Research on Software Effort Estimation Combined with Genetic Algorithm and Support Vector Regression

Jin-Cherng Lin, Chu-Ting Chang and Sheng-Yu Huang
Dept. of Computer Science & Engineering, Tatung University
Taipei 10451, Taiwan
jclin@ttu.edu.tw

Abstract— For software developers, accurately forecasting software effort is very important. In the field of software engineering, it is also a very challenging topic. Miscalculated software effort in the early phase might cause a serious consequence. It not only effects the schedule, but also increases the cost price. It might cause a huge deficit. Because all of the different software development team has it is own way to calculate the software effort, the factors affecting project development are also varies. In order to solve these problems, this paper proposes a model which combines genetic algorithm (GA) with support vector machines (SVM). We can find the best parameter of SVM regression by the proposed model, and make more accurate prediction. During the research, we test and verify our model by using the historical data in COCOMO, Desharnais, Kemerer, and Albrecht. We will show the results by prediction level (PRED) and mean magnitude of relative error (MMRE).

keywords: genetic algorithm, support vector machine, software effort, COCOMO, support vector machine regression.

I. INTRODUTION

Nowadays the quantity demanded of software is very huge. The scale and kind of the software is also varied. Hence, we have to estimate the effort more precise. If we cannot estimate the effort precisely, it might cause huge deficit.

In the past, when talking about the optimization algorithm of software effort estimation, we always mention to the artificial neural network (ANN) and simulated annealing algorithm (SA). These estimated methods can be separated into two kinds. One is like ANN [4] and grey system theory [5], estimated by a model which is built according to historical data. Other one is using COCOMO formula to compute the best parameter by the evaluation algorithm, such as simulated annealing algorithm. But ANN is difficult to implement, and the second method can only use in COCOMO data set. If we change our data set, the result will be inaccurate.

In order to make the prediction results become more precise, we bring a new model calles Genetic Algorithm Support Vector Regression (GASVR). GASVR combines GA with SVR and save their advantages to bulit the best model. The proposed GASVR will be compared with several well known software effort estimation models, such as Fuzzy Grey Relation Analysis (FGRA)[4], Artificial Neural Networks (ANN)[6] and Multiple Linear Regression (MLR) [7]. These comparison are based on three datasets: COCOMO81, Desharnais, Kemerer, and Albrecht[8]. We found our new model have exllcent result. The GASVR is more accurate than other estimation models, and the accuracy improves about 50%.

II. RELATED RESEARCH

2.1 Genetic Algorithm [9][10][11]

Genetic Algorithm (GA) is an algorithm used in researching the optimal solution randomly. It imitates the biological genes. After the evaluation, it will get the best condition. GA shows the genes by chromosome state. Let the chromosomes crossover, mutate, and select. After the competition, these survivals of the fittest are better solutions.

The chromosomes generate by random numbers. It will get its fitness value by the target function. According to the fitness value, the chromosome will crossover, mutate, and select. Crossover will choose two chromosomes randomly, called parent, and exchange their genes of each other. Then it will generate two new chromosomes, called child. Mutation means choose a chromosome randomly, and make it a little different. By crossover and mutation, we produce more chromosomes, and we can get the fitness value of each chromosome. Then we use the roulette wheel selection method to select. The selected chromosomes will use in the next evaluation.

2.2 Software Effort Estimation

In the past, there were many ways to estimate the software effort. We list some estimation method as following:

1. Algorithm Models
2. Expert Judgment
3. Analogy method
4. Top-Down
5. Bottom-Up

In recent decades, there are many new methods to be proposed, especially in intelligent computing area. The following are several famous method used in software effort estimation:

1. Artificial Neural Network (ANN) [6]
   In ANN, the most famous model is back-propagation network (BPN). ANN can be divided into two parts. First part is learning. In this part, it will put some training data into network and try to minimize the energy function by adjusting the weight value of the network. The second part of ANN is recalling. This is the man part of prediction. When we input the data that we need to be

predicted, the network will output the most suitable value.

2. Fuzzy Grey Relational Analysis (FGRA) [5]
   Azzeh proposed FRGA in 2009. FGRA combines grey theory with fuzzy theory. It adds fuzzy theory to grey theory. FGRA reduces to indetermination of the data and make the similar data become more numerical and clearly.

2.3 Support Vector Regression [14][15][16][17]

Support Vector Machine is brought by Vapnik. It bases on Chervonenkis dimension and structural risk minimization principle.

Assume a dataset {(xi, yi), … ,(xi, yi), xi ∈ Rn , yi ∈R, i=1,2,3, … ,l, 。 The target function f (x) in Rn space, we know y = f (x) in R space. When it is a classification problem, yi=-1 or 1. But in a regression problem, it can be a real number, shown as figure 1.

Assume f(x)=w*x+b , f(x) and yi difference are small. Then we can have the most precise prediction. And w is the best plane which is searched by SVR [6].

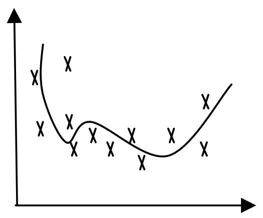


Figure 1. Support Vector Regression

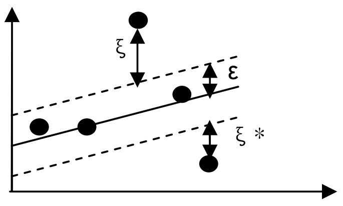


Figure 2. The ε-tube of the regression function

The standard support vector regression join the ε-tube, it makes SVR can solve the regression problem. But most problems have exceptions, so we use ξ to tolerate, shown as figure 2.

In the research, we use linear support vector regression. The regression problem is shown as (1).

$$\begin{aligned}&\min_{w,b,\xi,\xi^*} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{l}\xi_i + C\sum_{i=1}^{l}\xi_i^*\\ &\text{Subject to} \quad w^T\phi(x_i) + b - z_i \le \varepsilon + \xi_i,\\ &\quad z_i - w^T\phi(x_i) - b \le \varepsilon + \xi_i^*\\ &\quad \xi_i, \xi_i^* \ge 0,\ i = 1,\dots,l\end{aligned} \tag{1}$$

$\frac{1}{2}w^T w$: complexity of model

ε: ε-tube

ξ: slack variable

C: penalty parameter

By Lagrange multipliers and optimization, we can get the target function, show as (2)

Finally, we can get the formula of support vector regression, shown as (3)

III. GENETIC ALGORITHM COMBINED WITH SUPPORT VECTOR REGRESSION (GASVR)

In this research, we use genetic algorithm toolbox [5] to combine with support vector regression [6], named GASVR. Finding the good parameters by GA can make SVR predict better.

$$\begin{aligned}&\min_{\alpha,\alpha^*} \quad \frac{1}{2}(\alpha-\alpha^*)^T Q(\alpha-\alpha^*) + \varepsilon\sum_{i=1}^{l}(\alpha_i+\alpha_i^*) + \sum_{i=1}^{l} z_i(\alpha_i-\alpha_i^*)\\ &\text{subject to} \quad \sum_{i=1}^{l}(\alpha_i-\alpha_i^*) = 0,\ 0 \le \alpha_i, \alpha_i^* \le C,\ i = 1,\dots,l\\ &\text{where} \quad Q_{ij} = K(x_i,x_j) \equiv \phi(x_i)^T\phi(x_j)\end{aligned} \tag{2}$$

$$\sum_{i=1}^{l}(-\alpha_i+\alpha_i^*)K(x_i,x) + b \tag{3}$$

3.1Chromosome [9]

We put the chromosome into a Nind * Lind array. Nind means how many chromosome we have, and Lind means how many bits in a chromosome. If there are n variables need to be solved, there are x chromosomes, and y bits in every chromosome. Then we know our array size is [x, n*y], as figure 3.

$$\mathrm{Chrom} = \left[\begin{array}{ccccc} g_{1,1} & g_{1,2} & g_{1,3} & \cdots & g_{1,\mathrm{Lind}}\\ g_{2,1} & g_{2,2} & g_{2,3} & \cdots & g_{2,\mathrm{Lind}}\\ g_{3,1} & g_{3,2} & g_{3,3} & \cdots & g_{3,\mathrm{Lind}}\\ . & . & . & \cdots & .\\ g_{\mathrm{Nind},1} & g_{\mathrm{Nind},2} & g_{\mathrm{Nind},3} & \cdots & g_{\mathrm{Nind},\mathrm{Lind}}\end{array}\right]\begin{array}{l}\text{individual 1}\\ \text{individual 2}\\ \text{individual 3}\\ .\\ \text{individual Nind}\end{array}$$

Figure 3. Array of the chromosome

3.2Crossover [9]

We use the single point crossover. First, we choose two chromosomes randomly, and set the length of the chromosome to 1. And the crossover point is a random value between [1,-1]. We exchange the part of the chromosome which is behind the crossover point, and then it will generate two new child chromosomes.

3.3Mutation [9]

In the research, we use the single point mutation. We choose one bit of the chromosome randomly. If the value of the bit is 0, change the value to 1. If the value of the bit is 1, change the value to 0.

3.4Selection [9]

We use the roulette wheel selection methods. Every chromosome can get its fit value by fit function. If we assume the value of the fit value is the lager the better. So it can occupy the larger space of the roulette. And we choose chromosome by the probability.

3.5Input Data [18]

In GASVR, we need five parameters, they are: -s (svm_type), -t (kernel_type), -c (cost), -g (gamma), and –p (epsilon). We use epsilon-SVR as SVM type, and choose radial basis function as kernel type. We have to input variables as following:

-c(cost)：Punishment parameter.

-g(gamma)：Gamma in the kernel function

-p(epsilon)：ε-tube of epsilon-SVR.

3.6FlowChart

In the research, we represent the three solving variables by chromosome. Then we crossover, mutate, and select the chromosome. After a few generations, we can find the best parameter of SVR. SVR can produce the best model, and give the final prediction result. The flowchart is shown as Figure 4.

IV. EXPERIMENTS AND RESULTS ANALYSIS

4.1 Dataset [8]

In this research, we use COCOMO model to test.

4.2 Data Pre-processing

In COCOMO, the scale type of every column is different, so we do some preprocessing to the data. We preprocess the data before built the model, and expect the result will be more accurate and usable. In this paper, we preprocess the data which the difference is huge. The column we transform are KLOC and Effort. After transformation, the discrete degree of data has a huge reduction. (Fig 5)

4.3 Evaluation Indicators [2][4]

We use two evaluation indicators. The first one is Mean Magnitude of Relative Error (MMRE). The smaller value means the more accurate. MMRE is shown as (4).

$$MMRE = \frac{1}{N}\sum_{i=1}^{N}\frac{|actual\_effort - predicated\_effort|}{actual\_effort} \quad (4)$$

actual_effort: the real value of effort.
predicated_effort: the predict value of effort.
N: the number of project.

i: the ith project。

Other one is PRED. The lager value means the more accurate. PRED is shown as (5).

$$PRED(x) = \frac{k}{N}, k = \#\,(\forall i, MRE_i \le x) \quad (5)$$

x: a percentage of the value.
k: the number of the project which the value of MRE is less than x.
N: the number of project.

$\#(\forall i, MRE_i \le x)$: the total number that the difference is larger then x.

So the larger PRED means the more accurate estimation.

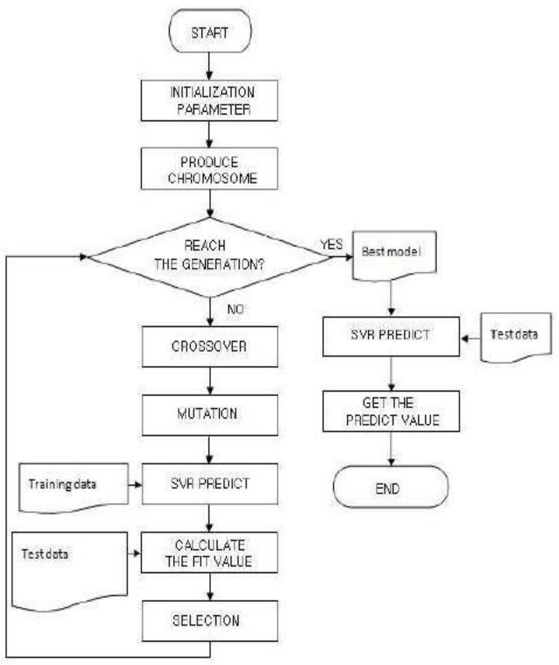


Figure 4. Flow Chart of the System

TABLE I. PREFERENCES

parameter	Generation counts	Population size
Value	200	20
parameter	Crossover rate	Mutation rate
Value	0.7	0.0017

TABLE II. EXPERIMENT RESULT OF COCOMO DATASET

	GASVR	FGRA[4]	ANN[8]
MMRE	0.2085	0.232	0.555
PRED(0.25)	0.825	0.667	0.5

4.4 Experiment Method and Result

We use COCOMO dataset as an example. First, we use (n-1) items of the data to be the training data, and the last one to be the test data. So we have 62 items of data to be the training data each time. In the beginning, we use no.1 to no.62 data to be the training data, as the same time it will also be the test data. Every generation system will get the most suitable parameter and its solution. We can compare the solution with the real data. After few generations, we will get the most precise parameter of the training data, and it will produce a model file. Using the model file estimates the last data. The flow path is shown as figure 6.

Before executing the system, we have to set the parameter of the GASVR. Our setting is shown as Table I.

We compare our result with other methods, as Table II.

As figure 7, our MMRE is much smaller than ANN, and our PRED(0.25) is much higher than other two model

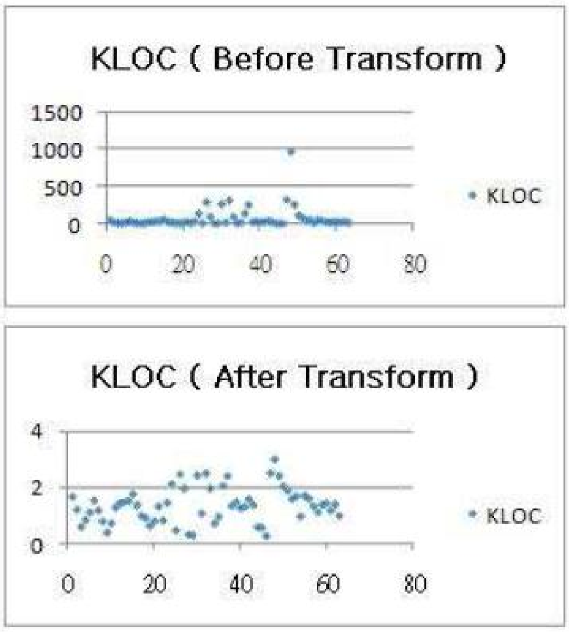
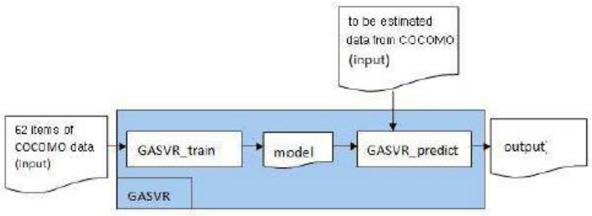


Figure 5. Discrete Degree of Data

V. CONCLUSION

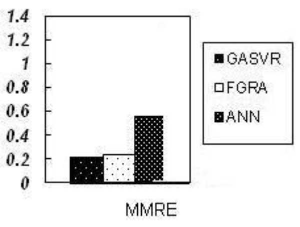
By the experiment result, we know that our new model –-GASVR-- can get more precise solution. GASVR can not only used in software effort estimation. According to our experience, SVR can find out the most suitable model, and GA can find the best parameter quickly and precisely.

In the future, we will improve our model. We will do preprocessing to the data, and combine our model with FUZZY. By these methods, GASVR may get more exact solution.



to be estimated data from COCOMO (input)
62 items of COCOMO data (input)
GASVR_train
model
GASVR_predict
output
GASVR

Figure 6. The Flowchart of the Experiment



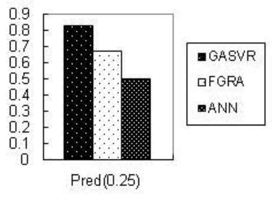


Figure 7. Compare with MMRE

Figure 8. Compare wth PRED(0.25)

REFERENCES

[1] The Standish Group International, http://www.standishgroup.com/ , visited in 2010,12

[2] Boehm, B. W., “software Engineering Economics,” IEEE Transcactions on Software Engineering,Vo1.10, No. 1, 1984, pp. 4-21

[3] Y.F. Li, M. Xie,and T.N. Goh. “A study of the non-linear adjustment for analogy based software cost estimation” Empir Softw Eng 14(6):603-643. DOI 10.1007/s10664-008-9104-6

[4] S.J. Huang and N.H. Chiu, “Applying fuzzy neural network to estimate software development effort,” Applied Intelligence, 2009,pp. 73-83

[5] M Azzeh, D Neagu, PI Cowling, “Fuzzy grey relational analysis for software effort estimation” Empir Softw Eng 15(1) :60-90.DOI 10.1007/s10664-009-9113-0

[6] Haykin, S. “Neural Networks: A Comprehensive Foundation, Prentice Hall”,1999, ISBN 0-13-273350-1.

[7] Hsu CJ, Huang CY, ”Improving Effort Estimation Accuracy by Weighted Grey relational Analysis During Software development”, 2007 14th Asia-Pacific Software Engineering Conference, pp. 534–541.

[8] G. Boetticher, T. Menzies and T. Ostrand, PROMISE Repository of empirical software engineering data http://promisedata.org/ repository, West Virginia University, Department of Computer Science, 2007

[9] Chipperfield, A. J., Fleming, P. J. and Pohlheim, H. P, “A genetic algorithm toolbox for MATLAB,” Proceedings of International Conference on Systems Engineering, Coventry, 6-8 September 1994, pp. 200-207.

[10] Goldberg, D.E. “Genetic Algorithm in Search, Optimization and Machine Learning,” Addision-Wesley, New York, 1989.

[11] Davis, L. “Handbook of Genetic Algorithm.” Van Nostrand Reinhold, New York, 1991.

[12] Kirsopp C, Shepperd MJ, Hart J,”Case-based Reasoning and Software Project Effort Prediction”, Proceedings of the Genetic and Evolutionary Computation Conference 2002, pp. 1367–1374.

[13] Mendes, E., Mosley, N., and Counsell, S. “A replicated assessment of the use of adaptation rules to improve Web cost estimation,” International Symposium on Empirical Software Engineering, pp. 100–109

[14] V.N. Vapnik, “The Nature of Statistical Learning Theory”,New York: Springer, 1995.

[15] V.N. Vapnik, “Statistical learning theory”. New York: Wiley, 1998

[16] B.-W Liu and P.-S Yu “Correction of QPESUMS Radar Rainfall Using Machine Learning Methods”, 2009.

[17] Anna Corazza, Sergio Di Martino, Filomena Ferrucci, Carmine Gravino, and Emilia Mendes. “Investigating the use of Support Vector Regression for web effort estimation” , Empir Softw Eng. IDO 10.1007/s10664-010-9138-4

[18] C.-C. Chang and C.-J. Lin, LIBSVM: a library for support vector machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm/ , visited in 2010.12.