neuro-Fuzzy approach presented in this paper is a novel combination of the above three approaches. It obtains a simple equation, defines a suite of fuzzy sets to represent human judgment Exactly, and uses neural network to learn the calibrated parameters from the historical project database. The equation from statistical analysis is fed into neural network learning. The calibrated parameters from neural network are then utilized in fuzzy sets and the users can specify the upper and lower bounds from their human judgment.

The first , the neural network technique is based on the principle of learning from previous data. This neural network is trained with a series of inputs and desired outputs from the training data so as to minimize the prediction error. Once the training is complete and the appropriate weights for the network links are determined, new inputs are presented to the neural network to predict the corresponding estimation of the response variable. The final component of our model, fuzzy logic, is a technique used to make rational decisions in an environment of uncertainty and imprecision. It is rich in its capability to represent the human linguistic ability with the terms of fuzzy set, fuzzy membership function, fuzzy rules, and the fuzzy inference process.

5.1 NEURO FUZZY

It immediately comes to mind, when looking at a neural network, that the activation functions look like fuzzy membership functions. Indeed, an early paper from 1975 treats the extension of the McCulloch-Pitts neuron to a fuzzy neuron (Lee & Lee, 1975; see also Keller & Hunt, 1985).

The one neuron in the output layer, with a rather odd appearance, calculates the weighted average corresponding to the center of defuzzification in the rule base. Backpropagation applies to this network since all layers are differentiable. Two possibilities for learning are apparent. One is to adjust the weights in the output layer, i.e ,all the singletons weight until the error is minimized. The other is to adjust the shape of the membership functions, provided they are parametric. also rule weight can be changed with training data. in Neuro fuzzy model is not necessary output be linear[22].

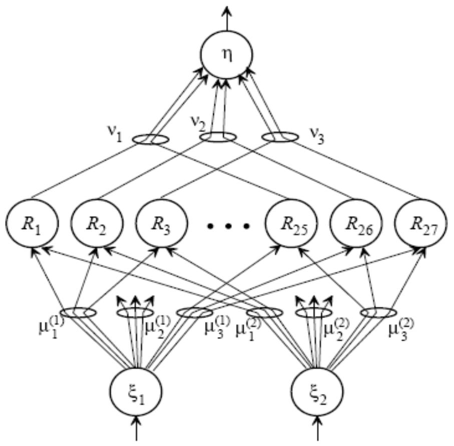


Figure 6. Neuro fuzzy model control system

5.1.1 NEURAL NETWORK

Developing a neural net solution means teaching the net a desired behavior. This is called the learning phase. Either sample data sets or a "teacher" can be used in this step. A teacher is either a mathematical function or a person that rates the quality of the neural net performance. Since neural nets are mostly used for complex applications where no adequate mathematical models exist and rating the performance of a neural net is difficult in most applications, most are trained with sample data(figure 7).

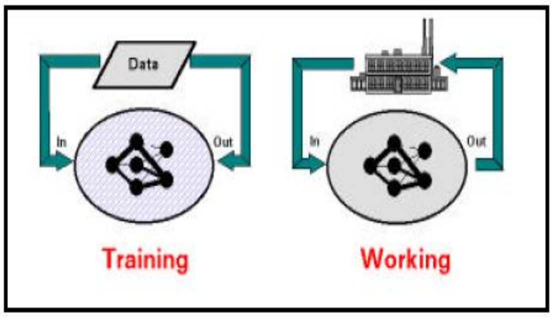


Figure 7. Training and working phase for supervised learning

5.1.1.1 NEURON MODEL

An elementary neuron with R inputs is shown figure 8. Each input is weighted with an appropriate w. The sum of the weighted inputs and the bias forms the input to the transfer function f. Neurons may use any differentiable transfer function f to generate their output.

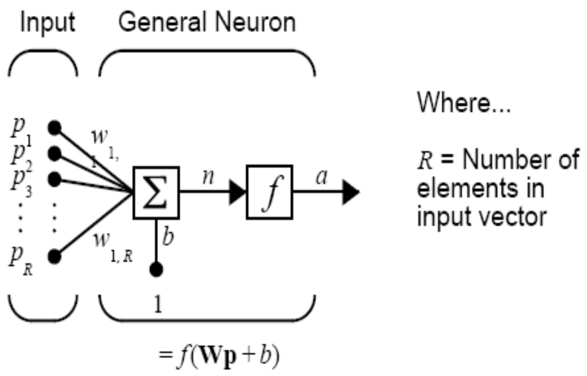


Figure 8.Structure a neuron

Feed forward networks often have one or more hidden layers of sigmoid neurons followed by an output layer of linear neurons. Multiple layers of neurons with nonlinear transfer functions allow the network to learn nonlinear and linear relationships between input and output vectors(figure 9). The linear output layer lets the network produce values outside the range –1 to +1. On the other hand, if you want to constrain the outputs of a network (such as between 0 and 1), then the output layer should use a sigmoid transfer function (such as logsig). This network can be used as a general function approximator. It can approximate any function with a finite number of discontinuities, arbitrarily well, given sufficient neurons in the hidden layer[20].

5.1.1.2 5.1.1.2 NEURAL NETWORK STEP FOR ESTIMATING USE CASE WEIGHT

Back-propagation feed forward neural network approach is used to predict the size of the software using USP approach. A neural network is constructed with 4 inputs and 1 output usability.

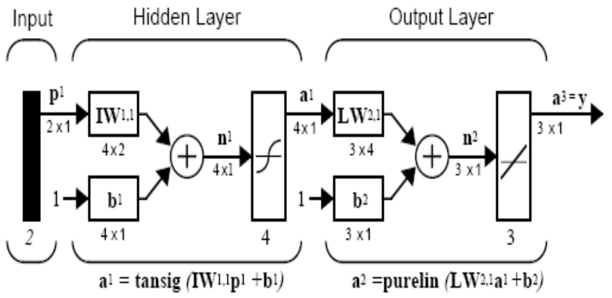


Figure 9. feed forward neural network with two layers

The input nodes represent the distinguishing parameters of usability approach(A1-A4) and the output nodes represent the usability. The network is constructed by using MATLAB. In our experiment the training function we have considered is trapmf, the adaptation learning function considered was backpropa, and the performance function used was Mean Square Error (MSE). The projects considered for this research are taken from the ISBSG data . The input nodes represent the following features of software projects:

1. Effectiveness(A1)
2. Efficiency(A2)
3. Satisfaction(A3)
4. Learn ability(A4)

5.1.1.3 FUZZY LOGIC CALIBRATION STEP

The classification tables are transformed into a continuous classification, this process is called fuzzyfication (a more formal definition to fuzzyfication could be found in [24]). This can be made through the generation of a trapezoidal fuzzy number to each complexity category found on the classification tables. Then, each classification table for Effectiveness, Efficiency, Satisfaction, Learn ability that represented by figure 10.

Membership functions for output actor weight are the triangular type, because these types of membership functions are appropriate to use in preserving the values in the complexity weight matrices. The fuzzy inference process using the Mamdani approach [21] is applied to evaluate use case component weight degree when the linguistic terms, the fuzzy sets, and the fuzzy rules are defined.

Fuzzy Logic Rules:

If A_1=simple and A_2=simple and A_3=simple and A_4=simple then usability=simple.

The antecedent result as a single number implies the consequence using the min (minimum) implication method. Each rule is applied in the implication process and produces one consequence. The aggregation using the max (maximum) method is processed to combine all the consequences from all the rules and gives one fuzzy set as the output. Finally, the output fuzzy set is defuzzified to a crisp single number using the centroid calculation method.

Afterwards, a fuzzy complexity measurement system that takes into account all elements of usability is built after the fuzzy logic system for each usability component is established, as shown in Figure 8. Each usability element is into a Fuzzy Logic System (FLS). The outputs of all five FLS are summed up and become the Neuro fuzzy usability Which is used to calculate usability(figure 11).

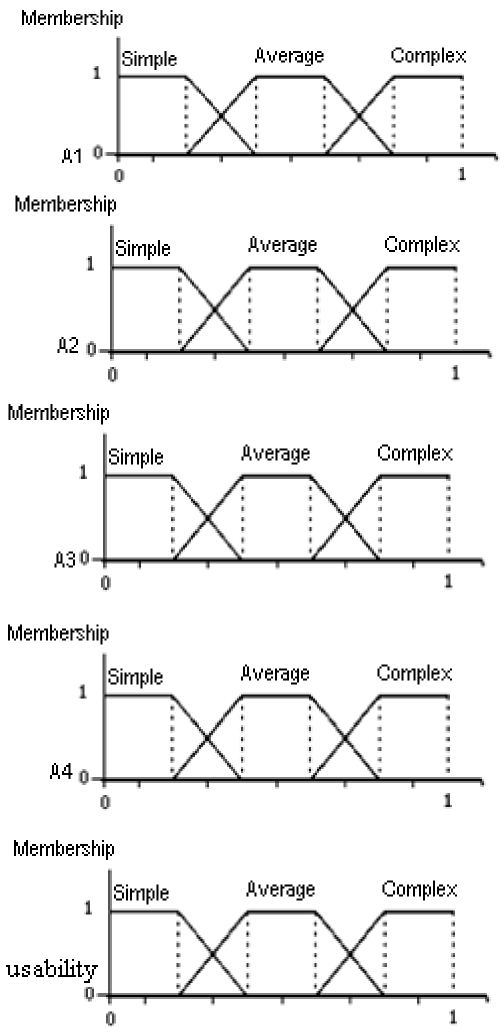


Figure 9.Fuzzy graph for A_1,A_2,A_3,A_4,usability

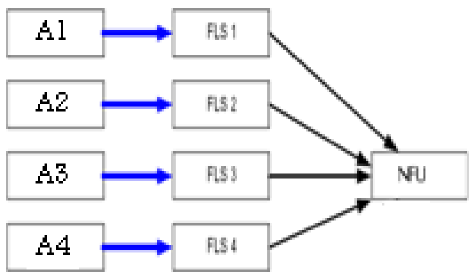


Figure 11: fuzzy inference process of Neuro-fuzzy usability model

5.1.1.3 SUGENO MODEL

A typical rule in a Sugeno fuzzy model has the form If Input 1 = x and Input 2 = y, then Output is z = ax + by + c.

For a zero-order Sugeno model, the output level z is a constant (a=b =0).

The output level z_i of each rule is weighted by the firing strength wi of the rule.For example, for an AND rule with Input 1 = x and Input 2 = y, the firing strength is w_i=AndMethod (F1(x), F2(y)) ;where F1,2 (.) are the membership functions for Inputs 1 and 2. The final output of the system is the weighted average of all rule outputs, computed as:

$$\text{Final output} = \frac{\sum_{i=1}^{N} w_i\, z_i}{\sum_{i=1}^{N} w_i}$$

A Sugeno rule operates as shown in the following diagram(Figure 12).ANFIS (Adaptive Neuro Fuzzy Inference System) is an architecture which is functionally equivalent to a Sugeno type fuzzy rule base (Jang, Sun & Mizutani, 1997; Jang & Sun,1995). Under certain minor constraints the ANFIS architecture is also equivalent to a radial basis function network. Loosely speaking ANFIS is a method for tuning an existing rule base with a learning algorithm based on a collection of training data. This allows the rule base to adapt.

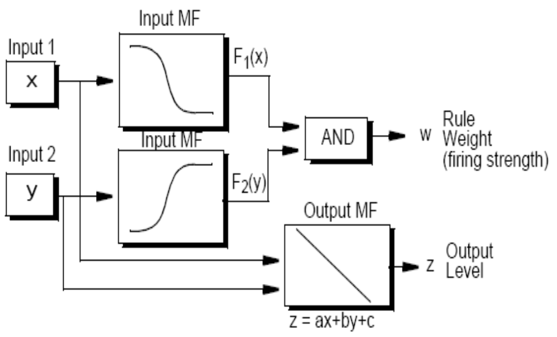


Figure 12. Sugeno model

5.1.1.4 ANFIS

Consider the fuzzy neural network in figure 13. The output of the first layer nodes are the degree of membership of linguistic variables. Typically, in this layer bell–shaped functions are used. Bell-shaped function is shown in Relationship 6. The purpose of learning in this layer is adjusting the parameters of membership function of inputs.

$$f(x) = \exp\left[\frac{-1}{2}\left(\frac{x-a_{i_1}}{b_{i_1}}\right)^2\right] \tag{25}$$

The second layer is 'rules layer'. In this layer, the condition part of rules is measured by usually Min fuzzy logic operator, and the result will be the degree of activity of rule resultant. Learning, in this layer, is the change of the amount of activity of rules resultant, regarding to the 'training data', given to the network. In the third layer, we'll get the linear combination of rules resultant rate, and in order to determine the degree of belonging to a particular category, Sigmund function is used in layer 4 [23].

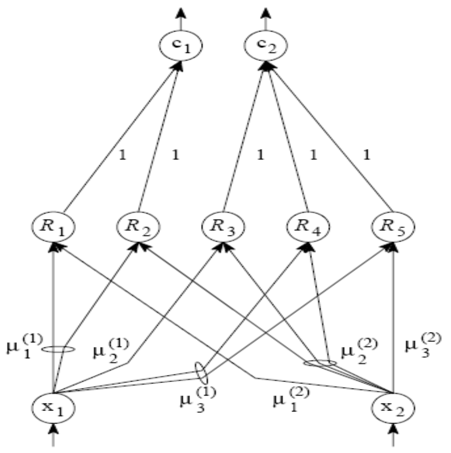


Figure 13.Adaptive Neuro fuzzy Network(anfis)

If a series of training vectors is given to the network in the form of the formula 26:

$$\{(x^k,y^k),k=1,\ldots,K\} \tag{26}$$

Where x^k refers to the K-th input pattern, then we have:

$$y^k=\begin{cases}(1,0) & \text{if } X^k \text{ belongs to class1}\\(0,1) & \text{if } X^k \text{ belongs to class 2}\end{cases} \tag{27}$$

The error function for K pattern can be defined by relationship 28:

$$Predicated\ Usability\ E_k = \frac{1}{2}[(O_1^k - y_1^k)^2 + (O_2^K - y_2^k)^2] \tag{28}$$

Where y^k is desired output, and O^k is computed output.

VI. DISCUSSION AND CONCLUSION

Several methods exist to compare cost estimation models. Each method has its advantages and disadvantages. In this work, The Magnitude of Relative Error (MRE) will be used. MRE for each observation i can be obtained as:

$$MER_i=\frac{|Actualusability_i - Predicatedusability_i|}{Predicatedusability_i} \tag{29}$$

MMRE can be achieved through the summation of MRE over N observations:

$$MMER = \frac{1}{2}\sum_1^N MER_i \tag{30}$$

After training neural network by ISBSG data ,we have applied Our proposed Neuro fuzzy system on seven samples of projects. The results are shown in the table below.

TABLE 1. EVALUATION RESULTS FROM SAMPLE PROJECTS

PROJECT #	Actual normalized usability	Normalized usability point	Normalized usability in Neuro fuzzy (mamdani)	Normalized Usability in Adaptive Neuro fuzzy (sugeno)	SVM	FSVM	MER (U)	MER(NFU)	MER(ANFU)	MER(SVM)	MER(FSVM)
1	1	0.96	0.97	0.98	0.97	0.99	0.04	0.03	0.02	0.03	0.01
2	1	0.87	0.99	0.99	0.9	0.98	0.15	0.01	0.01	0.11	0.02
3	1	0.91	0.86	0.94	0.93	0.99	0.10	0.16	0.06	0.08	0.01
4	1	0.98	0.95	0.97	0.97	0.98	0.02	0.05	0.03	0.03	0.02
5	1	0.99	0.98	0.99	0.99	0.99	0.01	0.02	0.01	0.01	0.01
6	1	0.85	0.87	0.88	0.89	0.92	0.18	0.15	0.14	0.12	0.09
7	1	0.89	0.84	0.9	0.88	0.93	0.12	0.19	0.11	0.14	0.08
MMER							0.31	0.31	0.19	0.26	0.12

As you can see in the table 1, usability estimation with fuzzy support vector machine (fsvm usability), MMER is less and fsvm accuracy further. For future work can be other different types of membership functions, different types of neural network and optimization algorithms like genetic algorithm considered. However, software development is a rapidly growing industry and these calibrated weight values will not reflect tomorrow's software. The advantage of Neural Networks is it has the learning capability to adapt new data. On the other hand, Fuzzy Systems has the capability to handle numerical data and linguistic knowledge simultaneously. In the future, when modern project data is available, the usability weight values will again need to be re-calibrated to reflect the latest software industry trend. The Neuro-fuzzy SVM usability is a framework for calibration and the Neuro-fuzzy usability calibration tool can automate the calibration process when data becomes available.

ACKNOWLEDGMENT

This work received support from the Department of Computer Engineering and Information Technology, Payame Noor University ,I.R. of Iran.

REFERENCES

[1] ACM Human Factors Computing Systems sanjay kumar dubey, Ajay Rana, Arun Sharma ," Usability Evaluation of Object Oriented Software System using Fuzzy Logic Approach", International Journal of Computer Applications (0975 – 8887), Volume 43– No.19, April 2012

[2] International Organization for Standardization. ISO 9241-11, Ergonomic requirements for office work with visual display terminals (VDTs), Part 11: Guidance on usability. Geneva Switzerland: Author, 1998.

[3] International Organization for Standardization International Electrotechnical Commission. ISO/IEC 9126-1, Software Engineering, product Quality, Part 1: Quality Model, Geneva Switzerland: Author,2001.

[4] E. Chang, and T. S. Dillon, ―A Usability-Evaluation Metric Based on a Soft-Computing Approach, IEEE transaction on Systems, man, and Cybernetics—Part A Systems and Humans, Vol. 36, No. 2, March 2006.

[5] R. Molich, and J. Nielsen, ―Heuristic evaluation of user interfaces, in Proc. (CHI), New York, 1990, pp. 249–256.

[6] J. Nielsen, Usability Engineering, Academic Press, 1993.

[7] A. Dillon, and M. Maquire, ―Usability measurement—Its practical value to the computer industry, in Proc. ACM/IFIP Human Factors Computer System (INTERCHI), Amsterdam, The Netherlands, 1993, pp. 145–148.

[8] H. Thimbleby, ―Formulating usability,SIGCHI Bull., Vol. 26, No. 2, Apr. 1994, pp. 59–64.

[9] N. Bevan, and M. Macleod, ―Usability measurement in context, Behav. Inf. Technol., vol. 13, No. 1/2, 1994, pp. 132–145.

[10] J. Levy, and J. Nielsen, ―Measuring usability preference vs. performance, Commun. ACM, Vol. 37, No. 4, Apr. 1994, pp. 66–75.

[11] Sanjay Kumar Dubey, Arpan Mittal,Prof. Dr. Ajay Rana, " Measurement of Object Oriented Software Usability using Fuzzy AHP", International Journal of Computer Science and Telecommunications Volume 3, Issue 5, May 2012.

[12] A. Idri, T.A. Khosgoftaar, A. Abran, Can neural networks be easily interpreted in software cost estimation? in: Proceedings of the IEEE International Conference on Fuzzy Systems (2002) 1162–1167

[13] Wei Xia , Luiz Fernando Capretz , Danny Ho , Faheem Ahmed ," A new calibration for Function Point complexity weights", Information and Software Technology 50 (2008) 670–683

[14] Pei-Yi Hao and Jung-Hsien Chiang," A Fuzzy Model of Support Vector Regression Machine

",International Journal of Fuzzy Systems, Vol. 9, No. 1, March 2007

[15] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin," A Practical Guide to Support Vector Classi_cation", National Taiwan University, Taipei 106 ,Taiwan .http://www.csie.ntu.edu.tw / ~cjlin.Initial version: 2003 Last updated: April 15, 2010

[16] Sangeeta Agrawal1, Rohit Raja2, Sonu Agrawal,"Support Vector Machine for age classification",International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 5, May 2012)

[17] D. Mahmoodi, et al., "FPGA Simulation of Linear and Nonlinear Support Vector Machine,"Journal of Software Engineering and Applications, vol. 5, No.4, pp. 320-328, 2011

[18] Evaggelos Spyrou, Giorgos Stamou, Yannis Avrithis and Stefanos Kollias," FUZZY SUPPORT VECTOR MACHINES FOR IMAGE CLASSIFICATION FUSING MPEG-7 VISUAL DESCRIPTORS", Integration of Knowledge, Semantics and Digital Media Technology, 2005. EWIMT 2005. The 2nd European Workshop on the (Ref. No. 2005/11099)

[19] Shigeo Abe and Takuya Inoue," Fuzzy Support Vector Machines for Multiclass Problems", ESANN'2002 proceedings - European Symposium on Artificial Neural Networks.Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 113-118 Shigeo Abe and Takuya Inoue," Fuzzy Support Vector Machines for Multiclass Problems",ESANN'2002 proceedings - European Symposium on Artificial Neural Networks Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 113-118

[20] Howard Demuth,Mark Beale," Neural Network Toolbox User's Guide",1992 – 2000

[21] E.H. Mamdani, Application of fuzzy logic to approximate reasoning using linguistic synthesis, IEEE Transactions on Computers 26 (12) (1977) 1182–1191.

[22] N.N.,"Fuzzy Logic Benchmarks for MCUs "http:// www. fuzzytech.com/e_ftedbe.htm (1998).

[23] Ashish Ghosh, B. Uma Shankar, Saroj K. Meher ,"A novel approach to neuro-fuzzy classification",Neural Networks, Volume 22, Issue 1, January 2009, Pages 100-109

[24] G. Klir and T. Folger. Fuzzy Sets, Uncertainty and Information. Prentice-Hall, 1998.

Mohammad Saber Iraji received B.Sc in Computer Software engineering from Shomal university, Iran, Amol;M.Sc1 in industrial engineering(system management and productivity) from Iran, Tehran and M.Sc2 in Computer Science . Currently, he is engaged in research and teaching on Computer Graphics, Image Processing, Fuzzy and Artificial Intelligent, Data Mining, Software engineering and he is Faculty Member of Department of Computer Engineering and Information Technology, Payame Noor University, I.R. of Iran.

Reyhane Mosaddegh is a Computer software engineering Graduated from Department of Computer Engineering and Information Technology, Payame Noor University,I.R. of Iran.