Complex & Intelligent Systems (2023) 9:3487–3497
https://doi.org/10.1007/s40747-021-00384-z

ORIGINAL ARTICLE

Feature selection algorithm for usability engineering: a nature inspired approach

Rajat Jain[1] · Tania Joseph[1] · Anvita Saxena[1] · Deepak Gupta[1] · Ashish Khanna[1] · Kalpna Sagar[2] · Anil K. Ahlawat[2]

Received: 9 September 2020 / Accepted: 15 April 2021 / Published online: 12 May 2021

Abstract

Software usability is usually used in reference to the hierarchical software usability model by researchers and is an important aspect of user experience and software quality. Thus, evaluation of software usability is an essential parameter for managing and regulating a software. However, it has been difficult to establish a precise evaluation method for this problem. A large number of usability factors have been suggested by many researchers, each covering a set of different factors to increase the degree of user friendliness of a software. Therefore, the selection of the correct determining features is of paramount importance. This paper proposes an innovative metaheuristic algorithm for the selection of most important features in a hierarchical software model. A hierarchy-based usability model is an exhaustive interpretation of the factors, attributes, and its characteristics in a software at different levels. This paper proposes a modified version of grey wolf optimisation algorithm (GWO) termed as modified grey wolf optimization (MGWO) algorithm. The mechanism of this algorithm is based on the hunting mechanism of wolves in nature. The algorithm chooses a number of features which are then applied to software development life cycle models for finding out the best among them. The outcome of this application is also compared with the conventional grey wolf optimization algorithm (GWO), modified binary bat algorithm (MBBAT), modified whale optimization algorithm (MWOA), and modified moth flame optimization (MMFO). The results show that MGWO surpasses all the other relevant optimizers in terms of accuracy and produces a lesser number of attributes equal to 8 as compared to 9 in MMFO and 12 in MBBAT and 19 in MWOA.

Keywords Software usability · Software development · Hierarchical usability model · Grey wolf optimization

Introduction

In recent years, software engineering practices have changed to develop software products that are good in quality. International Standard Organization (ISO) [1] has defined various quality factors like effectiveness, usability, efficiency, reliability, etc. which are crucial for the manufacturing of stellar software products. As stated by Boehm et al. (1976), evaluation of quality is just as essential as the assessment of functionality for any software product [12].

Amidst these factors which determine quality, usability plays a particularly important role that has to be taken into account during the various processes of software development [43]. Software engineering experts interpret usability in their own terms [2]. In simple words, usability of software is described as the ease with which a man-made object can be used, remembered and learnt. The object could be an application, tool, website, machine, process or any other thing

✉ Deepak Gupta
deepakgupta@mait.ac.in

Rajat Jain
rajat06jain47@gmail.com

Tania Joseph
taniajoseph37@gmail.com

Anvita Saxena
anvanvita2207@gmail.com

Ashish Khanna
ashishkhanna@mait.ac.in

Kalpna Sagar
kalpna.sagar@kiet.edu

Anil K. Ahlawat
anil.ahlawat@kiet.edu

[1] Department of Computer Science and Engineering, Maharaja Agrasen Institute of Technology, Delhi, India

[2] KIET Group of Institutions, Delhi-NCR, Ghaziabad, India

with which a man can interact. The definition of usability and characteristics of quality of software have been detailed by numerous standards and models over the years: Usability is defined with reference to the effort needed for use by the ISO/IEC 9126 [28]. The ISO/IEC 9126 further rewrites the interpretation of software usability as the potential of the software to be discerned by different users under different circumstances and/or situations. The ISO 9241-11 details usability by taking into consideration the efficiency and effectiveness, as well as the effectiveness of the software in a particular medium of usage [1]. The IEEE Std.610.12-1990 has defined usability with respect to input and output efficiency as well as learnability of the system [29]. ISO/IEC 25010 (2011) describes a quality in use model which consists of five components that chronicle the aftermath of interaction when the use of a product is tried in a subjective setting. It also defines a product quality model as having eight characteristics that associate with the static software traits and dynamic computer system traits [30]. ISO 9241-11:2018 spells out usability in terms of user performance and satisfaction, with special priority given to the fact that usability is reliant on the different conditions that a product is used in [31].

As seen above, several attempts have been made to define as well as evaluate software usability using a variety of methods, criteria, strategies, and different features [13–18] which tend to generate conflicting usability models, and this results in confusion and discrepancies in its usage and practice. If we look at the amount of data available in the last few years, we find that it has increased with respect to the number of features and instances which results in large amounts of data. Such data decrease efficiency of a model by increasing computational cost and slowing the rate of training the data. To increase the efficiency of models, a need for feature selection arises.

Feature selection is the method of choosing only important features from a given set of features [19–21]. It means selecting a subset from a given set of features to improve performance.They must be selected keeping in mind that a balance should be maintained amongst the number of selected features as well as the performance of the system. Over the years, feature selection has been used to reduce features in diverse fields, for instance,in health care data analysis [32], flash flood hazard assessment [33], and for medical image diagnosis [34], among other applications. It has also been used in the area of software usability, for reduction of usability attributes [8,9,11], to identify problematic usability attributes [35] and to detect usability deficiencies by monitoring the amount of time taken in different tasks by users [36].

Evolutionary algorithms refers to a set of algorithms that are primarily inspired by biological evolution like mutation, recombination, selection and reproduction [22,23]. Almost all optimization problems can be handled by these algorithms, as they function decently in approximating solutions. Most of these algorithms use fitness function calculation for optimizing problems [24–26]. Various evolutionary algorithms which have been used in the past for feature selection are grey wolf optimization [11], bat algorithm [3], chaotic crow search algorithm [4], whale optimization algorithm [5], genetic algorithms [6], cuckoo search [7], and recently studied MMFO [37].

With the goal of choosing only the important features in mind, we have decided to use an evolutionary algorithm for feature selection and we have chosen grey wolf optimization algorithm (GWO) for feature reduction. The GWO algorithm [11] imitates the mechanism used by grey wolves for hunting and also takes into account the way in which their chain of command (leadership) works. The social hierarchy and hunting technique of grey wolves are modeled mathematically to scheme out the GWO algorithm and perform optimization. Feature selection problems having binary datasets can reach an efficient solution through this algorithm. This paper optimizes a nature inspired algorithm used for optimization, called grey wolf optimization algorithm (GWO), which will be used for selecting the optimum features from a given collection of features that is chosen from a private dataset containing usability attribute information. Moreover, a detailed study of previous results of various papers with outcomes obtained is done. Thus, the main highlights for this paper are as follows: A modified evolutionary algorithm is deployed on a dataset to select the optimum features, named as modified grey wolf optimization algorithm (MGWO). An analogous study with results of different optimization algorithms is made. A hierarchical model, containing usability attributes and factors, has been used in this paper which represents all features and characteristics of software development. The modified algorithm is implemented on the given private dataset, which has been obtained through a survey and a resulting subgroup of the features is obtained. The reduced features are employed on 6 SDLC models, and it is then determined which model out of the 6 is the best, according to MGWO. The results of the implementation are compared with the outcomes given by implementing GWO,MBBAT, MWOA, and a study that has been done recently, named MMFO.

In recent years, many modifications have been done to the GWO Algorithm to help select optimal features. Kohli and Arora [38] acquaints the Chaos theory with the GWO algorithm, thus making a hybrid which is then used in optimization problems that are constrained. The objective behind including the Chaos theory is to increase its global convergence speed. Another modification of GWO [39] is used to classify images of galaxies with better precision, achieved by introducing opposition based learning (OBL), chaotic map, disruption operator (DO) and differential evolution (DE). A hybrid cuckoo search -grey wolf optimization (HCS-GWO)

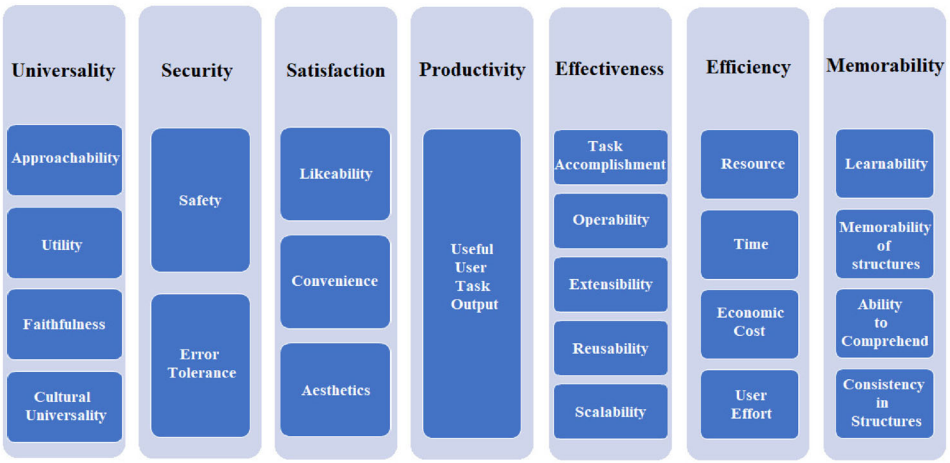

Fig. 1 Classification of various software usability factors

[40] has been used to fuse multi modal medical images together. The parameters of cuckoo search are used as the control parameters of the GWO in this particular hybrid approach. Other improvements include a hybrid GWO algorithm [41], which integrates Particle Swarm Optimization along with GWO to achieve better results, and a Variable Weight GWO (VW-GWO) [42] which considers the possibility of a wolf being followed. The subsequent section illustrates the major aspects of the hierarchical-based usability model, and is followed by Sect. 3 which acquaints us with the GWO for feature selection, succeeded by the implementation of the Modified GWO for usability feature reduction. Section 5 reviews our results, and compares them with other previously implemented algorithms to arrive at a conclusion. This is followed by the list of references.

Major contributions obtained through this paper are as follows:

1. Features required to predict software usability is reduced.
2. Grey Wolf Optimization algorithm is modified to produce minimal subset of attributes.
3. A comparison is done between results obtained through GWO and MGWO.
4. Comparison is done between various software usability models according to features obtained through MGWO.

The hierarchical-based usability model

Many usability models have been presented over the last twenty years and each model operates on its own set of features, and hence creates a considerable amount of problems for software engineers in the application of these models. Same features have different names in different models. This paper uses seven basic usability factors which are further classified into twenty three features and forty two characteristics. The purpose of this research paper is to use the algorithm

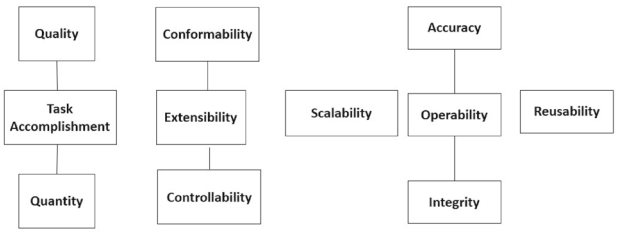

Fig. 2 Effectiveness—usability features and characteristics

to define a minimal subset of features that are used to define software usability. The 7 basic usability factors along with their features are described as below:

Effectiveness: Effectiveness can be defined as a degree of performance, accomplished by an individual when performing a particular task with full integrity. This factor can be broken down into five features. The features are extensibility, accomplishment of tasks, operability, reusability as well as scalability as shown in Fig. 2.

Efficiency: Efficiency can be defined as the ratio of expected output calculated by an end user to the number of invested resources. It comprises four features within itself. The features are economic costs, resource, time and user effort as shown in Fig. 3.

Memorability: Memorability can be defined as the extent of an end user's ability to memorize/remember different components of a software with utmost clarity. It also consists of four features. The features are Comprehensibility, consistency in structures, learnability, memorability of structures as shown in Fig. 4.

Productivity: Productivity can be defined as output obtained by end users from software. It does not contain many features as it is self explanatory. It consists of only 1 feature that is useful user task output as shown in Fig. 1.

Satisfaction: Satisfaction can be defined as the degree of satisfaction of an end user in their response or feeling after using a product/software. Satisfaction can be further divided

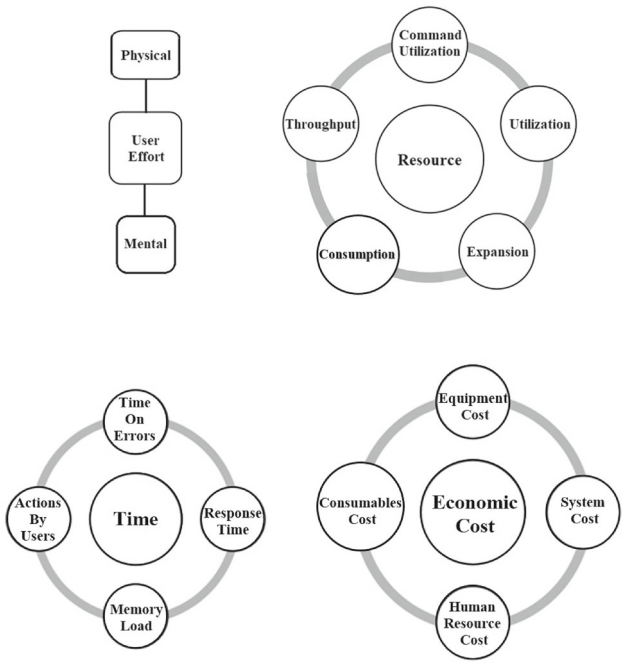

Fig. 3 Efficiency—usability features and characteristics

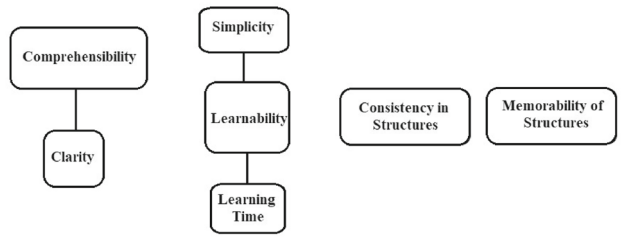

Fig. 4 Memorability—usability features and characteristics

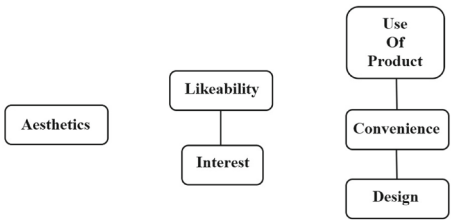

Fig. 5 Satisfaction—usability features and characteristics

into 3 smaller features. The features are likeability, convenience and aesthetics as shown in Fig. 5 .

Security: Security can be defined as the analysis of the extent of risks, how prone the hardware and software are to failures, and the damages that are likely to be caused to software. This factor is divided into 2 more features. The features are Error Tolerance and Safety as shown in Fig. 6

Universality: This feature is used to measure the degree to which a product can connect different users of different cultures, thus giving us an idea of the actual usage of the product in various perceptions. Universality can be split into four

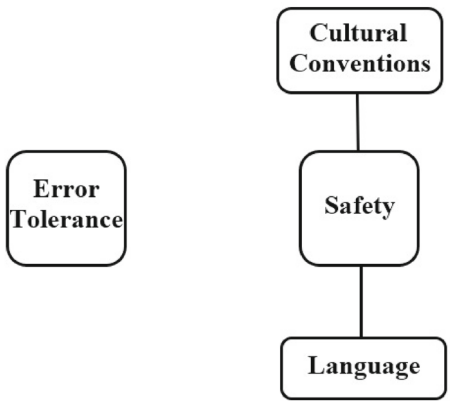

Fig. 6 Security—usability features and characteristics

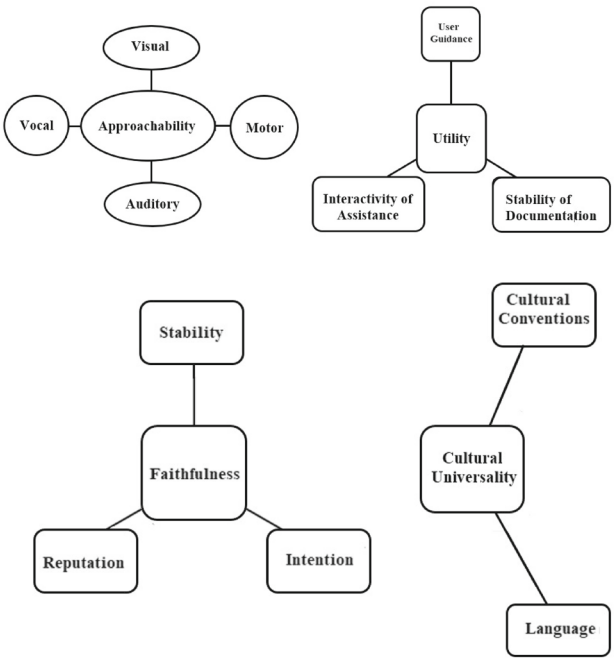

Fig. 7 Universality—usability features and characteristics

more features; utility, approachability, cultural universality and faithfulness, as shown in Fig. 7.

Grey wolf optimization algorithm

Grey wolf optimization algorithm (GWO) mimics the hunting mechanism used for prey and pays attention to the way in which the wolves are ranked in a pack (the hierarchy of leadership) [27]. With the chain of command of grey wolves in mind, the four types of grey wolves, namely, alpha, beta, delta and omega are assigned to perform their respective duties according to their place/rank in the pack. Four major phases of the process, i.e. hunting, looking for prey, cornering and/or

trapping the prey once encountered, and then finally ambushing the prey are performed to implement this optimization algorithm. Alpha wolves, intriguingly, may not be the toughest pack members, but are considered to be the best choice for managing all decisions related to the pack, including arrangements about sleeping, waking times, hunting and so on. Beta wolves are considered to be subordinates of alpha wolves that assist them in making decisions and other endeavors of the pack. Delta wolves are subordinates to both alpha and beta wolves and usually take on the responsibility of watching the territorial boundaries and warning the rest of the pack in case of any impending dangers. Omega wolves are considered to be lowest in their hierarchy. They do not play much role in hunting, and have to submit to all the other wolves.

Mathematical model

Encircling the prey

$$\vec{S} = \mid \vec{R}.\vec{Y_p}(t) - \vec{Y}(t) \mid \tag{1}$$

$$Y(\vec{t+1}) = \vec{Y_p}(t) - \vec{P}.\vec{S}. \tag{2}$$

Here $\vec{P}$ and $\vec{R}$ are termed as coefficient vectors while t refers to current iteration.$\vec{Y}$ refers to the current position of the prey and $\vec{Yp}(t)$ refers to the position of prey. Coefficient vectors are calculated as given below:

$$\vec{P} = 2\vec{a}.\vec{r_1} - \vec{a} \tag{3}$$

$$\vec{R} = 2.\vec{r_2}. \tag{4}$$

Here $\vec{a}$ is linearly decreased over the course of iterations from 2 to 0 and r1 and r2 are random vectors in range (0, 1).

Hunting

$$\vec{S_\alpha} = \mid \vec{R_1}.\vec{Y_\alpha} - \vec{Y} \mid \tag{5}$$

$$\vec{S_\beta} = \mid \vec{R_2}.\vec{Y_\beta} - \vec{Y} \mid \tag{6}$$

$$\vec{S_\delta} = \mid \vec{R_3}.\vec{Y_\delta} - \vec{Y} \mid \tag{7}$$

$$\vec{S_1} = \vec{Y_\alpha} - \vec{P_1}.(\vec{S_\alpha}) \tag{8}$$

$$\vec{Y_2} = \vec{Y_\beta} - \vec{P_2}.(\vec{S_\beta}) \tag{9}$$

$$\vec{Y_3} = \vec{Y_\delta} - \vec{P_3}.(\vec{S_\delta}) \tag{10}$$

$$\vec{Y}(t+1) = \frac{\vec{Y1} + \vec{Y2} + \vec{Y3}}{3}. \tag{11}$$

With the above equations $\vec{S}$ is calculated for alpha beta and delta wolves and then positions of alpha , beta and delta wolves is updated.

Pseudo-code for GWO

Algorithm 1 GWO Algorithm

procedure GWO(W,Max-iterations)
  Initialize the grey wolf population with random values.
  Initialize â and R
  Calculate fitness value of each and every search agent
  $\vec{W_a} \leftarrow f1$ ▷ select wolf having best fitness value
  $\vec{W_b} \leftarrow f2$ ▷ select wolf having second best fitness value
  $\vec{W_c} \leftarrow f3$ ▷ select wolf having third best fitness value
  while $iter \neq Max - Iteration$ **do**
    a=2-iter*((2)/Max-iterations)
    for `each wolf wi (i = 1..., m),` **do**
      $\vec{r1} \leftarrow Random[0, 1]$
      $\vec{r2} \leftarrow Random[0, 1]$
      $\vec{P1} \leftarrow 2 * \vec{a} * \vec{r1} - \vec{a}$
      $\vec{R1} \leftarrow 2 * \vec{r2}$
      $\vec{S_a} \leftarrow abs(\vec{R1} * \vec{W_a} - \vec{wi})$
      $\vec{Y1} \leftarrow= W_a - \vec{P1} * \vec{S_a}$
      `Use above equations again to calculate Y2,Y3`
      `Update Positions of wolf with average of Y1,Y2,Y3 vectors`
    end for
    Recalculate fitness of each wolf
    $UpdateW_a, W_b, W_c$
  end while
end procedure

Modified grey wolf algorithm for usability feature selection

In the proposed MGWO approach, the GWO algorithm procedure is modified with the aim of "usability feature selection". An optimal feature set is acquired, when the features are assigned as input for the modified algorithm. This optimal feature set is then used to model software development life cycle models. Since the combined effort of alpha, beta and delta wolves leads to hunting, we have added features representing alpha , beta and delta wolves in the selected features vector as they form an important part for calculation of usability. In GWO, since the updated solution depends on positions of alpha, beta and delta wolves, in MGWO, alpha wolves, beta wolves and delta wolves are initialized with positions of attributes having best fitness values. Fitness values at each iteration for an attribute is calculated by the sum of ones for that attribute in the dataset. Also in MGWO , positions of wolves are initialized by the values in the dataset.

Algorithm 2 MGWO Algorithm

procedure MGWO(W,Max-iterations,dataset)
  $Alpha\_pos \leftarrow zerovectororder(1,6)$
  $Beta\_pos \leftarrow zerovectororder(1,6)$
  $Delta\_pos \leftarrow zerovectororder(1,6)$
  $selected\_features \leftarrow emptylist[]$
  $Positions \leftarrow dataset\_values$
  for each iteration $(t \leftarrow 1, ...Max_Iterations)$, **do**
    for each feature xi $(i \leftarrow 1, ...m)$, **do**
      $fitness\_eachi \leftarrow fitness\_function(xi)$
      if $fitness\_eachi > Alpha\_score$ **then**
        $Delta\_score \leftarrow Beta\_score$
        $Beta\_score \leftarrow Alpha\_score$
        $Alpha\_score \leftarrow fitness\_eachi$
        $Delta\_pos \leftarrow Beta\_pos$
        $Beta\_pos \leftarrow Alpha\_pos$
        $Alpha\_pos \leftarrow xi$
        $Alpha\_idx \leftarrow i$
      else if $fitness\_eachi < Alpha\_Score$ && $fitness\_eachi > Beta\_score$ **then**
        $Delta\_score \leftarrow Beta\_score$
        $Beta\_score \leftarrow fitness\_eachi$
        $Delta\_pos \leftarrow Beta\_pos$
        $Beta\_pos \leftarrow xi$
        $Beta\_idx \leftarrow i$
      else if $fitness\_eachi < Alpha\_Score$ && $fitness\_eachi < Beta\_score$ && $fitness\_eachi > Delta\_score$ **then**
        $Delta\_score \leftarrow fitness\_eachi$
        $Delta\_pos \leftarrow xi$
        $Delta\_idx \leftarrow i$
      end if
    end for
    Append Alpha_pos, Beta_pos, Delta_pos to selected_features
    for each feature xi $(i \leftarrow 1, ...m)$, **do**
      for each model j $(j \leftarrow 1, ..., n)$, **do**
        $\vec{r1} \leftarrow Random[0,1]$
        $\vec{r2} \leftarrow Random[0,1]$
        $\vec{P1} \leftarrow 2 * \vec{a} * \vec{r1} - \vec{a}$
        $\vec{R1} \leftarrow 2 * \vec{r2}$
        $\vec{S_a} \leftarrow abs(\vec{R1} * \vec{W_a} - \vec{wi})$
        $\vec{Y1} \leftarrow = W_a - \vec{P1} * \vec{S_a}$
        Use above equations again to calculate Y2,Y3
        $var \leftarrow \frac{Y1+Y2+Y3}{3}$
        if var >0.5 **then**
          Positions[i][j]=1
        else if var<0.5 **then**
          Positions[i][j]=0
        end if
      end for
    end for
    return selected_features

Steps involved in the MGWO algorithm are explained below:

1. For the first six lines the variables Alpha_ pos, Beta_ pos,Delta_ pos,selected_ features Alpha_ score,Beta_ score,Delta_ score have been initialized. The position of the wolves are initialized with dataset values.
2. The loop in lines 7–47 run until Max_ iter times
3. In lines 8 and 29 fitness value of each attribute/wolf is calculated using fitness function and alpha_ wolf is selected such that it has maximum fitness value,beta_ wolf has second maximum fitness value and delta_ wolf has third maximum fitness value.
4. In line 30 index positions of alpha_ wolf,beta_ wolf and delta_ wolf are appended into the list of selected features.
5. The loop in line 31–46 updates positions of wolves. If the updated position is greater than 0.5 it is assigned 1 value and if it is less than 0.5 it is assigned 0 value.
6. Line 48 returns a list of selected features.

Implementation of modified GWO for feature selection

In this paper, a course of action has been followed for the implementation of the proposed model. Through this, we have aimed to establish the usability of the Software Development Life Cycle (SDLC) Models according to their usability attributes. The ranking has been accomplished by implementing the GWO algorithm. Six SDLC models have been analyzed using this algorithm on a dataset. The dataset contains these six SDLC models and their functionalities, and 7 factors and 23 attributes which describe their behavior well.

According to the hierarchical based usability model, 6 SDLC models are evaluated on the basis of 23 attributes. These models along with their attributes have been outlined in Fig. 8. The numbers 0 and 1 have been used to represent whether or not the SDLC models require a particular attribute. In this section, Python has been used to code the MGWO algorithm. Python is a dynamic language that is also portable, modular and interactive. The present section reviews experimental based setup, input parameters and dataset used.

Experimental based setup

To assess the suggested algorithm, a computing device with Processor Intel(R) Core(TM) i7-7500U CPU @ 2.70 GHz, 2904 Mhz, 2 Core(s), 4 Logical Processor(s) and 8 GB Ram under Ubuntu 16.04 is being used. The implementation is coded in python 3.6.3. The proposed algorithm is used to determine reduced optimal features for software usability.It is also used to calculate accuracy for each software development life cycle model. Hence implementation is divided into two categories. First is obtaining reduced optimal features and second is finding accuracy for each software development life cycle dataset.

The dataset

In the dataset used, the columns are filled with 23 usability features and rows are occupied by 6 software development life cycle models. 0 and 1, also called the binary numbers, are used to specify whether these features are present or not in the six software development life cycle models. This research paper has used a dataset which has been taken from [2,9].

Results and discussion

Through this section, the results of the application of the dataset to MGWO are analysed thoroughly. After conducting cross-validation for twenty iterations, the selected attributes v/s the number of iterations, as well as accuracy v/s number of iterations has been plotted and the same has been shown in Figs. 9 and 10. Figure 9 shows that when proposed algorithm is applied over the course of twenty iterations, 8 features are obtained as a result which can be used to predict usability. Figures 9 and 10 show that MGWO brings about a set of attributes that is optimal, contains 8 features, and is 75% accurate, over the course of final iteration. Hence, an optimized algorithm has been found that takes as input, a binary dataset, and produces a minimal subset of attributes of an output with quite good accuracy.

The plot of accuracy for different life cycle models has been depicted in Fig. 11. Now according to accuracy we can find which model is best for software development. In the graph below, we can see spiral and evolutionary models give quite a good accuracy for the features selected through MGWO.

MGWO algorithm selects eight attributes . The selected eight attributes are Operability, Cultural Universality, Resource, Task accomplishment, Learnability, User Effort and Safety. In this section, we compare the results of the Modified Grey Wolf Algorithm with other optimization algorithms that have been previously used for usability feature selection.. The results of MGWO have also been compared with standard GWO in Figs. 12 and 13. The comparison between MGWO and GWO for 20 iterations has been plotted in Fig. 12. It shows that the selected attributes in GWO are 13 which is more than the selected number of attributes in MGWO. Therefore, we can say that the proposed MGWO surpasses the results of standard GWO and gives us a lesser number of attributes. Moreover, accuracy obtained through MGWO is greater than accuracy obtained through GWO as shown in Fig. 13.

	Resource	Time	User Effort	Economic Costs	Likeability	Convenience	Aesthetics	Task Accomplish	Operability	Extensibility	Reusability	Scalability	Approachability	Utility	Faithfulness	Cultural Univers-ality	Useful Task o/p	Learnability	Memorability	Comprehensible	Consistency	Safety	Error Tolerance
Spiral	1	0	1	0	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	1
Iterative	1	0	0	0	1	0	1	1	1	0	0	0	0	1	0	1	0	1	1	1	1	1	1
RAD	1	0	1	1	0	1	0	1	1	1	1	0	0	1	0	0	0	1	1	1	0	1	0
Evolutionary	1	0	1	1	0	1	1	1	1	0	0	1	0	1	1	1	1	1	0	0	0	1	0
Waterfall	0	1	0	1	1	0	0	0	1	1	0	1	1	1	0	0	0	1	0	1	0	1	0
Build & Fix	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	1	0	0	0

Fig. 8 Illustration of the used dataset

We have compared results obtained through MGWO which results obtained through other algorithms like MMFO, MBBAT and MWOA as shown in Fig. 14. It has been seen that MGWO selects eight features while all other algorithms selects more than eight features. It shows that MGWO produces the minimum number of attributes as compared to other algorithms.

The plot of the number of selected features for each SDLC model has been shown in Fig. 15. It shows that Spiral and evolutionary models contain all the optimized features according to a private dataset that has been shown in Fig. 8.

The accuracy v/s selected attributes for MGWO has been plotted and shown in Fig. 16. Accuracy obtained is maximum when 8 features are selected.

Conclusions and future scope

The term "usability" has been defined, using a hierarchical-based usability model. In this model, usability of a software has been characterized with the help of 7 factors having 23 attributes in all. In this attempt, we have implemented

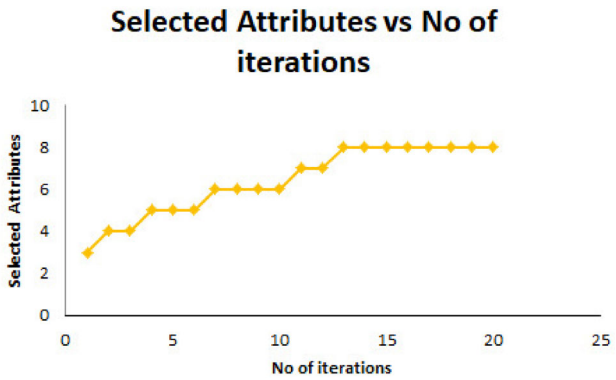

Fig. 9 Plot of selected attributes vs no. of iterations

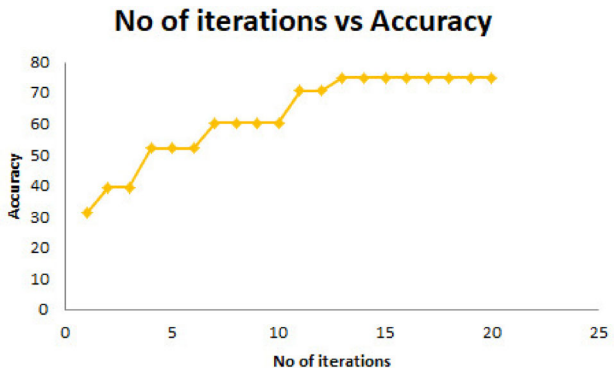

Fig. 10 Plot of selected attributes vs accuracy

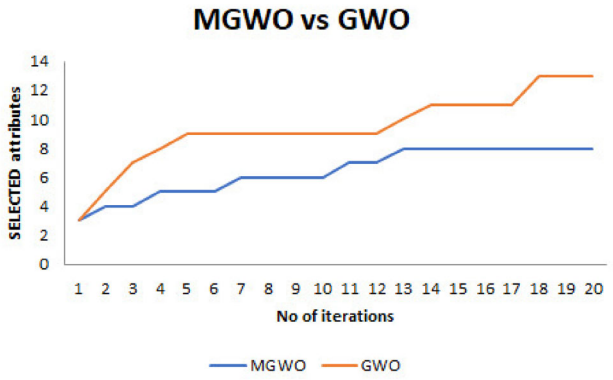

Fig. 12 Plot of selected attributes vs no. of iterations for MGWO and GWO

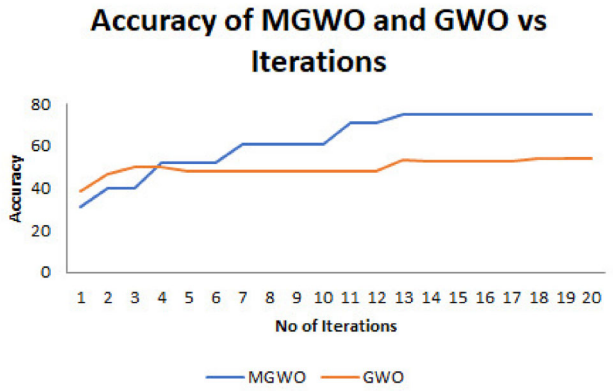

Fig. 13 Plot of accuracy of GWO and MGWO vs no. of Iterations

the modified grey wolf algorithm (MGWO) to the usability model for usability feature selection. MGWO aims to lessen the number of attributes, and provides us with a consistent feature subset that is best suited for the problem, and does so without lowering the system performance. The MGWO surpasses other optimization algorithms by predicting less number of attributes with quite good accuracy. Modified Grey Wolf Optimization algorithm is suitable for use by various researchers as a means to calculate the usability of numerous applications and software, and can help in deciding which usability features would most affect the same.

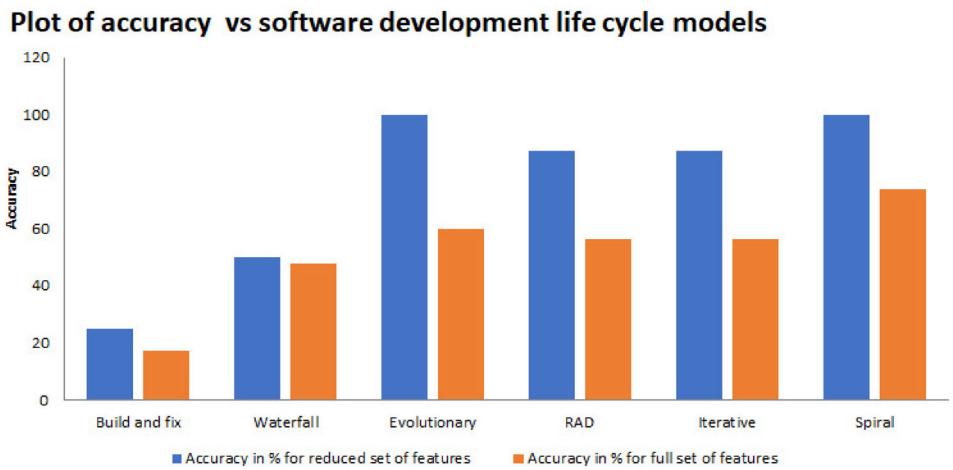

Fig. 11 Accuracy comparison for six SDLC models

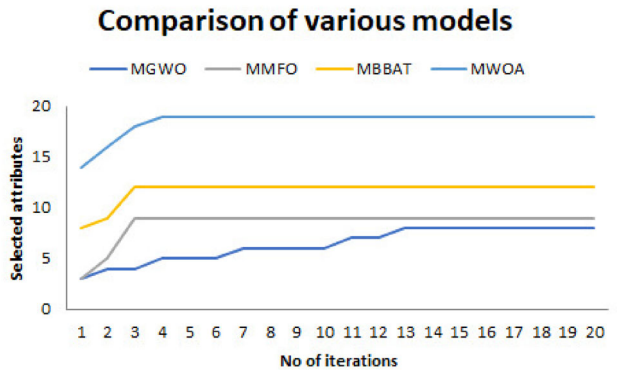

Fig. 14 Plot of selected attributes vs no. of iterations for MGWO, MMFO, MBBAT, and MWOA

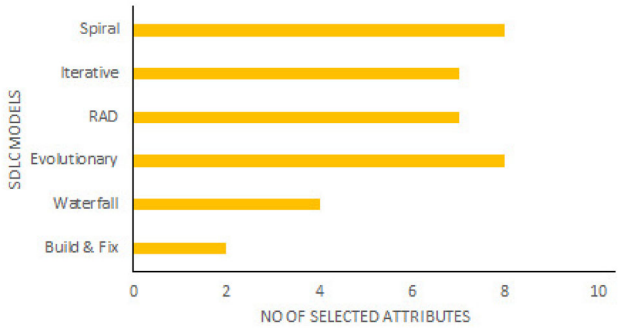

Fig. 15 Plot of number of selected features used for various SDLC models

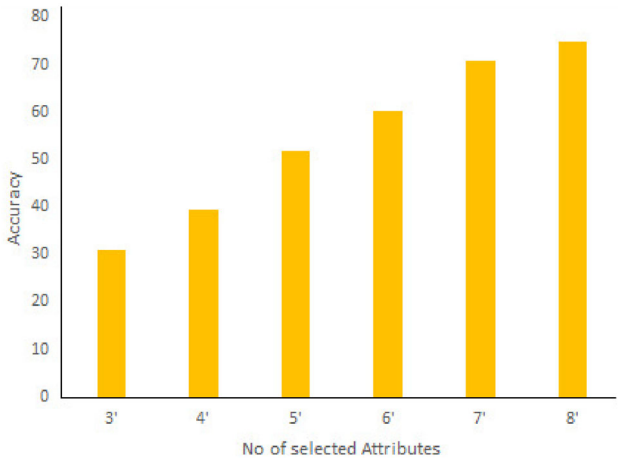

Fig. 16 Plot of accuracy vs selected attributes for MGWO

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Iso W (1998) 9241–11. Ergonomic requirements for office work with visual display terminals (VDTs), The international organization for standardization, p 45
2. Gupta D, Ahlawat AK, Sagar K (2017) Usability prediction and ranking of SDLC models using fuzzy hierarchical usability model. Open Eng 7(1):161–168
3. He Y, Zhou J, Li C, Yang J, Li Q (2008) A precise chaotic particle swarm optimization algorithm based on improved tent map. In: 2008 Fourth International Conference on Natural Computation (Vol. 7, pp. 569-573). IEEE
4. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
5. Yang XS (2010) A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010) (pp. 65-74). Springer, Berlin, Heidelberg
6. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: 2009 World congress on nature and biologically inspired computing (NaBIC) (pp. 210-214). IEEE
7. Gupta D, Ahlawat A, Sagar K (2014) A critical analysis of a hierarchy based Usability Model. In: 2014 international conference on contemporary computing and informatics (IC3I) (pp. 255–260). IEEE
8. Jain R, Gupta D, Khanna A (2019) Usability feature optimization using MWOA. In: International conference on innovative computing and communications (pp. 453–462). Springer, Singapore
9. Gupta D, Ahlawat AK (2017) Usability feature selection via MBBAT: a novel approach. J Comput Sci 23:195–203
10. Gupta D, Rodrigues JJ, Sundaram S, Khanna A, Korotaev V, de Albuquerque VHC (2018) Usability feature extraction using modified crow search algorithm: a novel approach. Neural Comput Appl: 1–11
11. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
12. Boehm BW, Brown JR, Lipow M (1976) Quantitative evaluation of software quality. In: Proceedings of the 2nd international conference on Software engineering (pp. 592–605)
13. Dillon A (2001) The evaluation of software usability. Taylor and Francis, London
14. Hartson HR, Andre TS, Williges RC (2001) Criteria for evaluating usability evaluation methods. Int J Hum Comput Interact 13(4):373–410
15. Seffah A, Donyaee M, Kline RB, Padda HK (2006) Usability measurement and metrics: a consolidated model. Softw Qual J 14:159–178
16. Shackel B (2009) Usability-context, framework, definition, design and evaluation. Interact Comput 21(5–6):339–346
17. Sagar K, Saha A (2017) A systematic review of software usability studies. Int J Inform Technol: 1–24
18. Madan A, Dubey SK (2012) Usability evaluation methods: a literature review. Int J Engi Sci Technol 4(2):590–599

19. Dash M, Liu H (1997) Feature selection for classification. Intell Data Anal 1(3):131–156
20. Li J, Cheng K, Wang S, Morstatter F, Trevino RP, Tang J, Liu H (2016) Feature selection: a data perspective. arXiv preprint arXiv:1601.07996
21. Cai J, Luo J, Wang S, Yang S (2018) Feature selection in machine learning: a new perspective. Neurocomputing 300:70–79
22. Back T (1996) Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford University Press
23. Dasgupta D, Michalewicz Z (eds) (2013) Evolutionary algorithms in engineering applications. Springer
24. Eiben AE, van Hemert JI, Marchiori E, Steenbeek AG (1998) Solving binary constraint satisfaction problems using evolutionary algorithms with an adaptive fitness function. In: International Conference on Parallel Problem Solving from Nature (pp. 201–210). Springer, Berlin, Heidelberg
25. Richter H (2002) An evolutionary algorithm for controlling chaos: the use of multi–objective fitness functions. In: International Conference on Parallel Problem Solving from Nature (pp. 308–317). Springer, Berlin, Heidelberg
26. Bowie JU, Eisenberg D (1994) An evolutionary approach to folding small alpha-helical proteins that uses sequence information and an empirical guiding fitness function. Proc Natil Acad Sci 91(10):4436–4440
27. Projects, Contributors to Wikimedia. 2020 "Algorithm Models/Grey Wolf Optimizer." Wikiversity,15 Apr 2020 en.m.wikiversity.org/wiki/Algorithm_models/Grey_Wolf_Optimizer
28. ISO I (1991) Information technology-software product evaluation-quality characteristics and guide lines for their use. Iso/iec is, 9126
29. Radatz J, Geraci A, Katki F (1990) IEEE standard glossary of software engineering terminology. IEEE Std 610121990(121990):3
30. ISO/IEC, 2011. ISO/IEC 25010: 2011 Systems and software engineering–systems and software quality requirements and evaluation (SQuaRE)–system and software quality models
31. Iso, I.S.O., 2018. 9241-11 (2018) Ergonomics of human-system interaction–part 11: usability: definitions and concepts. Int Org Standard. https://www.iso.org/obp/ui/#iso:std:iso, 9241(11)
32. Suresh A, Kumar R, Varatharajan R (2020) Health care data analysis using evolutionary algorithm. J Supercomput 76(6):4262–4271
33. Hosseini FS, Choubin B, Mosavi A, Nabipour N, Shamshirband S, Darabi H, Haghighi AT (2020) Flash-flood hazard assessment using ensembles and Bayesian-based machine learning models: application of the simulated annealing feature selection method. Sci Total Environ 711:135161
34. Fahad LG, Tahir SF, Shahzad W, Hassan M, Alquhayz H, Hassan R (2020) Ant colony optimization-based streaming feature selection: an application to the medical image diagnosis. Sci Program
35. Sagar K, Saha A (2017) Qualitative usability feature selection with ranking: a novel approach for ranking the identified usability problematic attributes for academic websites using data-mining techniques. Human Centric Comput Inform Sci 7(1):29
36. Tamir DE, Komogortsev OV, Mueller CJ, Venkata DK, LaKomski GR, Jamnagarwala AM (2011) Detection of software usability deficiencies. In: International Conference of Design, User Experience, and Usability (pp. 527–536). Springer, Berlin, Heidelberg
37. Gupta D, Ahlawat AK, Sharma A Rodrigues JJ (2020) Feature selection and evaluation for software usability model using modified moth-flame optimization. Computing
38. Kohli M, Arora S (2018) Chaotic grey wolf optimization algorithm for constrained optimization problems. J Computat Design Eng 5(4):458–472
39. Ibrahim RA, Abd Elaziz M, Lu S (2018) Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization. Expert Syst Appl 108:1–27
40. Daniel E, Anitha J, Gnanaraj J (2017) Optimum laplacian wavelet mask based medical image using hybrid cuckoo search-grey wolf optimization algorithm. Knowl Based Syst 131:58–69
41. Teng ZJ, Lv JL, Guo LW (2019) An improved hybrid grey wolf optimization algorithm. Soft Comput 23(15):6617–6631
42. Gao ZM, Zhao J (2019) An improved grey wolf optimization algorithm with variable weights. Comput Intell Neurosci
43. Capilla R, Kazman R, Romera C, Carrillo C (2020) Usability implications in software architecture: the case study of a mobile app. Pract Exp Softw