Software Development Cost and Time Forecasting Using a High Performance Artificial Neural Network Model

Iman Attarzadeh and Siew Hock Ow

Department of Software Engineering
Faculty of Computer Science & Information Technology
University of Malaya, 50603 Kuala Lumpur, Malaysia
attarzadeh@siswa.um.edu.my, show@um.edu.my

Abstract. Nowadays, mature software companies are more interested to have a precise estimation of software metrics such as project time, cost, quality, and risk at the early stages of software development process. The ability to precisely estimate project time and costs by project managers is one of the essential tasks in software development activities, and it named software effort estimation. The estimated effort at the early stage of project development process is uncertain, vague, and often the least accurate. It is because that very little information is available at the beginning stage of project. Therefore, a reliable and precise effort estimation model is an ongoing challenge for project managers and software engineers. This research work proposes a novel soft computing model incorporating Constructive Cost Model (COCOMO) to improve the precision of software time and cost estimation. The proposed artificial neural network model has good generalisation, adaption capability, and it can be interpreted and validated by software engineers. The experimental results show that applying the desirable features of artificial neural networks on the algorithmic estimation model improves the accuracy of time and cost estimation and estimated effort can be very close to the actual effort.

Keywords: Software Engineering, Software Project Management, Software Cost Estimation Models, COCOMO Model, Soft Computing Techniques, Artificial Neural Networks.

1 Introduction

Accurate and consistent software development effort prediction in the early stage of development process is one of the critical tasks in software project management. Project managers use effort estimation to make on-time and better managerial decisions during project development life cycle and especially for determination of project details, allocation of project resources, project tasks, schedule controlling, and process monitoring. In software development process, the effort directly related to software schedule, cost and manpower factors that are critical and important for any project. The software development effort estimation is counted as a very complex process because of essential project factors such as development environments, platform factors, human factors, product factors, customers’ needs, and finally the
difficulty of managing such large projects. During last decades, the governments and many mature organisations invest on software development to achieve their purpose.

Therefore, the accurate estimation of project time, cost, and staffing are needed to effectively plan, monitor, control and assess software development companies and project managers. The effort estimation in software engineering is based on two large methods: algorithmic methods and non-algorithmic methods. Algorithmic methods carry a mathematical formula that is inferred from regression model of historical data and project attributes. Constructive Cost Model (COCOMO) [1, 2] and Function Points [3] (FP) are two well-known methods of this category. Non-algorithmic methods [4, 5, 6], usually, are based on heretical projects information and comparing new project activities to past projects, then make estimation on the new project tasks. Expert judgment and analogy-based estimation [5] are two samples of non-algorithmic methods. This research work intends to use the soft computing approach, artificial neural networks, to propose a novel software effort estimation model incorporating constructive cost model to improve the precision of software effort estimation.

2 Related Work

Wittig and his colleagues proposed a simple neural network for software cost estimation. The purpose of that research was to examine the performance of back-propagation training algorithm. First, they used a metric model, (SPQR/20²), to generate adequate data for the experiment. Second, they used a set of actual experiments from developed past software. In both methods, Function Points (FPs) method used as the measurement method of the input parameter, Size, and the development hours used as the unit of system output, Effort [7]. The experiments results in their research work show the ability of neural networks to make better estimation. Samson et al. used another mathematical model, Cerebellar Model Arithmetic Computer (CMAC). Then applied proposed neural networks architecture on the CMAC to make effort estimation based on software code size. The CMAC model proposed by Albus [8] and it is an approximation perceptron function. The established model based on neural networks was trained on COCOMO dataset in order to estimate software development effort from size of software. Also, they used linear regression techniques in same manner to compare the acquired data. The results of the proposed prediction model performed better than linear regression on the same data set. In other research, Boetticher used more than 33,000 different experiments data, collected from separate software companies, to examine the proposed neural network [9].

He used different software metrics such as size, complexity, objects, and vocabulary to estimate software effort using neural networks. Boetticher in another research used bottom-up approach to apply the past experiments on the two separate neural networks. The bottom-up approach uses data collected from software products rather than software projects. In that model, the Source Lines of Code (SLOC) metric used as the input of neural network model to estimate the project effort. Karunanithi et al., also, proposed a cost estimation model based on artificial neural networks [10]. The established neural network was able to generalise from trained datasets. The