Analyzing Software Development Effort Using Software Computing Techniques Based on UCP

Dr S Srinivasa Rao, D Meghamala, B Satwika, G Priyanka

Abstract—In the development of any software project by using SDLC it is an efficient task to know the software effort of that project. Software effort can be known by the key measure of software size. UCP (Use case points) is one of the metrics to calculate the software effort and it is rapidly growing because of the popularity of its methodologies in the software industry. The UCP contains the disadvantage of EF changing from one organization to another. This paper mainly focused to solve the drawback of UCP by using some soft-computing techniques like GRNN and Naïve Bayes. The results suggest prediction of the instance of effort before the development of the application good and in this it has been found the accuracy of the soft computing techniques by taking the dataset of UCP attributes as inputs.

Index Terms— UCP, Soft computing, Environmental Factor, Technical Complexity, Software Effort.

1 INTRODUCTION

In the development of the software product knowing and estimating the size of the product is an important key measure in the needs and analysis phase itself. Use case points (UCP) is one of the key metrics to calculate the software effort by the measurement of software size. This method was first introduced by Karner[1] in the year in 1993. Karner has implemented in such a way to recognize the use cases in the product. One has to know how to identify the simple, average and complex types of use cases in the given product to estimate and they should always aware of the UML diagram to produce the different forms of use cases. Defining the use cases simple, average and complex in every situation is a bit difficult to do so, and the calculation of the Use case points also contains a drawback of Environmental factor and Technical complexity which differ from one organization to another.

The number of staff, the motivation level of the team, experience which are changing from one and another. There is much more increase in the popularity of UCP due to its methodologies and easy calculation rather it also has some disadvantages to be solved. Previously we do not have the predicting knowledge.

Hence now we have the techniques to calculate the effort before the development of software products by using soft computing techniques. This paper has been focused to know the values and accuracy of the soft computing techniques and also the nature of UCP in from different authors and researchers who do some work to predict and improve the metric UCP.

2 OUTLINE OF USE CASE POINTS

To calculate the software size of the product UCP is considered as the measurement kit to calculate. The calculation of UCP is always depends on the four key attributes. These key attributes are like UUCW, UAW, TCF, and EF. Karner has produced a metric and the sequence steps to calculate the UCP. Each attribute is calculated separately UUCW is defined as the Un-adjusted use case weight means the sum of weights of the use cases. The use cases divided into three factors like simple, average and complex according to weights complexity. The following is the formula to calculate the Un-adjusted use case weight.

\[ UUCW = \sum awi^\text{usecase} \]

Later on, the second step is to calculate the Unadjusted Actors weight. Unadjusted actors weight is defined by the sum of the number of actors by finding their weights. There is one formula to calculate the UAW is as follows.

\[ UAW = \sum awi^\text{actors} \]

TCF is called the technical complexity and also the Environmental Factor (EF) is calculated by having dividing according to the complexity factors. These both are calculated according to the defined steps as follows.

\[ TCF = 0.6 + (0.01 \sum (Ti^{*}Fwi)) \]

\[ EF = 1.4 - (0.03 \sum (Ei^{*}ewi)) \]
3 SOFT COMPUTING TECHNIQUES

Soft computing gives out representation and answers to real-time problems. The main thing about soft computing is the human mind. In the 1980s soft computing was developed and techniques were introduced. It’s one of the vital techniques used in engineering. The techniques are well used in many industrial applications within low cost and high digital performance. And this technique is used in medical applications by using a genetic algorithm. When we solve any problem, we find an accurate solution to calculate and estimate any problem. But we do not have any information regarding calculations that time we make exact calculations to find an accurate solution. This type of problem-solving is called soft computing.

3.1 NAÏVE BAYES CLASSIFIER:
In Machine learning, the Naïve Bayes classifier is one of the commonly used simple. The classifier is represented by a very simple Bayesian network. It is very popular in the next classification, and there is a traditional solution for problems. Naïve Bayes is one of the parts of a supervised learning algorithm.

In computer science, naïve Bayes is also under a variety of names like Simple Bayes and independence Bayes. In indecision rule, all the reference of Bayes theorem is used, but it not follows a Bayesian method.

\[
P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}
\]

Fig 1 Naïve Bayes Formula

The word Bayes is added to the algorithm because it is taken form Bayes Theorem that is a conditional probability.

3.2 GENERAL REGRESSION NEURAL NETWORK:
The GRNN[8] is first introduced by Specht in 1991. Its related work is based on a neural network classifier. It is a four-layer network used in the regression task. The layer which comes first is called as load layer which gives some process line. The pattern sheet is a second surface that forms the basic covering and efficient for design instruction. The summation layer is a third layer that consists of pair nerve cell (ordinal and total summation). The fourth layer is dividing the value of sum somatic cell on the worth of the statistic cell. This network don’t require a repetitive presentation algorithm.

Fig 2 General Regression Neural Network

In GRNN there is no need for back propagation. It uses the Gaussian function since it is high accuracy and estimation. It can handle noises in the input.

4 THEORETICAL BACKGROUND

In the last decade, whenever the task is to do, it is complex to estimate time and cost to do the work. To estimate the doing of work and the cost initially Functional Points have been introduced. Functional Points (FP) is the specific metric only used to estimate the schedule to complete a particular event. But by considering this FP there is a drawback of it is not evaluated for Objectory models and is not at all secure in the way. In the year 1993, the person named as Karner[1] proposed one measurement tool in the sense metric for the calculation. UCP is the key metric proposed by the Karner[1] for the object-oriented programming code. Use case points rather expanded to the global software development[2] by Mohammad Azzed to fulfill the need for the nursing expanding request among monstrous IT companies. Even though the requirement is fulfilled but there is a problem of inefficient of helping nature to the world PC code improvement. Some factors cannot implement the need of nursing people. There is a failure and some investigations should be made to reach the goal of implementing at the global level. A metric of calculating the size of the program code is again updated by the prediction techniques like General regression[4], Random forest and also by the fuzzy model[5]. In this paper there are some methods also introduced to predict the software effort by four techniques like MLP, GRNN, RBFNN, and CCNN. These soft-computing techniques have developed in the estimation and prediction of the numbers of persons for the product to complete. And the comparative results, as well as the accuracy and stability, also were determined and future scope is also exhibited by this paper.
5 Methodology

We have compared the results and accuracy of the prediction of the two algorithms Naive Bayes and GRNN. GRNN has been explained in the problem itself it is the efficient one and making the comparison of GRNN with Naïve Bayes. The dataset 40 records have been considered with attributes like UUCW, UAW, TCF, and EF.

### TABLE 1

**DATASET CONSISTING OF 40 RECORDS**

<table>
<thead>
<tr>
<th>UUCW</th>
<th>UAW</th>
<th>TC</th>
<th>EF</th>
<th>UCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>460</td>
<td>60</td>
<td>1.02</td>
<td>0.32</td>
<td>9008.64</td>
</tr>
<tr>
<td>550</td>
<td>30</td>
<td>20.1</td>
<td>0.62</td>
<td>205623</td>
</tr>
<tr>
<td>145</td>
<td>20</td>
<td>3.06</td>
<td>0.23</td>
<td>2041.02</td>
</tr>
<tr>
<td>360</td>
<td>50</td>
<td>5.04</td>
<td>0.65</td>
<td>58968</td>
</tr>
<tr>
<td>960</td>
<td>60</td>
<td>2.1</td>
<td>0.12</td>
<td>14515.2</td>
</tr>
<tr>
<td>750</td>
<td>40</td>
<td>6.1</td>
<td>0.24</td>
<td>43920</td>
</tr>
<tr>
<td>450</td>
<td>90</td>
<td>6.3</td>
<td>0.42</td>
<td>107163</td>
</tr>
<tr>
<td>550</td>
<td>50</td>
<td>5.1</td>
<td>0.35</td>
<td>49087.5</td>
</tr>
<tr>
<td>520</td>
<td>70</td>
<td>4.2</td>
<td>0.35</td>
<td>53508</td>
</tr>
<tr>
<td>620</td>
<td>50</td>
<td>3.04</td>
<td>0.93</td>
<td>87643.2</td>
</tr>
<tr>
<td>210</td>
<td>60</td>
<td>6.13</td>
<td>0.82</td>
<td>63335.16</td>
</tr>
<tr>
<td>620</td>
<td>30</td>
<td>5.31</td>
<td>0.12</td>
<td>11851.92</td>
</tr>
<tr>
<td>140</td>
<td>35</td>
<td>2.3</td>
<td>0.35</td>
<td>3944.5</td>
</tr>
<tr>
<td>320</td>
<td>65</td>
<td>6.1</td>
<td>0.24</td>
<td>30451.2</td>
</tr>
<tr>
<td>590</td>
<td>25</td>
<td>7.3</td>
<td>0.42</td>
<td>45223.5</td>
</tr>
<tr>
<td>360</td>
<td>45</td>
<td>6.7</td>
<td>0.63</td>
<td>68380.2</td>
</tr>
<tr>
<td>340</td>
<td>25</td>
<td>4.8</td>
<td>0.12</td>
<td>4896</td>
</tr>
<tr>
<td>420</td>
<td>35</td>
<td>5.08</td>
<td>0.24</td>
<td>17922.24</td>
</tr>
<tr>
<td>430</td>
<td>65</td>
<td>0.25</td>
<td>0.21</td>
<td>1467.375</td>
</tr>
<tr>
<td>310</td>
<td>60</td>
<td>1.3</td>
<td>0.36</td>
<td>8704.8</td>
</tr>
<tr>
<td>360</td>
<td>40</td>
<td>1.65</td>
<td>0.93</td>
<td>22096.8</td>
</tr>
<tr>
<td>320</td>
<td>90</td>
<td>1.48</td>
<td>0.37</td>
<td>15770.88</td>
</tr>
<tr>
<td>560</td>
<td>40</td>
<td>6.2</td>
<td>0.86</td>
<td>119436.8</td>
</tr>
<tr>
<td>254</td>
<td>50</td>
<td>9.3</td>
<td>0.54</td>
<td>63779.4</td>
</tr>
<tr>
<td>220</td>
<td>10</td>
<td>2.03</td>
<td>0.21</td>
<td>937.86</td>
</tr>
<tr>
<td>320</td>
<td>60</td>
<td>6.3</td>
<td>0.34</td>
<td>41126.4</td>
</tr>
<tr>
<td>620</td>
<td>30</td>
<td>5.6</td>
<td>0.25</td>
<td>26040</td>
</tr>
<tr>
<td>250</td>
<td>15</td>
<td>5.5</td>
<td>0.52</td>
<td>26000</td>
</tr>
<tr>
<td>230</td>
<td>18</td>
<td>3.6</td>
<td>0.43</td>
<td>41162.2</td>
</tr>
<tr>
<td>655</td>
<td>42</td>
<td>6.13</td>
<td>0.28</td>
<td>63353.61</td>
</tr>
<tr>
<td>201</td>
<td>66</td>
<td>7.6</td>
<td>0.24</td>
<td>11815.29</td>
</tr>
<tr>
<td>104</td>
<td>12</td>
<td>3.1</td>
<td>0.12</td>
<td>4869</td>
</tr>
<tr>
<td>950</td>
<td>52</td>
<td>5.8</td>
<td>0.22</td>
<td>1234</td>
</tr>
</tbody>
</table>

5.1 Code for the execution:

```r
> mydata <- read.delim(file.choose(), sep = ',', header = TRUE)
> str(mydata)
> dim(mydata)
>
Making the partition of the dataset into testing and training.
In the software R-studio, the dataset has been taken into the variable [mydata] and divided into the testing as well as training by the partition formula in the form of 30% and 70%.

```r
> tindex = sort(sample(nrow(mydata), nrow(mydata) * .7))
> mtraining = mydata[tindex,]
> mtesting = mydata[-tindex]
> dim(mtraining)
> dim(mtesting)
```

Later on, the required Library e1071 has been installed and called for the usage of Naïve Bayes.

```r
> install.packages(e1071)
> library(e1071)
> NB <- naiveBayes(ef ~ ., data = mtraining)
> print(NB)
> summary(NB)
> predNB1 <- predict(NB, mtesting, type = "class")
> summary(predNB1)
> table(mtesting$ef, predNB1)
> plot(predNB1)
```

But there is a problem in the prediction that is meant and says it is not suitable. With the help of the code executed in the software R-Studio which has proved that the Naïve Bayes is suitable for the prediction of the Effort for any software to be developed and it is proven that GRNN is the best for prediction.

### 6 RESULTS

At the phase of loading all 40 records data into the [mydata] variable and printing the dimensions as well as the attributes and structure. Analyzing the algorithms by the executed code says that the comparison of the GRNN algorithm mentioned in the review and the Naïve Bayes algorithm that we have used says it is not suitable for prediction of the effort but GRNN is suitable for this prediction. The results and execution of code have been shown in the following figure. These are the outputs of above Executed code:
> mydata<-read.delim(file.choose(),sep=',',header= TRUE)
> str(mydata)
'data.frame': 39 obs. of 6 variables:
$ wcc  : int  469 550 345 360 960 750 450 550 520 620 ... 
$ waw  : int  60 39 20 50 60 40 90 50 70 50 ... 
$ cc   : num  1.02 2.01 3.06 1.04 2.12 6.1 3.6 3.1 4.2 3.04 ... 
$ ef   : num  0.32 0.62 0.23 0.65 0.12 0.24 0.42 0.35 0.35 0.93 ... 
$ wcp  : num  9009 202023 2642.58968 24525 ... 
$ effort: num  270079 4368990 82211 1766440 435456 ... 
> dim(mydata)
[1] 39 6

FIG 3 OUTPUT OF DIMENSIONS OF MYDATA

Input of Training and Testing

> tindex<-sort(sample(nrow(mydata),nrow(mydata)*.7))
> mtraining<-mydata[tindex,]
> mtesting<-mydata[-tindex]
> dim(mtraining)
[1] 27 6
> dim(mtesting)
[1] 39 1

FIG 4 OUTPUT OF DIMENSIONS OF TRAINING AND TESTING

Installing package library

> install.packages("e1071")
Installing package into C:/Users/user/Documents/R/win-library/3.3
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.3/e1071_1.7-3.zip'
Content type 'application/zip' length 201829 bytes (194 KB)
downloaded 194 KB

package 'e1071' successfully unpacked and MS5 sums checked.
The downloaded library packages are in
C:/Users/user/AppData/Local/R temporayLib/3.3/donwloaded_packages

Warning message:
package 'e1071' was built under x version 3.5.3

FIG 5 OUTPUT OF PACKAGE Installation

Input: By using the formula of Naïve Bayes we are calculating the conditional probabilities. In this we are calling the Naïve Bayes Algorithm.

While predicting the data into the different forms by the algorithm Naïve Bayes which leads to unsuitable for the prediction of effort rather it is suitable for saying the classifying the dataset into a conclusion.
6 CONCLUSION

The estimation of any software size and effort for any software product is a critical task to complete. UCP has played a major role in the estimation of the effort required to complete but later on, there are some drawbacks for the dynamic project estimation and differ from one to another. So these UCP can make to be calculated by using some soft computing techniques which predict the effort in the early stage itself. The analysis and comparative results of the algorithms like GRNN and Naïve Bayes have executed and concluded that Naïve Bayes is not at all suitable for this prediction rather in the review itself GRNN is suitable and efficient in the prediction of the estimating effort.

REFERENCES