ISSN:0975-9646

Ranking SDLC Models using BBA

Aakash Singhal, Aman Agrawal, Jatin Makkar

Students, CSE Department
MAIT, GGSIP University
Delhi, India

Abstract— **A Software Development Life Cycle Model (or SDLC Model) is an abstraction of the period of time that starts when software product is conceived and ends when a product is no longer available for use. Feature selection attempts to discover the most discriminative information in several application domains. Binary Bat Algorithm (BBA: a binary version of the Bat Algorithm) is a new nature-inspired feature selection technique based on the bats behaviour. In this paper, various SDLC Models (already existing ones) have been worked upon to rank all these and find the most accurate model with the help of BBA.**

Keywords— **SDLC, Feature Selection, BBA, Best SDLC Model, Evaluating SDLC, Questionnaire SDLC, Training and Evaluating SDLC Models.**

I. INTRODUCTION

In software engineering, a Software Development Methodology (also known as a Software Development Life Cycle) is a division of software development work into distinct phases (or stages) containing activities with the objective of achieving better planning and management. It is often considered a subset of the Systems Development Life Cycle. A Software Development Life Cycle Model is an abstraction of a software process. It can also be defined as abstraction of the period of time that starts when software product is conceived and ends when a product is no longer available for use.

Common SDLC models include waterfall, prototyping, iterative and incremental development, spiral, rapid application development and various types of agile methodology. Each model has its pros and cons and thus, depending on various factors, a model can be termed appropriate for a software production.

BBA is a new and theoretical algorithm which, as of now, is the hot algorithm in the optimization field. It is a meta-heuristic algorithm, i.e., a higher level procedure aimed to find, generate, or select a heuristic (partial search algorithm) that may provide a adequately good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity.

In this paper, we have used BBA algorithm solely for the purpose of ranking of SDLC models and finding the most accurate SDLC model.

II. BINARY BAT ALGORITHM (BBA)

R. Y. M. Nakamura, L. A. M. Pereira in their paper, [1][2]Binary Bat Algorithm for [3] [5]Feature Selection modified the Bat Algorithm proposed by Yang for the purpose of feature selection. In Bat algorithm, as proposed by Yang, each bat moves in the search space heading to the continuous-valued positions. However, in case of feature selection, the search space is modelled as a - dimensional Boolean lattice, in which the bat travels across the corners of a hypercube. Since the problem is to determine whether a given feature is selected or not, the bat's position is then represented by binary vectors.

They proposed a binary version of the Xang's Bat Algorithm restricting the new bat's position to only binary values (presence/absence) using a sigmoid function:

$$S(v_i^j)= 1/ (1+ \exp(-v_i^j))$$

Based on the equation above, they modified the position of the bat in the search space. The position was restricted to only {0,1} indicating absence or presence of a feature respectively. Each bat is associated with a velocity that indicates the rate at which the feature changes. The confinement of search space to Boolean lattice makes it applicable to real world and thus, allows implemented for real world problems.

Algorithm.1. - FEATURE SELECTION USING BBA

INPUT: Labelled training Z_1 and evaluating set Z_2, population size m, number of features n, number of iterations T, loudness A, pulse emission rate r, ϵ, α and γ values.

OUTPUT: Subset of features F that gives the maximum accuracy over Z_2.

AUXILIARY: Fitness vector fit of size m, initial frequency vector r^0 of size m, global best position vector $\vec{x}$ of size n, and variables acc, $rand$, $maxfit$, $maxindex$, $globalfit$, β, $f$max, $f$min.

```
1. For each bat $b_i$ ($\forall i$ = 1..., $m$), do
2.         For each feature $j$ ($\forall j$ = 1..., $n$), do
3.         $x_j \leftarrow$ Random {0, 1}
4.         $v_j \leftarrow 0$
5.         $A_i \leftarrow Random$ [1, 2]
6.         $r_i \leftarrow Random$ [0, 1]
7.         $fit_i \leftarrow -\infty$
8.         $globalfit \leftarrow -\infty$
9. For each iteration $t$ ($t$ = 1..., $T$), do
10.        For each bat $b_i$ ($\forall i$ = 1..., $m$), do
11. Create $Z'_1$ and $Z'_2$ from $Z_1$ and $Z_2$, respectively,
12. such that both contains only features in $b_i$ in
13. which $x_i^j \neq 0$, $\forall j = 1, \ldots, n$.
14. Train classifier over $Z'_1$, evaluate its over $Z'_2$
15. and stores the accuracy in $acc$.
16. $rand \leftarrow$ Random [0, 1]
17. If ($rand < A_i$ and $acc > fit_i$), then
```

```
18. $fit_i \leftarrow acc$
19. $A_i \leftarrow \alpha A_i$
20. $r_i \leftarrow r_i^0 [1 - exp(-\gamma t)]$
21. $[maxfit, maxindex] \leftarrow max(fit)$
22. If ($maxfit > globalfit$), then
23. $globalfit \leftarrow maxfit$
24. For each dimension $j$ ($\forall j = 1, . . ., m$), do
25. $\hat{x}^j \leftarrow x^j_{maxindex}$
26. For each bat $b_i$ ($\forall i = 1, . . ., m$), do
27. $\beta \leftarrow$ Random [0, 1]
28. $rand \leftarrow$ Random [0, 1]
29. If ($rand > r_i$)
30. For each feature $j$ ($\forall j = 1, . . ., n$), do
31. $x_i^j = x_i^j + \epsilon A$
32. $\sigma \leftarrow$ Random {0, 1}
33. If ($\sigma < 1/ (1+ \exp(-v_i^j))$), then
34. $x_i^j \leftarrow 1$
35. else $x_i^j \leftarrow 0$
36. $rand \leftarrow$ Random [0, 1]
37. If ($rand < A_i$ and $fit_i < globalfit$), then
38. For each feature $j$ ($\forall j = 1, . . ., n$), do
39. $f_i \leftarrow f_{\min} + (f_{\max} - f_{\min}) \beta$
40. $v_i^j \leftarrow v_i^j + (\hat{x}^j - x_i^j) fi$
41. $x_i^j \leftarrow x_i^j + v_i^j$
42. $\sigma \leftarrow$ Random {0, 1}
43 If ($\sigma < 1/ (1+ \exp(-v_i^j))$), then
44. $x_i^j \leftarrow 1$
45. else $x_i^j \leftarrow 0$
46. For each feature $j$ ($\forall j = 1, . . ., n$), do
47. $F^j \leftarrow \hat{x}^j$
48. Return $F$.
```

The first loop in line 1-7 initializes the population of bats. The bat's position is then initialized with randomly chosen binary values in Lines 2-3, which corresponds whether a feature will be selected or not to compose the new dataset. Lines 11-15 compose new [4]training and evaluating sets with selected features, and Line 17-22 evaluate each bat in order to update its fitness value. Moreover, the loudness A_i and pulse rate emission r_i are updated if a new solution have been updated. While loudness decreases once the bat finds its prey, the pulse rate emission increases. In Line 21, the function max outputs the index and the fitness value of the bat that maximizes the fitness function.

Line 22-25 update the global best position, i.e., $\hat{x}$ with the position of the bat that has achieved highest fitness function. The loop in Lines 26-45 is responsible for updating bat's position and velocity.

The source code in Lines 29-35 performs the local search. At line 39, we update the frequency. In their paper, the proposed f_{min} and f_{max} are 0 and 1 respectively. We have not modified these values in our project and have continued with these values only.

III. Model Selection Algorithm

The [3]feature selection algorithm given in section II is a random algorithm which needs experimental data to obtain a permanent solution. Owing to its random nature, it gives a different solution each time it is implemented. Thus, we have devised a model selection algorithm and used the experimental data from the above algorithm as its basis. Thus, it can be said that the algorithm given below is not an independent algorithm but is dependent on the algorithm for its implementation.

Algorithm.2. - MODEL SELECTION ALGORITHM

INPUT: The number of model, **m**, name of each model, **name**, factors present in the model, **fact[n]**, where n is the same as in algorithm above. It also consists of another array, **fact_main[x],** which represents the number of features of each feature dataset present in the model. **x** is the number of dataset present. **Feature []**, set of optimal features, obtained.

OUTPUT: The best model amongst the initialized ones.

```
For each model i (□ ∀i=1,2,3……, m), do
1.      For each model feature j (∀j=1, 2…., n), do
2.          $fact_i[j]$ ⟵ Assign attributes present in model
3.          $name_i$ ⟵ Initialize name of model
4.          $fact\_main_i$ ⟵ Initialise this array
5.   For each model i (□ ∀i=1,2,3……,m), do
6.          $Sum_i$: = 0
7.          For each model feature set j (∀j=1, 2….,    x), do
8.          $Sum_i$ ⟵ $Sum_i$ + ($fact\_main_i[j]$ /global[j])
9.   **BestModel** (mod, m)
```

In the algorithm.2, in **Lines 1-5,** each model is initialized with name, factors present in it and the number of attributes of each dataset present in it.

In **Lines 6-9**, the sum of attributes of each model is calculated and stored in the variable **Sum,** associated with each model.

The function **BestModel (mod, m)** is invoked in **Line 10,** which finds the model with maximum number of features present from the optimal models and considers it the best among the models present.

The algorithms above finally solve the problem at hand, and give us the best SDLC Model among the ones present. In our implementation, we have considered only 6 already existing models and have worked upon them. Although, considering the scope of this project, we can extend the application to be of more dynamic nature, i.e. user will be able to select the features upon which he wants to evaluate a model and also create a model himself.

IV. Evaluation And Assessment

Seven major features namely **Efficiency, Effectiveness, Satisfaction, Memorability, Security, Universality,** and **Productivity** were used to judge the models and rank them accordingly. These seven major factors were bifurcated into 23 sub-features whose presence or absence judged the SDLC models considered for evaluation.

Figure 1. shows a line graph representing the count of different sub-features that are used to distinguish different SDLC models. Each line represents a major feature. The values, however, depicts the number of sub-features that are used out of the total count of sub-features for a major feature. Each SDLC model is judged on the basis of these selected features.

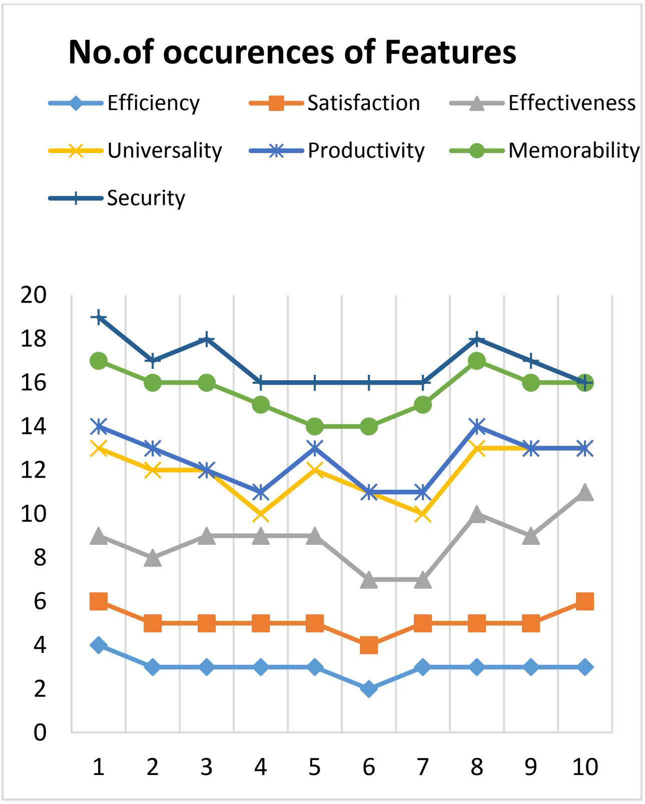

Figure.1: No of occurrences of features

A questionnaire was prepared was conducted as a survey to identify the presence or absence of the features in the different SDLC models which were being worked upon. The questionnaire can be viewed at https://megacloud.io/b57b225a35aca0b6 **Table 1.** Demonstrates the final ranking of the models been worked upon along with their calculated accuracies. The accuracies were calculated on the basis of the result obtained after running [2]BBA on a set of features, and then running **Algorithm 2** on that set of features.

Table 1: Ranking of SDLC models

SDLC MODEL	ACCURACY	RANKING
Spiral Model	0.915	1
Iterative Model	0.7095	2
RAD Model	0.6132	3
Prototype Model	0.55545	4
Waterfall Model	0.53225	5
Build & Fix Model	0.14439	6

V. CONCLUSIONS

In this paper, various Software Development Life Cycle Models (already existing ones) were worked upon. The result shows the ranking of sdlc models and the most accurate model with the help of Binary Bat Algorithm. The factors upon which the aptness of all the SDLC models under consideration can be evaluated were identified in this project with the help of surveys and questionnaires.

REFERENCES

[1] Xin-She Yang: ***A New Metaheuristic Bat-Inspired Algorithm*:** Department of Engineering, University of Cambridge, Trumping ton Street, Cambridge CB2 1PZ, UK. **(2010)**

[2] R. Y. M. Nakamura, L. A. M. Pereira, K. A. Costa, D. Rodrigues, J. P. Papa: ***BBA: A Binary Bat Algorithm for Feature Selection* :** Department of Computing Sao Paulo State University ˜Bauru, Brazil. **(2012)**

[3] Douglas Rodrigues, Kelton A. P. Costa, Lu´ıs A. M. Pereira: ***A Wrapper Approach for Feature Selection based on Bat Algorithm and Optimum-Path Forest*:** Department of Computing, Univ. Universidad Estadual Paulist, and Bauru, Brazil. **(2014)**

[4] ***Planning Fitness Training Sessions Using the Bat Algorithm*:** Iztok Fister Jr.1, Samo Rauter2, Karin Ljubi˘c Fister3, Dušan Fister1, and Iztok Fister. **(2013)**

[5] **Naïve-Bayes Guided Bat Algorithm for Feature Selection** by Ahmed Maji Taha, Adia Mustapha and Soong-Der Chen **(2013)**