IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 17 July 2022, accepted 6 August 2022, date of publication 16 August 2022, date of current version 19 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3198654

RESEARCH ARTICLE

Adaptive Binary Bat and Markov Clustering Algorithms for Optimal Text Feature Selection in News Events Detection Model

WAFA ZUBAIR AL-DYANI[1], FARZANA KABIR AHMAD[2], AND SITI SAKIRA KAMARUDDIN[2]
[1]Department of Computer Science, College of Computers and Information Technology, Hadhramout University, Hadhramout, Yemen
[2]School of Computing, College of Arts and Sciences, Universiti Utara Malaysia, Sintok, Kedah 06010, Malaysia

Corresponding author: Wafa Zubair Al-Dyani (wafazb1084@gmail.com)

ABSTRACT Wrapper Feature Selection (FS) methods based on the Binary Bat Algorithm (BBA) have recently been employed in a variety of detection applications to determine the most relevant feature subset. Despite the outstanding achievement of BBA in these domains, BBA has never been applied in Event Detection (ED). In our recent work, a novel wrapper FS approach based on BBA and Markov Clustering (MCL) method has been developed to bridge this gap and combat the curse of high dimensionality feature space for heterogeneous news text documents. However, ED from a massive number of heterogeneous news text documents with varying text lengths is a challenging task. The exploration performance of the BBA declines as the scale of the feature space grows due to the fast convergence rate problem that causes the BBA to fall into local optimum solutions. BBA's loudness (A) and emission rate (r) are significantly responsible for controlling the convergence behaviour. As a result, this study proposes two adaptive techniques for the A and r parameters to adjust BBA's convergence behavior as the dataset size changes. A new variant called Adaptive BBA (ABBA) with MCL (ABBAMCL) is proposed to improve the performance of the ED model. The ABBAMCL method has been tested over 10 benchmark datasets and two primary Facebook news datasets using several evaluation measures. The empirical results demonstrate the ability of ABBAMCL to identify a small number of informative features to detect real-world events from heterogeneous news text documents. Furthermore, with a p-value of 0.00, the statistical results show that the ABBAMCL FS method based on the proposed controlling techniques outperforms most of other FS methods in producing high-quality event clusters.

INDEX TERMS Binary bat algorithm (BBA), Markov clustering (MCL), feature selection (FS), event detection (ED), wrapper methods, heterogeneous news, text clustering, adaptive techniques.

I. INTRODUCTION

Nowadays, Event Detection (ED) from electronic text documents is a challenging task due to the rising volume of news documents available on multiple platforms such as diverse news media, forums, weblogs, emails, and Social Networking Sites (SNS) like Facebook and Twitter [1], [2]. This fact highlights the necessity for a reliable automatic ED model to substitute a time-consuming and largely human approach for matching a large number of text documents to their related real-world events. The ED task is a popular data mining research topic that focuses on automatically recognising real-world events from a variety of data streams [3]. ED was first introduced as a part of a research project known as Topic Detection and Tracking (TDT). TDT defines a topic as "a collection of events/stories that report on the same incident," while an event is defined as "a specific incident that occurred at a certain time and location," and answers questions such as "what is the event?" where and when did it happen as well as who was involved?" [1]. Generally, ED models are categorized into either (a) New Event Detection (NED) models (online ED models) or (b) Retrospective Event Detection (RED) models (offline ED models). NED aims to detect recent occurrences in online data streams, whereas RED uses

The associate editor coordinating the review of this manuscript and approving it for publication was Nazar Zaki.

an offline style to uncover significant past hidden events from a historical repository.

Over the last decade, ED has been extensively researched [4]. Surprisingly, conducting ED from digital news text documents has recently received a lot of attention as it is claimed to be a reliable source for exchanging various types of information and experience on real-world events [5], [6], [7]. Such news documents can be found on a variety of platforms, including official news websites, news feeds, digital libraries, and SNS like Facebook, Twitter, Instagram, etc. Additionally, it is conveyed that ED from multiple news sources is more effective in comparison to a single news source [8]. As a result, many academics have developed models for identifying events from heterogeneous news sources that differ in structure, written styles, language, or length [9], [10], [11], [12], [13], [14].

According to our previous survey [2], the ED model has been widely studied from heterogeneous news sources to support policymakers from various disciplines in making appropriate future decisions based on what they have experienced from previous events [15]. In addition, researchers from the news analysis area can use ED models in performing various investigations, such as determining which news channels frequently publish news articles [16], what are the most discussed events by each news channel [17], or which events people are highly attracted to and interested in sharing about them [18], which subsequently assists news channel managers in enhancing strategies for selecting the types of news events to be published in the future. Finally, the ED model can make it simple and efficient for news readers to find their desired news documents about a specific event.

Essentially, ED from text documents requires preprocessing and transformation of the documents into term-frequency vectors. This transformation raises the problem of generating feature spaces with various dimensions, such as small, medium, or high-dimensional feature spaces. Such spaces contain redundant, irrelevant, and noisy features that might mislead detection methods, resulting in a considerable reduction in the ED model's detection accuracy [19], [20]. For these reasons, most ED studies have used different traditional FS techniques to select relevant informative features, such as Term Frequency Inverse Document Frequency (TFIDF) [12], [13], [16], [17], [18] and Term Frequency (TF) [11], [21], [22]. However, these techniques have the issue of specifying a threshold in advance to select the best features. In contrast, some scholars have employed different methods like Latent Dirichlet allocation (LDA) [5], word embeddings [23], Part of Speech (POS) [24], and Named Entity Recognition (NER) [10]. However, LDA requires a predefined number of topics, whereas the word embedding method is essentially a supervised method that needs a significant amount of data to train. In contrast, POS and NER basically depend on the existing lexicons or dictionaries to recognize distinct parts of named entities or words [10], [25]. Apart from these studies, several researchers have completely ignored the FS phase and proceeded directly to the ED phase, resulting in poor ED model performance [12].

During the last decade, text mining scientists have suggested new FS methods or improved existing ones to select optimal feature subsets and enhance the model's performance [26], [27], [28]. Among such methods, wrapper FS methods based on binary versions of different Meta-Heuristic Algorithms (MHAs) have been proposed to solve FS problems in numerous data mining applications and have shown outstanding performance, such as the Binary Bat Algorithm (BBA) [29], Binary Particle Swarm Optimization (BPSO) [30], Binary Gravitational Search Algorithm (BGSA) [31], Binary Dragon Fly Algorithm (BDFA) [32], Binary Cuckoo Search (BCS) [33], etc. The wrapper FS methods based on BBA have recently demonstrated superior performance for handling FS problems in different data mining applications. Examples of such applications are detection tasks like intrusion detection [34], spam detection [35], community detection [36], e-fraud detection [37], anomaly detection [38], etc. However, such applications are still in the early stages of development, and further investigation into them is required, as well as other real NP-hard applications should be considered [39], [40], [41]. Despite the outstanding achievement of BBA in these domains of research, BBA has never been applied to ED [2], [42]. In our recent work [43], we developed a novel wrapper FS approach based on BBA and the Markov Clustering (MCL) method to effectively overcome this gap and combat the curse of high dimensionality feature space for heterogeneous news text documents. The essential concept was to wrap the BBA with the MCL method. MCL is a well-known graph ED method that has been used successfully in the ED domain [44], [45], [46], [49].

The experimental results as well as the statistical measures have validated the superiority of BBAMCL in detecting highly accurate real-world events from multiple heterogeneous news text documents compared to other benchmark FS methods. However, on certain datasets, the BBAMCL has shown slightly poor performance. The reason behind such poor performance was the early convergence rate of BBA, which prevented it from effectively exploring the whole feature space. The initial values of two BBA parameters, pulse emission rate (r) and loudness (A) [50], are primarily responsible for this convergence behavior. The study in [43] assigned predefined fixed values for such parameters that may not be ideal for all datasets. As a matter of fact, the application domain, scope, and the size of the given datasets all influence the A and r parameters' values [51]. The optimal values for the different BBA parameters are still being investigated, and it is very difficult to determine their best values [52], [53]. To address this problem, this study aims to improve BBA's convergence behavior to enhance the performance of the previously proposed wrapper BBAMCL FS method. Apparently, this research proposes new adaptive techniques for controlling BBA's A and r parameters automatically. As a result, a new variant called Adaptive BBA (ABBA) was developed. It is utilized to select the optimal feature subset for identifying the events from multiple heterogeneous news documents of varying lengths.

The rest of the paper is organized as follows. Section II reviews and discusses the most related studies. Section III describes the materials, and the methodology proposed in this study is described in Section IV. The parameter settings for the applied methods are illustrated in Section V. Experimental results are offered in Section VI, followed by a discussion in Section VII. Finally, conclusions are stated in Section VIII.

II. RELATED WORKS

Various wrapper FS methods based on MHAs have been developed to solve the optimal FS problem in different data mining applications [54], [55], [56]. Such applications involved data classification, data clustering, diseases detection, spam detection, intrusion detection, and many other applications [56], [57], [58], [59]. MHAs for instance Particle Swarm Optimization (PSO) [60], Genetic Algorithm (GA) [61], Bat Algorithm (BA) [59], Grey Wolf Optimizer (GWO) [62], Krill Herd (KH) [63], Cuckoo Search (CS) [64], Butterfly (BF) [65], Dragonfly Algorithm (DFA) [66], Firefly Algorithm (FA) [67], Ant Colony Optimization (ACO) [68], Artificial Bee Colony (ABC) [69], Whale Optimization Algorithm (WOA) [70] have been widely used by researchers in solving FS problems. In spite of numerous works in FS area, wrapper FS methods based on binary variations of MHAs have been widely established in the literature. Methods such as Binary Bat Algorithm (BBA) [37], [71], [72], [73], Binary Particle Swarm Optimization (BPSO) [74], Binary Butterfly (BF) [75], Binary Ant Lion (BAI) [76], Binary Harris Hawks (BHH) [77], Binary Gravitational Search Algorithm (BGSA) [31], Binary Grey Wolf Optimizer (BGWO) [62], Binary Artificial Bee Colony (BABC) [78], Binary Dragonfly Algorithm (BDFA) [32], Binary Firefly Algorithm (BFA) [79], Binary Butterfly (BBF) [75], Binary Salp (BS) [80], Binary Flower Pollination Algorithm (BFPA) [81], Binary Cuckoo Search (BCS) [33], Binary Whale Optimization Algorithm (BWOA) [82], Binary Krill Herd (BKH) [83] have been proposed by many studies. This is because the FS problem is stated to be a binary optimization problem [41], [55], [84], [85]. These methods have achieved extraordinary results when compared to the continuous versions of MHAs.

Despite the impressive results of binary FS methods, most of them have been applied to solve FS problems in classification applications, while just a few have been utilized in clustering applications, as illustrated in Table 1. This occurred since FS in unsupervised learning (clustering) is more complex and challenging than in supervised learning (classification) [85]. The availability of labelled data in classification applications makes the evaluation process for FS methods much easier compared to the clustering applications where no annotated data is presented. Additionally, most of the studies have been tested on instances with a small number of features (see Table 1). Accordingly, many methods have yet to be tested to solve FS problems in real-world applications with high-dimensional feature spaces and sparse or noisy features. Hence, there is still a significant scalability gap in the FS methods, which needs to be addressed in the future [84], [86], [87]. Inclusively, several researchers have pointed out that the future direction in the FS field is to improve the performance of the existing FS methods rather than develop new MHAs [55], [88], [89].

Numerous researches have shown that BBA is superior to other binary MHAs in addressing FS problems and achieving impressive results [34], [35], [37], [41], [71], [72], [73], [84], [85], [88], [90], [91], [92], [93], [94], [95], [96], [97], [98], [99], [100], [101], [102], [103], [104], [105], [106], [107], [108], [109], [110], [111], [112], [113], [114]. In context of classification, Nakamura *et al.* [90] proposed a wrapper FS method that integrates the basic BBA with the speed of the Optimum-Path Forest (OPF) classifier to find the optimum feature subset that maximizes the classification accuracy. Apart from OPF, different classifiers have been utilized by many scholars to introduce various wrapper FS methods, including BBA with K-Nearest Neighbor (KNN) and BBA with Naïve Bayes (NB) [92], BBAKNN [101], BBA-KNN and BBA-Support Vector Machine (SVM) [100], and BBA-NB [94]. Some researchers have used various types of transfer functions to test the standard BBA. For instance, Rodrigues *et al.* [91] applied BBA with two transfer functions (e.g., sigmoid and hyperbolic tangent) to determine the best feature set that could increase the classification accuracy. Similarly, different variants of BBA based on six different transfer functions were investigated in [103].

On the other hand, a number of researchers have introduced various improvements into BBA to enhance the classification accuracy. Table 1 shows various type of classifiers that have been employed with BBA over the past years, such as SVM, KNN, Linear Regression (LR), Decision Tree (DT), and Random Forest (RF) in different kinds of applications. For instance, BBA based on Greedy Crossover (GC) with SVM [111], a sub-population technique was utilized to enhance the BBA's performance in the Niche BBA-KNN method [109]. In [102], Cross Entropy (CE) was embedded into BBA to increase the diversity of solutions, while modified velocity and position equations with mutation operators were introduced to enhance the performance of the classification process [88]. Additionally, a hybrid FS method was developed by Taha *et al.* [113], which first applied the Mutual Information (MI) technique to rank the features. Subsequently, the previously developed BBA-NB [94] was employed to find the optimal feature set. In contrast, Tawhid and Dsouza proposed a hybrid BBA-Enhanced PSO-KNN for improving classification tasks [84].

For enhancing the detection of white blood cells, wrapper FS methods based on standard BBA with various classifiers (e.g., LR, DT, KNN, and RF) were utilized [71]. Enache and Science [34] adopted the FS method introduced in [93] and integrated it with SVM and NB to create two wrapper FS methods for improving the accuracy of the intrusion detection model. Similarly, BBA [108], [110], and its improved versions [97], [98], [99] were integrated with different classifiers to introduce various wrapper FS methods for improving the performance of the intrusion detection models. In addition, Rajalaxmi and Ramesh [35] utilized the basic BBA with JRip for effective spam detection on Facebook.

Moreover, Janani *et al.* [107] proposed a hybrid GA-BBA with two classifiers named NB and KNN for better spam detection. Breast cancer detection was improved using wrapper FS methods based on basic BBA with various classifiers such as Feedforward Neural Network (FNN) [95], SVM, J48, and NB [104]. For the same detection task, the Correlation Feature Selection (CFS) filter technique was hybridized with BBA based on GC and SVM [105], while standard BBA was integrated with SVM [96] and Extreme Learning Machine (ELM) [106]. On the other hand, BBA-LibSVM was employed to select an optimal feature subset for E-fraud detection [37]. Last but not least, BBA was integrated with DT, RF, and KNN were applied to improve the detection of seizure episodes [112].

Regardless various works in this area, BBA in classification task is still a straightforward task since the data trained in these models are pre-labelled. In contrast, few studies have been observed to work on BBA for clustering tasks. Researchers such as BBA-Kmeans [85], BBA-Kmeans based on modified velocity equation and mutation operator [72] as well as a tunned BBA-Kmeans with KNN [73] were developed for high clustering performance. Apart from classification and clustering tasks, some scholars have utilized BBA to solve FS problems in other applications. For instance, Xingwang Huang *et al.* [41] proposed their enhanced BBA, named the Dynamic Inertia Weight Binary Bat Algorithm (DIWBBA) algorithm. The enhancements include introducing an adapting dynamic inertia weight strategy to control the magnitude of the velocity of BBA to increase the convergence rate and achieve the right balance between exploration and exploitation. The experimental results confirm the effectiveness of their algorithm over five zero-one knapsack datasets and thirteen benchmark functions.

In spite of the good results that previous studies have achieved, several limitations have been identified. Such problems need to be investigated further. For instance, many researchers have employed wrapper FS methods based on BBA with various classifiers for classification applications, but just a few studies have focused on clustering applications. This could be because FS for unsupervised clustering learning is more difficult than FS for supervised classification learning [85]. Furthermore, applications for BBA in real NP-hard applications (e.g., clustering, anomaly intrusion detection, spam classification, WBC classification, breast cancer classification, etc.) are still in their infancy. This implies that more research needs on these applications and other real NP-hard applications is required [39], [40], [41].

Furthermore, most BBA studies have been conducted on small datasets with limited features. According to Brezocnik *et al.* (2018), datasets with up to 150 features are considered to be small datasets (low-dimensional datasets) and datasets with more than 2000 features are large datasets (high-dimensional datasets). According to Table 2.6, the majority of BBA studies tested their methods on small datasets [34], [35], [41], [71], [72], [73], [84], [85], [93], [94] [95], [97], [98], [102], [104], [108], [110], [113] and medium datasets with feature greater than 150 and less than 2000 [88], [90], [91], [92], [100], [107], [109], [111], [112]. On the other hand, several research papers have examined large datasets [37], [96], [99], [101], [103], [105], [106]. Despite the variation in data sizes, all previous research employed pre-defined values for the two most critical BBA parameters: A and r. According to the literature, such settings may work well for some datasets but not for all, as the values of r and A mostly depend on the scale of the dataset and the application's domain [39], [52]. Real-world applications, like text mining applications, typically have large datasets with a high number of features (greater than 2000 features). Thus, more investigation is required to develop techniques to find the optimal settings for the A and r parameters [75], [84], [87], [124], [125], [126]. To solve this issue, many researchers in the literature have introduced various techniques for tuning or controlling different BBA's parameters.

Setting up the parameters of BBA has a significant impact on its behavior when searching for promising regions, as well as affects the quality of final solutions [53], [127]. Consequently, tuning and controlling parameters become optimization problems [128]. In the literature, many parameter setting up methods have been introduced. According to Parpinelli *et al.* [127], who adapted and merged the classifications introduced in [129] and [130], the methods for setting parameters are classified into two major categories: tuning (offline) and controlling (online) techniques. On the one hand, parameter tuning techniques examine different values for parameters prior to algorithm implementation. The main goal is to determine the best setting values for the parameters, which can then be utilized as recommendations for any future runs. Tuning parameters typically use static values or previously adjusted values for parameters. This category is further divided into three types: 1) manually setting up parameters (the trial-and-error method); 2) carrying out pre-planned experiments; and 3) applying meta-heuristic algorithms. Manual setting is based on the user's previous experience, in which they adjust the parameter values before each execution. Experimental planning involves setting up tests for a group of parameters' values first, then identifying the best configuration through analyzing the obtained results. The meta-heuristic setting technique is based on changing the parameter values using an algorithm. However, tuning parameters with these techniques has been found to be time-consuming due to the large number of values that each parameter must be tested until the optimal value is identified [131], [132].

On the other hand, controlling parameter techniques adjust the values of the parameters as the algorithm executes.

TABLE 1. Summary of BBA related works.

Studies	Method	Predefined A & r	Data Type	#Max Number of Features	Application	Data Mining Task
[84]	Hybrid BBA+EPSO + KNN	√	20 Real world datasets from UCI	60	Not specified	Classification
[88]	MBAFS+SVM	√	12 Real world datasets from UCI	166		
[90]	BBA+ OPF	√	5 Real world datasets from UCI	180		
[91]	BBA+ OPF	√	6 Real world datasets from UCI	256		
[92]	BBA+ KNN & BBA+ NB	√	2 Textual Chines datasets	1500		
[94]	BBA+NB	√	9 Real world datasets from UCI & 3 datasets from [115]	56		
[100]	BBA+ KNN and SVM	√	10 Real world datasets from UCI	856		
[101]	BBA+KNN	√	5 Real world datasets from UCI & 2 datasets from [116]	7129		
[102]	BBA_CE +KNN	√	10 Real world datasets from UCI	60		
[103]	BBA + Classifier (not mentioned)	√	3 Datasets: Antimicrobial agents [117], Anti-hepatitis C virus activity [118], Neuraminidase inhibitors of influenza A viruses [119]	3657		
[109]	Niche BBA + KNN	√	3 Real world datasets from UCI	500		
[111]	BBA_GC + SVM	√	4 Real world datasets from UCI	166		
[113]	MI+BBA+NB	√	12 Real world datasets from UCI	56		
[34]	BBA[94]+NB & SVM	√	NSL-KDD dataset [120]	80	Intrusion Detection	Classification
[93]	BBA+SVM	√		41		
[97],[98]	BBA(E)+SVM and C4.5	√		41		
[99]	BBA_E and BBA_L + SVM, C4.5, and NB	√		7129		
[108]	BBA+NB, BBA+J48	√	KDDcup99 dataset	80		
[110]	Hybrid BBA+DNN	√	CICID2017 dataset [120]	80		
[35]	BBA+ JRip	√	Facebook users' profiles dataset	18	Spam Detection	Classification
[107]	Hybrid GA_BBA+ NB and KNN	√	Chicago hotel's reviews [121]	1771		
[95]	BBA+ FNN	√	WBC dataset from UCI	32	Breast Cancer Detection	Classification
[104]	BBA + SVM, J48, and NB	√	Breast cancer dataset from UCI	10		
[96]	BBA+SVM	√	10 gene datasets from UCI	24481	Genes Cancer Detection	Classification
[105]	Hybrid CFS+ BBA_GC + SVM	√	Cancer microarray datasets from UCI	24481		
[106]	BBA+ ELM	√	DNA microarray datasets from UCI	10368		
[37]	BBA+ LibSVM	√	20 e-Fraud datasets from UCI	4508	e-Fraud Detection	Classification
[71]	BBA+ LR, DT, KNN, and RF	√	WBC datasets from [122]	35	White Blood Cell Detection	
[112]	BBA+DT, RF, and KNN	√	One real world dataset from UCI	179	Detection Seizure Episodes	Classification
[72]	BBA+Kmeans	√	4 Real world datasets from UCI & 4 datasets from [123]	60	Not Specified	Clustering
[73]	BBA+Kmeans and KNN	√	4 Real world datasets from UCI	60		
[85]	BBA_Kmeans	√	4 Real world datasets from UCI	60		
[41]	DIWBBA	√	13 Benchmark functions + Knapsack problem	75	13 Benchmark Functions + Zero Knapsack Problem	

In other words, the designers of the algorithms exclude themselves from the responsibility of modifying such parameters. Controlling parameter techniques include a number of significant properties that make them more appealing to a wide number of researchers working on enhancing MHAs [127], [131], [133]. For instance, parameter values do not need to be initialized before running the algorithm, and their values can change based on feedback from the search process. It consumes less computing time to find optimal solutions and avoid falling into local optimal solutions, which improves the quality of the results. Similar to the tuning parameter category, the controlling parameter category is also classified into three types: deterministic (rule-based); adaptive (feedback-based); and self-adaptation (aggregated control) [134]. Deterministic control techniques modify the parameter values during algorithm implementation without relying on feedback from the search process. In contrast, adaptive control techniques exploit feedback information from the search process to determine how the value of the parameter must be adjusted. Self-adaptive control techniques encode the search space into parameter values, and the parameter is directly coded in the solution vector as extra dimensions and later adjusts itself throughout the optimization process.

Adaptive control techniques consist of several techniques. Simple rules, fuzzy control, learning automata, and entropy are a few examples. Among these techniques, deterministic and simple rule techniques have been widely used for setting up parameters of different MHAs [127]. This is due to their simplicity and low computational effort [130], [127]. Deterministic control techniques employ a rule to adjust the parameter value, which updates the parameter value without relying on the search space's feedback [130]. Literally, a predefined function is employed to modify the parameter value throughout iterations, e.g., the parameter value starts with a small value and gradually grows during the iteration process. In the case of simple rule control techniques, the parameter values are adjusted based on simple rules that are established based on the feedback information. This information could be fitness value, iteration number, diversity measurements, etc. [130].

As it is reported in Table 1, almost all studies have utilized the auto-zooming feature in BBA with predefined values for A and r parameters. This is a limitation as the values for such parameters could be a good choice for some applications but not for others because the optimal values basically depend on the application domain and the scale of the datasets used [39], [53]. Among the studies in Table 1, Alam [73] utilized a tuning trial and error technique to set up three parameters: A, r, and f to boost the exploration ability of BBA. However, tuning techniques are reported to be time-consuming and difficult to apply as many values must be investigated before the ideal values are recognized [127], [131], [135], [136]. Utilizing controlling techniques, Taha *et al.* [94] introduced an adaptive simple rule to control the maximum values for the velocity and A parameters to increase the exploration process of BA. However, Taha *et al.* [94] have only concentrated on adjusting A while ignoring the r parameter, despite the fact that the interaction of both parameters is critical for achieving a fair balance between BBA exploration and exploitation processes [71], [137], [138], [139]. Xingwang Huang *et al.* [41] applied an adaptive inertia weight to control the degree of BBA's velocity to obtain the right balance between global search and local search. As a result, a new variant has been introduced called DIWBBA. However, the authors have totally neglected the two most influential factors (A and r parameters) in obtaining the optimal balance.

To highlight, several techniques have been designed in the literature to tune or control A and r parameters [125], [131], [137], [140], [141], [142], [143], [144], [145], [146], [147], [148]. However, these techniques have been developed to enhance the performance of the continuous version of BA, which has been applied to solve the FS problem for datasets that are not textual. As a matter of fact, BA differs from BBA in the representation of the feature space and the selection mechanism for the optimal feature subset [84], [149]. To illustrate, a threshold has to be defined in advance to select the best features for the continuous version BA, while BBA does not need this as each selected feature is represented by 1 and 0 indicates that the feature is not chosen [149].

Due to the variance in the representation and the utilized datasets, there is no guarantee that existing tuning and controlling techniques for BA will enhance the performance of the BBA in finding the optimal feature subset for achieving accurate detection of events [84], [149], [150], [151]. This is mostly because of the No-Free Lunch theorem [152], which states that there is no universal algorithm to solve all kinds of optimization problems. In other words, there is no one algorithm that can solve FS problems for all datasets [153]. To the best of our knowledge, BBA has not yet been applied to solve FS problems in the ED field of application. This scenario motivates this study to adapt the DIWBBA [41] by designing two adaptive simple rule techniques to control A and r parameters for improving the exploration ability of the DIWBBA and balancing it with the exploitation process. Such techniques prevent DIWBBA from falling into local optimum solutions and overcome the premature convergence problem. As a result, a new variant was developed, which is called Adaptive BBA (ABBA). Subsequently, this research used ABBA to construct the wrapper FS method to solve the FS problem of the introduced ED model in this study and obtain highly accurate detection of events.

III. MATERIALS

A. BAT ALGORITHM (BA)

The basic Bat Algorithm (BA) was designed by X. S. Yang [154], who idealized the echolocation property of microbats and converted it into a numerical optimization algorithm of four main steps as follows:

- A population of bats (solutions) is initialized using randomly selected values from a set of real numbers according to (1):

$$x_{ij} = x_{min} + \varphi(x_{max} - x_{min}) \tag{1}$$

where $i = 1, 2 \ldots N$, $j = 1, 2, \ldots d$, x_{min} and x_{max} are the lower and higher borders for dimension *j,* respectively. φ is a randomly selected value from [0,1].

- For every single bat, it has frequency f_i, velocity v_i, and position (solution) x_i which are updated during the iterations. Hence, the new frequency f_i^t, velocity v_i^t, and position (solution) x_i^t at time step t are computed using (2), (3), and (4):

$$f_i^t = f_{min} + (f_{max} - f_{min})\beta, \tag{2}$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*)f_i^t, \tag{3}$$

$$x_i^t = x_i^{t-1} + v_i^t, \tag{4}$$

where β is a random number selected from [0,1]. x_* is the current global best solution for all N bat that has been found so far. In addition, each bat has a frequency uniformly chosen from $[f_{min}, f_{max}]$. To improve the local search, each bat walks randomly around the best solution that was found. This way, a new solution for every single bat is generated locally using (5):

$$x_{new} = x_{old} + \varepsilon A^t, \tag{5}$$

where ε is a random number selected between [−1,1], and $A_t = < A_i^t >$ denotes the average loudness of all bats at t iteration. x_{old} is the best solution identified using Equation 1.

- At the beginning, different initial values are assigned to the r and A parameters of BA using random techniques. After that, at every iteration, both parameters r and A are adjusted utilizing (6) and (7), where this happens only if the new solutions are improved.

$$A_i^{t+1} = \alpha A_i^t, \tag{6}$$
$$r_i^{t+1} = r_i^0 \left[1 - \exp(-\gamma t)\right] \tag{7}$$

where γ and α are constants; for any $0 < \alpha, \gamma < 1$.

- If the obtained solutions satisfy the given condition, then these solutions are saved as the best solutions. Finally, all bats are ranked to identify the current best solution (x_*).

Algorithm 1: Bask Bat Algorithm (BA)

Input: Original feature sets, initialize parameters of BBA (population size (n), number of features (dim), max number of iterations for BBA (T_{ba}), loudness (A), pulse emission rate (r). alpha (α), gamma (γ). Beta (β). maximum frequency (f_{max}). minimum frequency (f_{min})
Output: Optimal feature subset that gives the highest objective function value

Objective function: $f(x), x = (x_1, \ldots, x_d)^T$
Initialize the bat population: $X_i (i = 1, 2, \ldots, n)$ and v_i
While ($t < T_{ba}$)
  Generate new solutions by adjusting frequency, and updating velocities and locations/solutions [equations (2) to (4)]
  IF (rand > r_i)
    Select a solution among the best solutions
    Generate a local solution around the selected best solution [equation 5]
  End If
  Generate a new solution by flying randomly
  IF (rand < A_i & $f(x_i) < f(x*)$)
    Accept the new solutions
    Update r_i and reduce A_i [equations 6 and 7]
  End If
  Rank the bats and find the current best solution (x∗)
End While
Postprocess results and visualization

Algorithm 1 [154] shows the important steps of the standard BA. In many applications, BA has been utilized as an optimum FS technique to select optimal features [35], factor values [155], or values [156] from a range of options in order to improve the performance of a model or a system. Many studies [39], [84], [85], [157], [158] have demonstrated the effectiveness of BA in outperforming various existing benchmark MHAs. The reason for this achievement is the diversity of solutions and frequency variations [159]. Furthermore, all BA parameters can be updated during iterations, whereas the majority of MHA parameters are fixed [53], [94], [145], [160].

Moreover, BA can automatically zoom into the optimal solution region with local intensive exploitation. In contrast, many MHAs lack this ability [71], [137], [159]. Finally, BA is highly effective for obtaining good solutions to various complex and optimization problems in a short period of time due to its fast convergence rate. Having such key advantages, BA has been widely used to select the best features in a variety of data mining applications, including clustering, truck and trailer routing problems, community detection, spam detection, e-fraud detection, and image processing [43], [131], [159], [161], [162], [163], [165]. However, there is a great lack of research on applying BA to solve FS problems in text mining applications like ED or TDT [39], [40]. Such applications appear to be popular and active research topics [1], [4], [166], and are distinguished by the high dimensionality of their feature spaces, on which BA has not been well validated [75], [84], [125], [159].

Several changes have been made to the BA structure to address issues like the fast convergence rate and poor exploration ability. This has resulted in emergence of numerous variants [39], [40], [53], [167], [166], [167], [168], [169], [170], [171], [172]. The BBA [37], [71], [88], [90] is one of the most well-known variations of BA for handling FS problems in different data mining applications.

B. BINARY BAT ALGORITHM (BBA)

Nakamura et al. [90] introduced a binary version of BA called BBA. The BBA has similar steps as the basic BA but with a slight variance in the update solution equation (4). In other words, (4) is replaced with binary vectors by applying a sigmoid function (8):

$$S(v_i^j) = \frac{1}{1 + e^{-v_i^j}}, \tag{8}$$

Thus, new BA's solution is modified using (9):

$$x_i^j = \begin{cases} 1 & if\ S(v_i^j) > \sigma \\ 0 & if \text{ otherwise} \end{cases} \tag{9}$$

where (1) indicates that the feature is selected and (0) represents that the feature is not discarded, where $\sigma \sim U(0, 1)$.

Due to the similarity of their structures, BBA has the same characteristics and advantages as BA. However, BBA has shown promising results in solving FS problems in a variety of real NP-hard applications compared to the basic BA [4], [48] and other binary MHAs [35], [37], [99], [101], [173], [174], [175], [176], [177], BBA As a result, BBA has emerged as a promising solution to FS problems in a wide range of data mining applications [34], [35], [36], [37], [38]. For the same reason, a wrapper FS method based on BBA and MCL has been proposed in our previous work [43] to increase the accuracy of the ED model for heterogeneous textual datasets. However, BBA-MCL has not performed well in several datasets due to the use of fixed values for BBA's A and r parameters. This causes BBA to converge quickly and fall into local optimum solutions for certain datasets. To overcome this problem, this study developed a new variant of BBA based on adaptive techniques for controlling A and r parameters.

C. ADAPTIVE BINARY BAT ALGORITHM (ABBA)

This section presents a complete description of the proposed ABBAMCL to solve the FS problem for the ED model. The ABBA algorithm in the proposed ABBAMCL FS method has the same steps as the original BBA. However, several modifications were introduced to increase the convergence rate as well as improve the exploration and exploitation capabilities of BBA. As a result, a new variant of the BBA was introduced, named ABBA. The changes were made to three parts of the original BBA: 1) the velocity equation; 2) accepting new generated solution conditions; and 3) updating the r and A equations. (1) and (2) were inspired by Xingwang Huang *et al.* [41] and Chakri *et al.* [125], respectively. As for the third improvement, it was proposed by this study. The key improvements and features of ABBAMCL are described as follows:

D. UPDATE VELOCITY EQUATION

The velocity update equation (3) consists of two parts. The first part v_i^{t-1} represents the velocity of the population, while the second part $(x_i^{t-1} - x_*)\, f_i$, controls the velocity of the ith position x_i^{t-1} with the guidance of the global best solution x_*. These two parts affect the global and local search of the BBA, as the first part is reported to decrease the convergence rate, while the second part leads to a premature convergence problem [178]. Some BBA variants have recently been introduced to address this issue [41], [148], [178], [179], [180]. Xingwang Huang *et al.* [41], Yılmaz and Küçüksille [87] proved that BBA can generate better solutions with the help of the neighbor bat (*kth* solution). For this reason, inspired by Xingwang Huang *et al.* [41], the velocity update equation of the original BBA was modified using (10) and (11):

$$v_i^t = \omega v_i^{t-1} + (x_i^{t-1} - x_*) f_i S_1 + \left(x_i^{t-1} - x_k^t\right) f_i S_2 \quad (10)$$

$$S_1 + S_2 = 1, \quad (11)$$

where ω represents the dynamic inertia weight strategy which controls the v_i^t of the ith bat, x_k^t denotes one of the best solutions arbitrarily chosen from the population (i != k), S_1 is a self-adaptive learning parameter for the global best solution x_*, while S_2 is a learning parameter of *kth* solution. S_1 ranges from 0 to 1 and S_2 ranges from 1 to 0. In this modification, the kth solution information is employed to guide the kth solution to avoid BBA from falling into local optimum solution. To illustrate, as S_1 is enlarged, the effect of the x_* solution becomes better than the x_k^t and vice versa. The update formula for S_1 is given in (12):

$$S_1 = 1 + (S_{init} - 1)\left(\frac{\text{iter}_{max} - \text{iter}}{\text{iter}_{max}}\right)^n \quad (12)$$

where S_{init} indicates the initial impact value of S_1, $iter_{max}$ denotes the maximum number of iterations, *iter* denotes the current iteration, and n represents a nonlinear modulation index. S_1 is nonlinearly increased from S_{init} to 1 as *iter* is increased meanwhile S_2 is reduced from $(1 - S_{init})$ to 0, correspondingly. The small S_1 and large S_2 allow bats to discover the whole search space instead of flying in the direction of the best bat. In contrast, a large S_1 and a small S_2 let the bats reach the global best solution in the later stages of the search process. The modification to the velocity update equation helps in enhancing the global search and improving the convergence of the BBA into the global best solution during the final stages of the optimization. As for ω, it was inspired by Lei and Pu [181] to control the degree of the velocity for the *ith* bat and it is computed using (13):

$$\omega = \omega_{max}\left(\exp\left(-m \times \left(\frac{\text{iter}}{\text{iter}_{max}}\right)^m\right)\right) \quad (13)$$

where *iter* represents the current iteration, $iter_{max}$ denotes the maximum number of iterations, ω_{max} indicates the maximum value of inertia weight, and m is a constant number greater than 1. To automatically shift from global search to local search, Xingwang Huang *et al.* [41] introduced an adaptive strategy that states BBA shifts to extensive exploitation with a small value of m If the current global best solution is not enhanced after c iterations or continues the exploration search with the present m. The strategy is illustrated by (14):

$$m = \begin{cases} m & x_*^{i+c} \geq x_*^i \\ s_1 s_2 m & \text{otherwise} \end{cases} \quad (14)$$

where c denotes the defined interval of iterations while x_*^{i+c} and x_*^i indicate the $(i+\text{c})th$ and *ith* values of global best solution x_*, respectively. This modification contributed to the dispersion of the solutions obtained by BBA from binary search space and hence achieved more accurate event clusters.

E. ACCEPT NEW GENERATED SOLUTION CONDITION

Acceptance of a new solution involves the satisfaction of two conditions. First, the solution must generate an objective value larger than the global best solution (i.e., for the maximization problems). Second, a randomly produced number must be lower than the current corresponding A. In some cases, the movement of the bat may generate a solution better than the global best solution, but it cannot be accepted because the randomly produced number is greater than the current A. Especially near the end of the iterative process, where the value of A becomes low. For that reason, and inspired by Chakri *et al.* [125], an amendment was made to allow the BBA to accept the new solution whenever it is better than the global best solution, even if the randomly generated number is not lower than A's value, as it is given in (15):

$$\textit{if}\,(\text{rand} < A_i)\ \text{OR}\ (f\,(x_i) < f\,(x_*)) \quad (15)$$

F. UPDATE A AND R EQUATIONS

Tuning the parameters of an algorithm is time-consuming [131]. Thus, to avoid the trial-and-error procedure for choosing the appropriate values of r and A parameters of BBA, an ABBA is proposed in this study, which has at least two advantages:

- No predefined setting values are needed for A and r before the ABBA run is begun.

- The A and r parameters are updated throughout the ABBAMCL's run based on the number of the feature vectors within the search space of the given dataset and the iteration process.

The equations (6) and (7), which were introduced by X. S. Yang [154], have been used to update A and r and cause them to reach their final values during the iterative process rapidly. As a result, the possibility of exploring all unseen features is reduced due to a higher r rate and low A. Therefore, this study proposed adaptive techniques for increasing and decreasing r and A monotonically, as depicted in (16) and (17):

$$r_i = \begin{cases} r_i + \left(\dfrac{1}{\mathrm{FV}} \times it\right) & \text{it} \leq 50 \\ r_i + \left(\dfrac{r_i}{it}\right) & \text{otherwise} \end{cases} \tag{16}$$

$$A_i = \begin{cases} A_i - \left(\dfrac{1}{\mathrm{FV}} \times it\right) & \text{it} \leq 50 \\ A_i - \left(\dfrac{A_i}{it}\right) & \text{otherwise} \end{cases} \tag{17}$$

where A_i and r_i are the loudness and emission rate for i*th* bat in the population, *it* represents the current iteration, and FV denotes the number of feature vectors in i*th* iteration. Initially, ABBA starts to perform a global search to explore the feature search space more effectively. This strategy was achieved by assigning a low value to r_i for the first 50 iterations. However, this low value should not be continued. Thus, as the iterations approached the end (100 iterations), a large value was given to the r_i, so bats could shift into the exploitation stage.

On the other hand, A_i accepts or rejects the new produced solution. Its importance appears when it discards some solutions; it prevents BBA from falling into local optimum solutions and dodges premature convergence. Therefore, a large value has been given to A_i for the first 50 iterations to increase the chances of accepting some solutions. Consequently, allow bats to explore the feature more thoroughly to avoid becoming trapped in the local optima solution. In contrast, a small value has been given to A_i for the later iterations, so that more new generated solutions are accepted and allow the bats to converge into the optimal solution. The initial values for r_i and A_i have been defined using (18) and (19) as follows:

$$A_{i(init)} = 1 - \left(\frac{1}{\text{Number of feature vectors}}\right) \tag{18}$$

$$r_{i(init)} = \left(\frac{1}{\text{Number of feature vectors}}\right) \tag{19}$$

To sum up, this study proposed simple adaptive techniques to control the r and A parameters so ABBA can explore the whole feature space more effectively and avoid getting stuck in local optimum solutions at early stages.

G. MARKOV CLUSTERING ALGORITHM (MCL)

The MCL algorithm was introduced by Van Dongen [182]. MCL was initially used in bioinformatics to identify protein interactions, sequence analysis, and determination of protein function [183], [184]. Later, MCL has been widely used as a clustering method in a variety of fields, including ED [45], [46], [47], [48], [49], spam detection [185], [186], video processing [187], image processing [188], etc. MCL is well-known in the ED field as a popular dynamic graph ED method that set apart from other graph-based methods due to its several advantages [44], [45], [183], [189]. For instance, MCL is simple to use, works well with both weighted and unweighted graphs, produces well-balanced non-hierarchical clustering results, and can handle noisy data within a graph. On top of that, it is faster than other graph-based methods and does not require determining the number of clusters in advance. This is very vital when dealing with the ED task, as the number of events cannot be predicted in real life. Given such features, MCL has been used with BBA as a wrapper FS method to improve ED's performance in our previous work [43] as well as in this study. Initially, MCL walks randomly through a graph G and does not leave the cluster until it visits many nodes within the cluster. Algorithm 2 [183] shows the pseudocode of the MCL algorithm.

Algorithm 2: Basic Markov Clustering Algorithm (MCL)

Input: Undirected weighted graph G, initialize parameters of MCL, pruning threshold (p), expansion (exp), inflation (inf).
Output: Clustering matrix M into Clusters

```
1   Create the adjacency matrix A of graph G
2   Add self-loop to graph G, A = A +1
3   Normalize each column of matrix A, to get Markov M
4   M = AD -1, where D is the matrix degree diagnosis of A
5   Repeat steps 6 and 11 until a steady state is reached (convergence):
6   |   M = Expansion (M): = M * M
7   |   M = Inflation (M. ∧ inf)
8   |   M = Prune (M) using p
9   |     | Mij = 0 if Mij < minval
10  EndSteps
11  Interpret resulting matrix M to identify the clusters
```

MCL implements three main steps, namely expansion, inflation, and pruning. In the expansion step, new flows are opened, and more flows are increased between the existing nodes within the transition probability matrix M. In other words, expansion aims to generate new non-zero values in M by introducing new edges and removing the unwanted old ones from the graph. On the other hand, in the inflation step, every single element of the matrix M is raised by the inflation parameter r. This operation is known as the Hadamard operation, and it is calculated using (20):

$$\text{Minf} = \text{Inflate}\,(M, r) = \frac{M\,(i.j)^r}{\sum_{k=1}^{n} M\,(k, j)^r} \tag{20}$$

This step tends to strengthen the strong edges and break down the weak edges. Finally, the pruning step is carried out after the inflation operation within each iteration in order to save memory. Whereby, it eliminates the weak edges and removes zero values from the M matrix. In summary, expansion, inflation, and pruning steps are iteratively employed until there is no more change in M's elements or there are slight changes from the elements in the previous matrix of MCL, i.e., till M is converged. As a result, the given graph is partitioned into many clusters without any overlapping.

IV. THE PROPOSED METHODOLOGY

Typically, the ED model has five main phases: (1) data preparation phase, (2) text preprocessing phase, (3) FS phase, (4) ED phase, and (5) evaluation phase. The implementation of the experiment for this study was done using Python (version 3.7) on a machine with Windows 10 and 8GB of RAM. The next subsections describe every single phase in the ED model as well as the parameter settings for the applied methods.

A. DATASETS PREPARATION PHASE

To assess the effectiveness of the proposed wrapper ABBAMCL FS method in terms of ED performance, 12 datasets were used. Specifically, 10 secondary datasets (benchmark news datasets) and 2 primary datasets (Facebook news posts) were used (see Table 2). These datasets were shaped from five main datasets (20newsgroup, news aggregators, RSS news feeds, news articles, and Facebook news posts) as being done by studies in [13], [61], [190], [191], [192]. The real reason behind this formation is to generate textual datasets that are sparse and represent feature spaces with various dimensionalities, such as low, medium, and high-dimensional feature spaces. In addition, such datasets were employed to test and evaluate the performance of the proposed ABBAMCL method with different numbers of documents, features, and events. To highlight, stratified random sampling was used to determine the number of documents and events, while TFIDF was applied to obtain the number of features.

TABLE 2. Characteristics of the text news datasets.

Dataset name	Short name	#Documents	#Features	#Events
20newsgroup	DS1	100	2894	5
20newsgroup	DS2	100	2529	10
20newsgroup	DS3	100	2823	20
20newsgroup	DS4	200	4637	10
20newsgroup	DS5	200	5383	20
20newsgroup	DS6	300	6556	20
News aggregator	DS7	800	1085	4
News aggregator	DS8	2000	2106	4
News articles	DS9	1467	14770	56
RSS news feed	DS10	2095	3818	56
Facebook news posts	DS11	1139	3742	33
Facebook news posts	DS12	1074	3420	16

To illustrate, six distinct datasets (DS1-DS6) were created from 20newsgroups. DS1 includes 100 random documents that belong to 5 events. DS2 contains 100 random documents that were assigned to 10 events. DS3 includes 100 documents that were allocated to 20 events. DS4 consists of 200 random documents that were assigned to 10 events. DS5 contains 200 documents that belong to 20 events. DS6 includes 300 documents that were allocated to 20 events. Additionally, D7 and D8 were shaped by randomly choosing 800 and 2000 documents from the news aggregator dataset, respectively, which were assigned to 4 events. DS9 consists of 1467 news articles, while DS10 contains 2095 RSS news feed documents, which were scattered over 56 events. Finally, D11 includes 1139 Facebook news posts that were allocated to 33 events, whereas D12 contains 1074 Facebook news posts that were assigned to 16 events.

B. TEXT PRE-PROCESSING PHASE

In this section, the textual contents of news documents were processed, and term-vector representations were constructed. This study applied five widely used preprocessing steps in the area of ED, namely, stop words removal, URL removal, tokenization, stemming, and text document representation using the weighting scheme TFIDF.

C. FEATURE SELECTION PHASE

Let n be a collection of news text documents that include a set of features $F = \{f_{1,1}, f_{i,2}, \ldots, f_{i,j}, \ldots, f_{n,m}\}$, where m is all unique features from n documents, i represents the number of news document, and j denotes the number of features. Let $SF = \{ sf_{1,1}, sf_{i,2}, \ldots, sf_{i,j}, \ldots, sf_{n,t} \}$ represents the set of informative features which were chosen by the proposed wrapper ABBAMCL FS method with a new number of features, t denotes the new number of exclusive features, and $sf_{i,j} \in \{0,1\}$, if $sf_{i,j} = 1$, means j feature is selected while 0 denotes that j feature is rejected. Indeed, the proposed wrapper ABBAMCL FS method implicitly consists of a search technique (ABBA) which works in parallel with the underline graph-based ED (MCL) for the ED phase of the ED model. Hence, the FS phase implicitly works in parallel with the next ED phase. The implementation of ABBAMCL for choosing the optimal features is depicted in Fig. 1.

D. EVENT DETECTION PHASE

In this phase, MCL works simultaneously with BBA from the FS phase. Every single bat in the swarm of the ABBAMCL method begins with randomly initialized solutions. The set of bats is denoted by rows, i.e., vectors. The jth location in the bat indicates the status of the jth feature, i.e., if $j = 1$ ➜ selected or $j = 0$ ➜ not selected. Additionally, the solution generated by every bat is used to build an undirected-weighted graph, which is later given to the MCL to divide it into distinct event clusters. Subsequently, the modularity Q metric was employed to measure the granularity of the clustering process, whereby a large value of Q represents the high quality of the cluster's structure. At each iteration, the fitness Q value is calculated for every single solution produced by the ABBAMCL FS method to decide if an enhancement is recognized to admit and save it or discard it. After that, all generated solutions of ABBAMCL are ranked, and the solution with the highest fitness Q value is chosen as the optimum feature subset at that iteration. This process is repeated for all iterations until the stopping criteria for ABBA is met,

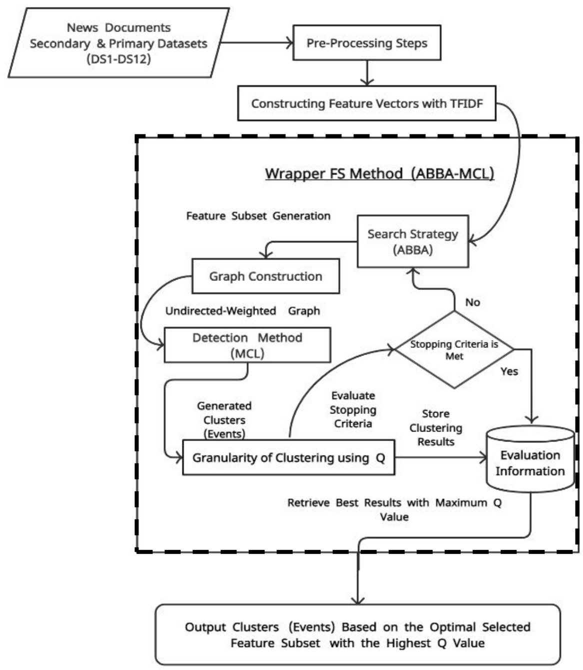


FIGURE 1. **The proposed wrapper ABBAMCL FS method.**

Algorithm 3: Proposed Wrapper ABBAMCL FS Method

Input: Original feature sets, initialize parameters of BBA (population size (n), number of features (dim,) max number of iterations for BBA (T_{bba}), loudness (A), pulse emission rate (r), alpha (α), gamma (γ), Beta (β); maximum frequency (f_{max} minimum frequency (f_{min}), initial value of impact (S_{init}), maximum inertia value (W_{max}) nonlinear modulation index (nmi), constant number (m); initialize parameters of MCL (pruning_threshold (p), expansion (exp), inflation (inf))

Output: Optimal feature subset that gives clusters (events) with the highest Modularity (Q) value

```
1   Initialize each bat in the population Xi (i = 1, 2, . . . n)
2   Find the current best solution by evaluating all solutions Xi . . . n as follows:
3       Calculate cosine similarity for the feature vectors in dim to create the adjacency matrix A
4       Construct the graph based on Adjacency matrix A
5       Run MCL (see Algorithm 2)
6       Calculate the fitness of each xi using Q metric [equation 21]
7   While (t < Tbba)
8       For each bat i = 1to n do:
9           For each feature j = 1to dim do:
10              Generate new solutions by adjusting fi [equation 2]
11              Update vi [equation (10-14)]
12              Update location [equation &&9]
13              (rand > r)
14                  Select a solution among the best solutions
15                  Generate a local solution around the selected best solution [equation 5]
16              End If
17          End For
18          Generate new solutions by flying randomly
19          Evaluate new solutions by repeating steps (2-6)
20          If (rand < A) OR (f(xi) < f(x*))[equation15]
21              Accept the new solutions
22              Increase r [equation 16] and reduce A [equation 17]
23          End If
24          Rank the bats and find the current best solution (x∗)
25      End For
26  End While
27  Return the optimal subset of features along with the set of clusters (events).
```

i.e., the iteration number reaches 100. Lastly, the output event clusters based on the optimum feature subset associated with the highest Q value are selected as the final output event clusters. Algorithm 3 shows the pseudocode for the proposed wrapper ABBAMCL FS method with the above-mentioned changes. The enhancement parts are shown in bold.

E. EVALUATION PHASE

Various evaluation metrics were used in this study for this phase. To begin with, Q was employed as a fitness function for the proposed wrapper ABBAMCL FS method to evaluate its performance in terms of selecting the optimal feature subset. Newman [193] proposed a Q metric to evaluate the quality of the identified event clusters. The main idea of Q is to compute the density of intra-cluster edges for a given random graph that has no clear structure for clusters and compare it with the density of inter-cluster edges. In fact, according to the type of graph under research, various formulas of Q have been introduced in the literature, e.g., directed-weighted graph or undirected-weighted graph. In this study, an undirected weighted graph was constructed and given to the MCL. Therefore, the formula that is given by (21) was used:

$$Q = \frac{1}{2\mathrm{m}} \sum\nolimits_{i,j} \left[w_{ij} - \frac{k_i k_j}{2\mathrm{m}} \right] \delta \left(C_i, C_j \right) \tag{21}$$

where m represents the number of edges, k_i denotes the degree of node i, while k_j indicates the degree of node j, C_i represents the community that node i belongs to, C_j denotes the community that node j is assigned to, and $(C_i, C_j) = 1$, if i and j are allocated to the same community, otherwise it equals *0*. In addition, three evaluation measurements, namely, F-measure (F), Precision (P), and Recall (R), were also utilized to evaluate the performance of the proposed wrapper ABBAMCL FS method. In the literature, F, P, and R have been widely used to evaluate various FS methods in the text clustering area [194], [195]. The same metrics have been employed to assess the performance of different detection methods in the ED field [9], [11]. F is based on two measures: P and R, which are computed using (22) and (23), respectively.

$$P(i, j) = \frac{n_{i,j}}{n_j}, \tag{22}$$

$$R(i, j) = \frac{n_{i,j}}{n_i}, \tag{23}$$

where, $n_{i,j}$ represents the number of documents for class i in cluster j, n_j indicates the number of documents for cluster j and n_i denotes the number of documents for class i. F for F for the class i in cluster j is computed using (24):

$$F(i, j) = \frac{2 \times P(i, j) \times R(i, j)}{P(i, j) \times R(i, j)} \tag{24}$$

$P(i, j)$ indicates the precision of documents for class i in cluster *j*, $R(i, j)$ represents the recall of documents for class i in cluster j, and the F for all clusters is calculated according to (25):

$$F = \frac{\sum_{j=1}^{K} F(i, j)}{K}, \tag{25}$$

where K represents the total number of generated clusters.

Total average for F, P, and R are calculated using (26):

$$M_{AVG} = \frac{1}{w} \sum_{i=1}^{w} M_i^* \tag{26}$$

where M in M_i^* denotes either F, P, or R measure in ith run. Besides all the above evaluation metrics, the Selected Feature Ratio (SFR) metric is also utilized and calculated using (27):

$$\mathrm{SFR} = \frac{1}{w}\sum_{i=1}^{w}\frac{\mathrm{length}(x)_i^*}{|D|} \tag{27}$$

where w represents the total number of runs, $|D|$ indicates the total number of features, and $(x)_i^*$ denotes the length of the chosen feature subset at ith run. To illustrate, F_{AVG}, P_{AVG}, R_{AVG}, and SFR results for 10 runs over 12 news datasets were recorded to evaluate the performance of the proposed wrapper ABBAMCL FS method. In addition, ABBAMCL's performance was compared against the standard MCL, BBA, BGA, BPSO, BBDFA, BGSA, BCS, BBF, BHH, BS, BGWO, and DIWBBA.

V. PARAMETER SETTINGS

Table 3 shows the parameter settings for all the methods that were applied in the experiment for this study. In the case of MCL, several preliminary experiments were conducted to determine the values of the inflation and pruning parameters of MCL (see Table 3). For the other MCL's parameters, the default values used by the original study were employed [182]. On the other hand, the parameters for the comparative MHAs were adopted from their original studies: GA [29], BPSO [29], BDFA [66], BGSA [31], BCS [33], BBF [75], BHH [77], BS [196], BGWO [197], DIWBBA [41], and BBAMCL [43]. These studies have verified the effectiveness of such values in obtaining good results. A population size of 20 was employed for all comparative algorithms, and every single algorithm was terminated after 100 iterations. The results from the proposed method ABBAMCL were recorded for 10 independent runs and compared against the results obtained by GA, BPSO, BDFA, BGSA, BCS, BBF, BHH, BS, BGWO, DIWBBA, and BBAMCL. To emphasize, the values used for the BBA's parameters were the same as the settings used in our previous work [43].

VI. EXPERIMENTAL RESULTS

This section presents the experimental results for the proposed wrapper ABBAMCL FS method performance over the 12 datasets from three different perspectives as follows:

A. EVALUATION METRICS

The performance of the proposed ABBAMCL FS method and other comparative methods based on the total average value of the evaluation metrics F, P, R, and SFR (i.e., F_{AVG}, P_{AVG}, R_{AVG}, SFR), and the computational time for 10 independent runs are given in Tables 4, 5, 6, 7 and 8 respectively. The best results are shown in bold text. According to Table 4, the ABBAMCL FS method has higher F scores in comparison with other benchmark FS methods for the majority of datasets (DS1-DS7, DS9, and DS11). BBFMCL achieved the best F values in two datasets (DS10 and DS12), while BGWOMCL obtained the highest F score in DS8.

TABLE 3. Initial parameters setting for the comparative methods.

Algorithm	Parameter	Value
ABBA	Fmin	0
	Fmax	2
BBA	Loudness A	0.5
	Emission Rate r	0.5
	Alpha	0.9
	Gamma	0.9
	Fmin	0
	Fmax	2
GA	Selection ratio	Roulette wheel
	Crossover ratio	0.9
	Mutation	0.005
BPSO	C_1, C_2	2,2
	Inertia Weight (w)	0.9 to 0.4
	Max velocity	6
BCS	alpha	0.1
	beta	1.5
	Probability of discarding host nest (pa)	0.25
BGSA	Constant number (G_0)	100
BDFA	Separation (s)	[0.2,0]
	Alignment (a)	[0.2,0]
	Cohesion (c)	[0.2,0]
	Enemy factor (e)	[0,0.1]
	Food factor	[0.2,0]
	Inertia weight	[0.9,0.4]
BGWO	Alpha (a)	0.99
	Beta (β)	0.01
BBF	a	0.1
	c	min = 0.01 & max = 0.25
BHH	a	0.99
	β	0.01
BS	c_1	c_1 = 1-0 (nonlinear decreasing parameter)
	c_2	$c_2 \in [0,1]$
	c_3	$c_3 \in [0,1]$
DIWBBA	Loudness (A)	0.25
	Emission Rate (r)	0.5
	Alpha	0.9
	Gamma	0.9
	Min frequency $F_{\min}$	0
	Max frequency $F_{\max}$	2
	Max inertia (w_{max})	0.9
	Modulation index (n)	2
	m	50
	q	50
	S_{init}	0.6
MCL	Expansion	2
	Inflation (vary depending on preliminary experiments)	1.5 (DS3, DS12), 2 (DS5, DS7, DS9), 3 (DS8, DS10), 5 (DS2), 6 (DS6, DS11), 7 (DS1), 9 (DS4)
	Loop value	1
	Pruning_threshold (vary depending on preliminary experiments)	0.1 (DS1-DS6, DS12), 0.01(DS8-DS11), 0.001(DS7)
	Pruning_frequency	1
	Convergence_check_frequency	1

Table 5 outlines P scores for the proposed ABBAMCL method and all comparative FS methods. It shows that

TABLE 4. performance of FS methods based on F_{AVG}.

Datasets	BGAMCL	BPSOMCL	BDFAMCL	BGSAMCL	BCSMCL	BBFMCL	BHHMCL	BSMCL	BGWOMCL	DIWBBAMCL	BBAMCL	ABBAMCL
DS1	0.356	0.332	0.348	0.378	0.365	0.378	0.373	0.379	0.371	0.337	0.360	**0.394**
DS2	0.353	0.338	0.340	0.347	0.339	0.342	0.328	0.342	0.333	0.346	0.357	**0.374**
DS3	0.309	0.301	0.313	0.315	0.317	0.246	0.255	0.244	0.268	0.309	0.322	**0.354**
DS4	0.348	0.347	0.305	0.329	0.338	0.309	0.310	0.305	0.309	0.326	0.349	**0.360**
DS5	0.281	0.303	0.298	0.299	0.283	0.238	0.256	0.241	0.238	0.281	0.311	**0.316**
DS6	0.258	0.280	0.277	0.274	0.267	0.180	0.198	0.170	0.186	0.258	0.283	**0.297**
DS7	0.552	0.550	0.550	0.553	0.552	0.554	0.554	0.555	0.555	0.551	0.555	**0.558**
DS8	0.310	0.302	0.303	0.317	0.306	0.362	0.358	0.363	**0.366**	0.312	0.310	0.320
DS9	0.617	0.668	0.660	0.637	0.609	0.418	0.438	0.435	0.459	0.622	0.672	**0.675**
DS10	0.618	0.597	0.587	0.622	0.618	**0.701**	0.696	0.685	0.697	0.617	0.573	0.627
DS11	0.840	0.829	0.833	0.846	0.842	0.788	0.805	0.791	0.809	0.841	0.843	**0.847**
DS12	0.596	0.564	0.568	0.582	0.581	**0.629**	0.624	0.606	0.608	0.576	0.577	0.598

TABLE 5. Performance of FS methods based on P_{AVG}.

Datasets	BGAMCL	BPSOMCL	BDFAMCL	BGSAMCL	BCSMCL	BBFMCL	BHHMCL	BSMCL	BGWOMCL	DIWBBAMCL	BBAMCL	ABBAMCL
DS1	0.488	0.494	0.513	0.533	0.513	0.370	0.384	0.352	0.357	0.472	0.519	**0.557**
DS2	0.363	0.377	0.362	0.373	0.370	0.257	0.238	0.252	0.249	0.360	0.387	**0.398**
DS3	0.244	0.249	0.257	0.259	0.260	0.151	0.167	0.148	0.182	0.244	0.265	**0.299**
DS4	0.377	0.390	0.351	0.345	0.362	0.231	0.240	0.211	0.221	0.336	0.393	**0.398**
DS5	0.221	0.255	0.254	0.244	0.231	0.151	0.164	0.149	0.147	0.221	**0.274**	0.269
DS6	0.196	0.233	0.236	0.213	0.203	0.104	0.126	0.100	0.111	0.194	0.242	**0.256**
DS7	0.989	0.986	0.987	0.989	0.988	0.990	0.989	0.990	0.991	0.987	0.992	**0.993**
DS8	0.976	0.974	0.975	0.975	0.974	**0.977**	0.975	**0.977**	0.976	0.974	0.975	0.974
DS9	0.467	0.535	0.525	0.487	0.457	0.347	0.285	0.285	0.304	0.468	**0.548**	0.540
DS10	0.662	0.666	0.670	0.668	0.660	0.659	0.653	0.657	0.653	0.665	0.671	**0.672**
DS11	0.835	0.828	0.836	**0.855**	0.836	0.714	0.751	0.728	0.746	0.844	0.853	0.853
DS12	0.720	0.728	0.719	0.724	0.709	0.560	0.575	0.556	0.546	0.721	**0.739**	0.723

TABLE 6. Performance of FS methods based on R_{AVG}.

Datasets	BGAMCL	BPSOMCL	BDFAMCL	BGSAMCL	BCSMCL	BBFMCL	BHHMCL	BSMCL	BGWOMCL	DIWBBAMCL	BBAMCL	ABBAMCL
DS1	0.282	0.251	0.264	0.295	0.285	0.411	0.399	**0.435**	0.405	0.265	0.278	0.305
DS2	0.349	0.309	0.322	0.326	0.314	0.572	0.551	0.574	**0.592**	0.337	0.333	0.353
DS3	0.431	0.382	0.400	0.408	0.411	0.742	0.622	**0.746**	0.592	0.431	0.414	0.399
DS4	0.325	0.314	0.271	0.317	0.318	0.563	0.500	**0.621**	0.549	0.322	0.315	0.330
DS5	0.389	0.376	0.365	0.391	0.372	0.638	0.668	**0.719**	0.707	0.389	0.366	0.387
DS6	0.383	0.355	0.337	0.389	0.391	**0.690**	0.628	0.683	0.688	0.392	0.345	0.351
DS7	0.383	0.381	0.381	0.384	0.383	0.384	0.385	0.386	0.385	0.382	0.386	**0.388**
DS8	0.184	0.179	0.180	0.189	0.190	0.223	0.219	0.223	**0.225**	0.193	0.184	0.192
DS9	0.911	0.889	0.891	0.925	0.913	0.917	**0.969**	0.949	0.946	0.930	0.868	0.904
DS10	0.580	0.555	0.522	0.582	0.582	**0.753**	0.747	0.719	0.751	0.576	0.501	**0.588**
DS11	0.844	0.830	0.830	0.837	0.848	0.883	0.870	**0.873**	0.886	0.838	0.834	0.842
DS12	0.510	0.461	0.469	0.487	0.492	**0.729**	0.686	0.685	0.705	0.506	0.473	0.511

TABLE 7. Performance of FS methods based on *SFR*.

Datasets	BGAMCL	BPSOMCL	BDFAMCL	BGSAMCL	BCSMCL	BBFMCL	BHHMCL	BSMCL	BGWOMCL	DIWBBAMCL	BBAMCL	ABBAMCL
DS1	0.59	0.49	0.59	0.70	0.76	0.52	0.53	0.54	0.62	0.61	0.49	**0.47**
DS2	0.63	0.62	0.54	0.60	0.71	0.53	0.59	0.49	0.62	0.59	0.54	**0.39**
DS3	0.65	0.56	0.55	0.60	0.67	0.52	0.56	0.48	0.58	0.65	0.55	**0.47**
DS4	0.81	0.64	0.61	0.75	0.75	0.54	0.61	0.55	0.66	0.65	0.69	**0.53**
DS5	0.72	0.59	0.65	0.74	0.77	0.52	0.63	**0.50**	0.64	0.72	0.61	0.53
DS6	0.68	0.63	0.62	0.65	0.74	0.56	0.61	0.51	0.63	0.67	0.63	**0.50**
DS7	0.81	0.68	0.65	0.72	0.79	0.63	0.65	0.60	0.69	0.77	0.54	**0.40**
DS8	0.78	0.70	0.69	0.70	0.79	0.65	0.68	0.62	0.71	0.76	**0.61**	**0.61**
DS9	0.76	0.75	0.76	0.77	0.83	0.70	0.74	0.67	0.75	0.71	0.44	**0.43**
DS10	0.82	0.73	0.72	0.78	0.82	0.67	0.70	**0.63**	0.72	0.75	0.64	**0.63**
DS11	0.81	0.72	0.69	0.94	0.80	0.62	0.68	**0.58**	0.67	0.87	0.82	0.69
DS12	0.84	0.70	0.67	0.77	0.79	0.61	0.68	**0.58**	0.69	0.67	0.67	0.62

ABBAMCL achieved the best *P* scores in 7 datasets (DS1-DS4, DS6, DS7, and DS10). In contrast, BBAMCL achieved the best second *P* values in 3 datasets (DS5, DS9, and DS12) in comparison with the other comparative FS methods. The BGSAMCL, BBFMCL, and BS-MCL came in third place with the best *P* values for a single dataset.

Based on Table 6, it is found that BSMCL obtained better *R* scores in comparison with other FS methods in 5 datasets (DS1, DS3-DS5, and DS11). BBFMCL ranked second with the best *R* results in 3 datasets (DS6, DS10, and DS12). In the same context, the proposed ABBAMCL and BGWOMCL achieved the best *R* scores in two datasets for each, whereas BHHMCL obtained the best *R* in one dataset, as depicted in Table 6.

Table 7 shows the outstanding performance of the ABBAMCL FS method in terms of selecting the lowest number of features in almost all datasets (DS1-DS4, DS6-DS10) for which it has recorded the best *F* (see Table 4).

TABLE 8. Average computational time (in seconds) for different FS methods.

Datasets	BGAMCL	BPSOMCL	BDFAMCL	BGSAMCL	BCSMCL	BBFMCL	BHHMCL	BSMCL	BGWOMCL	DIWBBAMCL	BBAMCL	ABBAMCL
DS1	122.99	315.62	336.30	398.60	352.83	264.08	282.75	227.63	134.91	111.53	**9.30**	87.13
DS2	51.92	111.30	164.53	168.98	155.26	285.32	323.84	196.09	180.19	116.83	**4.00**	112.09
DS3	25.85	150.67	180.93	145.68	151.71	358.70	410.28	293.82	237.16	152.85	**7.10**	142.45
DS4	159.59	449.96	567.84	615.84	553.44	1115.72	1408.91	811.63	757.90	555.32	**26.50**	403.80
DS5	134.05	520.20	595.91	650.66	642.17	1206.78	1383.28	897.40	852.30	434.05	**56.60**	389.53
DS6	356.58	1086.11	1216.81	1289.69	1453.33	5433.75	2710.59	2006.93	1685.02	1248.49	**160.00**	1014.06
DS7	**1920.37**	5108.23	5650.70	5568.82	5549.21	29871.79	28545.79	4045.28	4037.56	2566.99	2025.60	2516.10
DS8	**10710.10**	27830.11	24433.63	23810.00	26384.53	126895.549	122969.07	15870.18	17044.19	22491.34	17925.10	19265.34
DS9	**7779.60**	16443.46	19136.13	23210.61	22435.23	19744.34	28503.44	16772.83	12723.95	19367.60	10426.96	19110.21
DS10	19179.30	50421.10	50799.09	46544.95	42919.04	36936.14	48161.24	26134.45	33456.02	22546.50	**14026.02**	19148.36
DS11	**2558.40**	11421.85	16745.90	11094.35	9859.25	12430.37	9800.81	5599.78	6602.31	6027.86	5402.60	6022.79
DS12	**1156.60**	4349.02	5545.95	3612.57	6075.29	14022.32	13280.11	9164.83	8427.82	2730.49	1766.70	1885.16

BSMCL placed second with the minimum number of features for 4 datasets (DS5, and DS10-DS12). For DS8 dataset, BBAMCL ranked third, sharing the lowest number of features with ABBAMCL.

Table 8 shows the obtained computational time for the various FS methods. The computational time of the BBAMCL is the shortest time for 7 datasets (DS1-DS6 and DS10). However, it failed to achieve the best F values (see Table 4). GAMCL ranked second with short computational time for 5 datasets (DS7-DS9, DS11, and DS12). On the other hand, the proposed ABBAMCL ranked third with a longer computational time for all datasets compared to the BBAMCL and GAMCL methods. Consequently, the most significant finding is, the computational time reduction is observed in ABBAMCL in comparison to other FS methods like BGSAMCL, BCSMCL, BBFMCL, BHHMCL, BSMCL, BGWOMCL, DIWBBAMCL, BDFAMCL, and BPSOMCL.

B. CONVERGENCE RATE

Fig. 2 illustrates the convergence curves of all compared FS methods for 12 datasets (DS1-DS12). Note that the fitness function is the Q value obtained from 100 runs based on the best F scores for every single FS method. In Fig. 2, the proposed ABBCMCL is marked with a plus sign and demonstrates that the performance of ABBAMCL was superior for almost all datasets. For datasets DS1, DS6 – DS9, DS11, and DS12, it can be clearly seen that ABBAMCL outperformed other comparative FS methods, which converged quicker and deeper to find the optimum solution. Such enhancement is mostly due to the controlling strategy for r and A parameters, which greatly improves the performance of ABBAMCL in FS. Another interesting point to highlight is that ABBAMCL achieved a better fitness value than BBAMCL, which showed competitive performance in datasets i.e., DS2-DS6, and DS9-DS12. This indicates that the proposed ABBAMCL FS method surpassed BBAMCL by overcoming the drawbacks of BBA in both premature convergence and the controlling of r and A parameters.

C. STATISTICAL RESULTS

The Friedman statistical test was performed based on the F score. Table 9 outlines the average rankings of the applied FS methods.

TABLE 9. Results of friedman rank test based on F_{avg}.

FS Method	Mean Rank	Ranking
ABBAMCL	**10.92**	**1**
BBAMCL	8.29	2
BGSAMCL	7.96	3
BHHMCL	6,63	4
GAMCL	6.25	5
BCSMCL	6.08	6
BGWOMCL	5.92	7
BBFMCL	5.63	8
BSMCL	5.50	9
DIWBBAMCL	5.29	10
BPSOMCL	4.88	11
BDFAMCL	4.67	12
p-value	0.00	

TABLE 10. Results of wilcoxon signed-rank test based on F_{AVG}.

Algorithms	Total Datasets	+	-	=	p-value
ABBAMCL – GAMCL	12	12	0	0	**.002**
ABBAMCL – BPSOMCL	12	12	0	0	**.002**
ABBAMCL – BDFAMCL	12	12	0	0	**.002**
ABBAMCL – BGSAMCL	12	12	0	0	**.003**
ABBAMCL – BCSMCL	12	12	0	0	**.002**
ABBAMCL – BBAMCL	12	12	0	0	**.002**
ABBAMCL- DIWBBAMCL	12	12	0	0	**.002**
ABBAMCL-BBFMCL	12	9	3	0	**.071**
ABBAMCL-BHHMCL	12	8	4	0	.136
ABBAMCL-BSMCL	12	9	3	0	**.060**
ABBAMCL-BGWOMCL	12	9	3	0	**.071**

The p-value was computed at 0.00, which is less than the (0.05) that was assumed to be a significant level for all datasets. The proposed wrapper ABBAMCL FS method was ranked the highest, followed by BBAMCL, BGSAMCL, BHHMCL, GAMCL, BCSMCL, BGWOMCL, BBFMCL, BSMCL, DIWBBAMCL, BDFAMCL, and last place BPSOMCL. This proves that the performance of the proposed ABBAMCL FS method is significantly better than other comparative FS methods. p-values for Wilcoxon's signed-rank test are shown in Table 10. They were calculated at (0.05) significance level and used ABBAMCL as the reference method. In such a statistical test, a null hypothesis is rejected if p-values are smaller than 0.05, which means there is a significant difference in clustering (detection) performance between two different FS methods. The results from

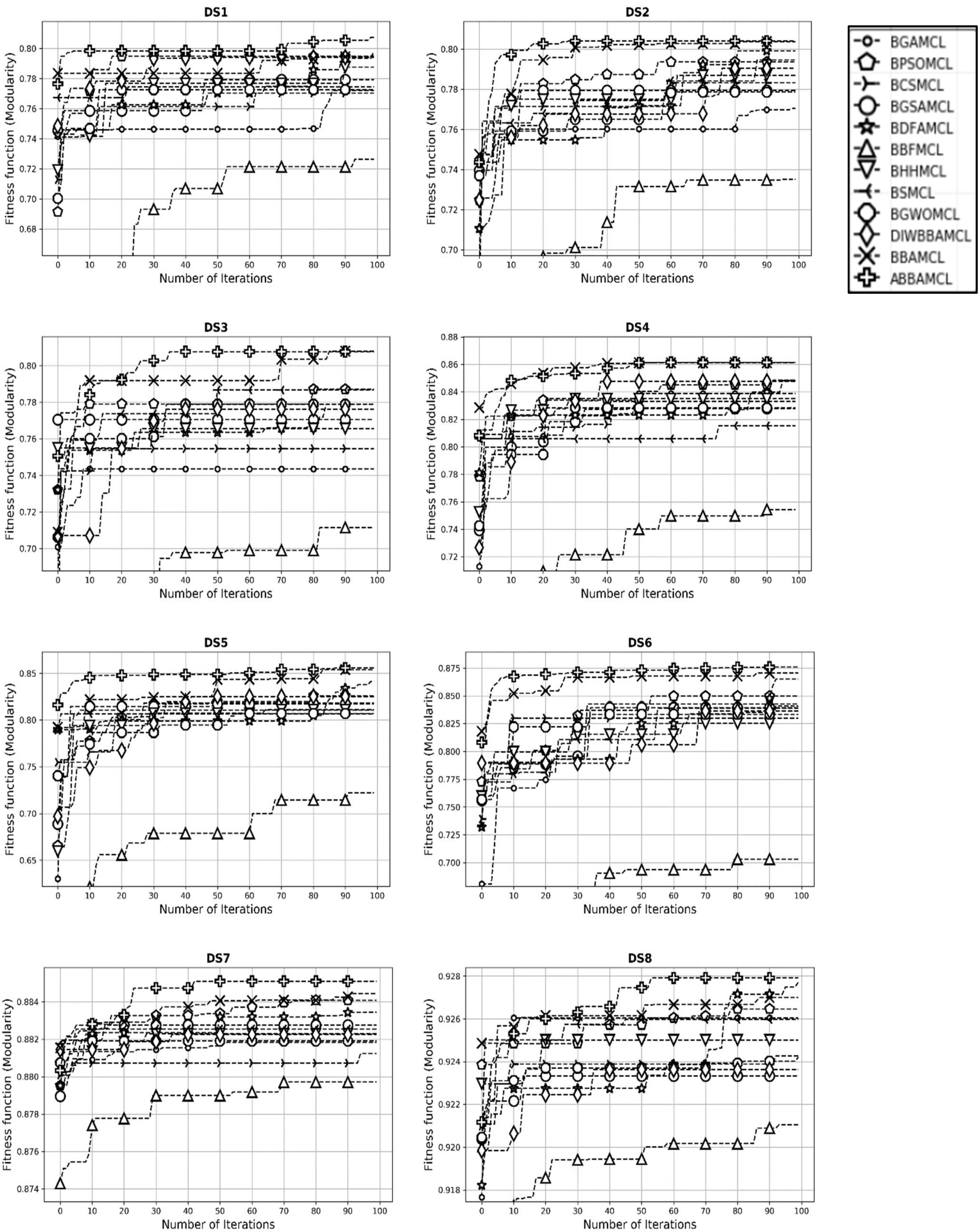


FIGURE 2. Convergence graph of all FS methods for DS1-DS12 datasets.

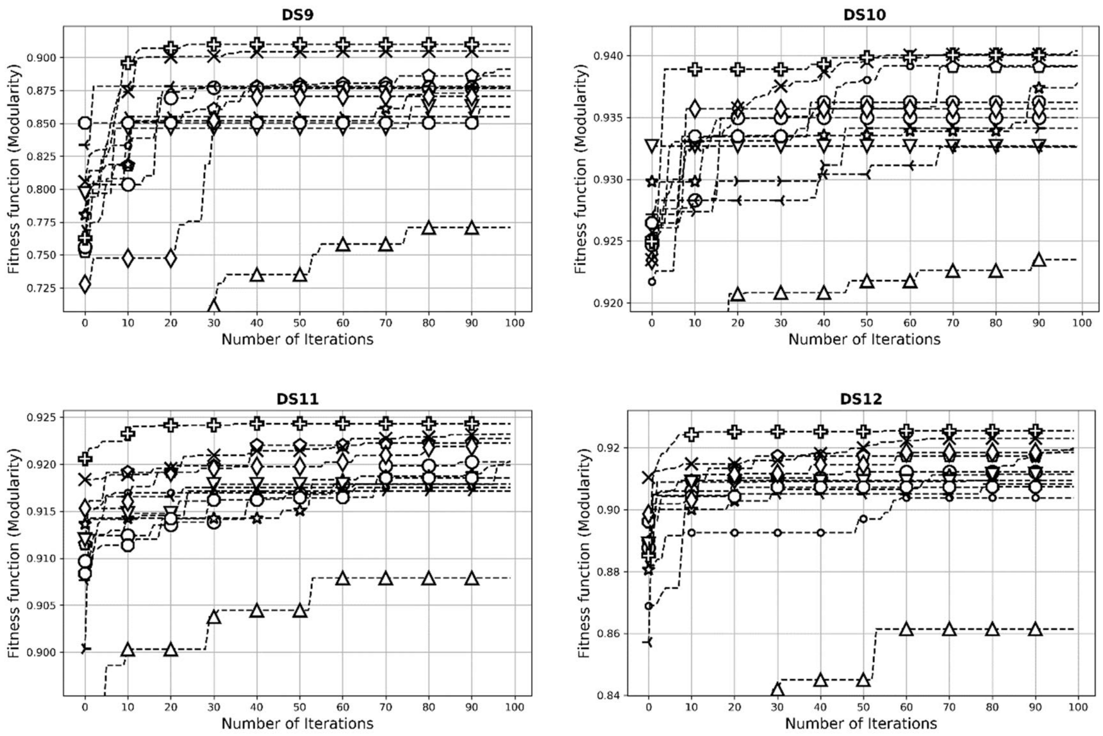


FIGURE 2. ***(Continued.)*** **Convergence graph of all FS methods for DS1-DS12 datasets.**

Table 10 confirm that the proposed ABBAMCL FS method is significantly better than the comparative FS methods for all datasets.

VII. DISCUSSION

The experimental results based on the F score have shown superior performance of the proposed wrapper ABBAMCL FS method in comparison with other FS methods over almost all datasets (see Table 4). This is due to the improvements that have been made to various parts of the original BBA structure. Such developments support ABBAMCL to overcome the problem of a fast convergence rate in the early stages. To illustrate, such modifications provide ABBAMCL with the ability to discover the whole feature space more effectively and determine the best solution without falling into the local optimum solution. Consequently, improve the clustering performance of the MCL to produce high-quality event clusters with high F scores. Such results have confirmed what has been stated in the literature that through good control of the r and A parameters, BBA's performance can be improved more and much better results can be achieved [53], [55], [88], [127]. Moreover, the results have verified the effectiveness of the proposed adaptive techniques in controlling r and A parameters during the implementation of the ABBAMCL. These techniques generate values that perfectly create a good balance between the exploration and exploitation processes of ABBA.

To be specific, ABBA_MCL had the highest *P* scores for DS1-DS4, DS6, and DS7 datasets (see Table 5). These datasets are the same ones where BBA has obtained the highest *F* scores as well (see Table 4). This happens due to the improved exploration ability of ABBA, which guides it to the regions in feature space where the most informative feature subsets about events are found. As a result, ABBAMCL succeeded in placing news documents in the correct event clusters in most datasets and increasing the *F* and *P* scores. In contrast, ABBAMCL achieved the best *R* scores in only two datasets (DS7 and DS10) (see Table 6). Despite the low performance of ABBAMCL in some datasets in terms of *P* and *R* measures, it achieved the best *F* scores for the majority of datasets, which is reported in the literature to be the most important evaluation metric in the ED and FS domains [9], [11], [60]. The results from Table 7 confirmed the capability of the proposed wrapper ABBAMCL FS method to select the minimum number of features that can improve the performance of the MCL in detecting events.

Table 8 shows that ABBAMCL ranked third in achieving the least time compared to other FS methods. The reason behind the longer computational time is the proposed enhancements paired with the BBA, which give ABBAMCL more power exploration ability to better discover the feature space. As a result, the convergence rate of the ABBA has increased, resulting in higher computational time. Implicitly, the ABBA algorithm has more improvement steps to be implemented compared to the other applied methods, which affects the overall computational time of the ABBAMCL. Nevertheless, ABBAMCL achieved the best clustering performance according to *F* score results (refer to Table 4).

Based on several ED studies for historical heterogeneous news text documents, the F score is a more important metric than the computational time in the ED field.

The Friedman rank test was used to show the significant statistical differences between the applied methods. Table 9 proves the superiority of the ABBAMCL in terms of discovering highly accurate real-world events from multiple heterogeneous news text documents, whereby a lower p-value of 0.000 was attained. This evidence reveals the significant detection performance of the ABBAMCL in comparison to other baseline FS methods. The findings of the Wilcoxon signed-rank test illustrated the effectiveness of the BBA-MCL through the number of victories it achieved over other baseline FS methods on different datasets (see Table 10).

One important point to highlight is the slightly lower P and R scores that are observed for several datasets in comparison to other FS methods (see Table 5 and Table 6). The reason behind such low performance might be the poor performance of MCL in ABBAMCL. This is because ABBAMCL's performance is affected by the performances of both ABBA and MCL, and their interaction with each other. As a matter of fact, MCL also suffers from a fast convergence rate, which essentially depends on the *inf* and pruning p parameters of MCL [51], [198], [199]. Therefore, introducing techniques to control the values of such parameters can enhance the performance of the MCL and achieve more accurate event clusters with higher P and R scores. To address this problem, our future work will focus on enhancing the convergence behaviors of MCL to increase the effectiveness and performance of the wrapper ABBAMCL FS method even more.

VIII. CONCLUSION

In the last decade, the ED from electronic multiple heterogeneous news documents has been extensively studied. However, news documents of various lengths generate spaces of different dimensions, which in turn affect the overall performance of the ED model. Previous ED studies have either ignored the FS phase or employed traditional FS methods, which failed in identifying the optimal informative features subset. To solve such a problem, many scholars from the text mining field have proposed different wrapper FS methods based on MHAs, including BBA. In our earlier work, the wrapper FS method based on BBA and MCL has achieved better results in the context of ED than any other method. However, BBAMCL has shown some poor performance for several datasets due to the fast convergence rate problem of BBA. To overcome this problem, this study proposes simple adaptive techniques to control the values of A and r parameters and solve the fast convergence rate drawback of BBA in BBAMCL. Totally, 12 news text datasets were utilized to assess the performance of ABBAMCL against eleven wrapper FS methods, namely, BBAMCL, GAMCL, BPSOMCL, BCSMCL, BDFAMCL, BGSAMCL, BBFMCL, BHHMCL, BSMCL, BGWOMCL, and DIWBBAMCL. The experiment was measured using F_{AVG}, P_{AVG}, R_{AVG}, SFR and computational time metrics. The evaluation results have shown the superior performance of ABBAMCL in terms of choosing a minimal optimal feature subset and obtaining the highest F, P, and R scores. Additionally, ABBAMCL has revealed a better convergence rate in comparison with other wrapper FS methods. The statistical test results verify that ABBAMCL has outperformed other FS methods significantly at 0.00. Despite the outstanding performance of the ABBAMCL, several drawbacks are identified, such as the fast convergence rate of MCL, which affects the overall performance of the ABBAMCL and leads to poor results in some datasets according to P and R scores. For future work, more recent MHAs with MCL can be considered as comparative methods. Additionally, MCL can be further improved by enhancing the convergence rate of MCL through tuning or controlling the inflation, expansion, and pruning parameters of MCL to operate effectively on different data sizes. Finally, other comparative graph ED methods can be included and tested.

ACKNOWLEDGMENT

The authors of this paper would like to express thanks to the editor and all reviewers for the valuable comments.

REFERENCES

[1] A. Goswami and A. Kumar, "A survey of event detection techniques in online social networks," *Social Netw. Anal. Mining*, vol. 6, no. 1, p. 107, Dec. 2016, doi: 10.1007/S13278-016-0414-1.

[2] W. Z. AL-Dyani, F. K. Ahmad, and S. S. Kamaruddin, "A survey on event detection models for text data streams," *J. Comput. Sci.*, vol. 16, no. 7, pp. 916–935, Jul. 2020, doi: 10.3844/JCSSP.2020.916.935.

[3] Z. Fu, X. Sun, J. Shu, and L. Zhou, "Plain text zero knowledge watermarking detection based on asymmetric encryption," *Adv. Sci. Technol.*, vol. 48, pp. 126–134, Jan. 2014, doi: 10.14257/ASTL.2014.48.21.

[4] N. Panagiotou, I. Katakis, and D. Gunopulos, "Detecting events in online social networks: Definitions, trends and challenges," in *Solving Large Scale Learning Tasks. Challenges and Algorithms* (Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9580. Cham, Switzerland: Springer, 2016, pp. 42–84.

[5] I. Mele, S. A. Bahrainian, and F. Crestani, "Event mining and timeliness analysis from heterogeneous news streams," *Inf. Process. Manage.*, vol. 56, no. 3, pp. 969–993, May 2019, doi: 10.1016/j.ipm.2019.02.003.

[6] R. Gashi and H. G. Ahmeti, "Impact of social media on the development of new products, marketing and customer relationship management in Kosovo," *Emerg. Sci. J.*, vol. 5, no. 2, pp. 125–138, Apr. 2021, doi: 10.28991/ESJ-2021-01263.

[7] H. Wada, "Assessing the social media user's credibility rating of shared content, and its utilization in decision making," *Emerg. Sci. J.*, vol. 5, no. 2, pp. 191–199, Apr. 2021, doi: 10.28991/ESJ-2021-01269.

[8] G. Leban, B. Fortuna, J. Brank, and M. Grobelnik, "Event registry: Learning about world events from news," in *Proc. 23rd Int. Conf. World Wide Web*, 2014, pp. 107–110, doi: 10.1145/2567948.2577024.

[9] Y. Wei, L. Singh, D. Buttler, and B. Gallagher, "Using semantic graphs to detect overlapping target events and story lines from newspaper articles," *Int. J. Data Sci. Anal.*, vol. 5, no. 1, pp. 41–60, Feb. 2018, doi: 10.1007/S41060-017-0066-X.

[10] I. Moutidis and H. T. P. Williams, "Utilizing complex networks for event detection in heterogeneous high-volume news streams," in *Proc. Int. Conf. Complex Netw. Appl.*, 2019, pp. 659–672, doi: 10.1007/978-3-030-36687-2_55.

[11] E. Rasouli, S. Zarifzadeh, and A. J. Rafsanjani, "WebKey: A graph-based method for event detection in web news," *J. Intell. Inf. Syst.*, vol. 54, no. 3, pp. 585–604, Jun. 2020, doi: 10.1007/S10844-019-00576-7.

[12] C. Mhamdi, M. Al-Emran, and S. A. Salloum, "Text mining and analytics: A case study from news channels posts on Facebook," in *Intelligent Natural Language Processing: Trends and Applications*. Cham, Switzerland: Springer, 2018, pp. 399–415.

[13] R. Prasad, D. Bisandu, and M. Liman, "Clustering news articles using efficient similarity measure and N-grams," *Int. J. Knowl. Eng. Data Mining*, vol. 5, no. 1, p. 1, 2018, doi: 10.1504/IJKEDM.2018.10016103.

[14] S. Yu and B. Wu, "Exploiting structured news information to improve event detection via dual-level clustering," in *Proc. IEEE 3rd Int. Conf. Data Sci. Cyberspace (DSC)*, Jun. 2018, pp. 873–880, doi: 10.1109/DSC.2018.00140.

[15] Q. H. Ramadan and M. Mohd, "A review of retrospective news event detection," in *Proc. Int. Conf. Semantic Technol. Inf. Retr.*, Jun. 2011, pp. 209–214, doi: 10.1109/STAIR.2011.5995790.

[16] S. A. Salloum, C. Mhamdi, M. Al-Emran, and K. Shaalan, "Analysis and classification of Arabic newspapers' Facebook pages using text mining techniques," *Int. J. Inf. Technol.*, vol. 1, no. 2, pp. 8–17, 2017.

[17] S. A. Salloum, M. Al-Emran, S. Abdallah, and K. Shaalan, "Analyzing the Arab Gulf newspapers using text mining techniques," in *Proc. Int. Conf. Adv. Intell. Syst. Inform.*, 2017, pp. 396–405, doi: 10.1007/978-3-319-64861-3_37.

[18] e. al. S. A. Salloum, "Mining social media text: Extracting knowledge from Facebook," *Int. J. Comput. Digit. Syst.*, vol. 6, no. 2, pp. 73–81, Mar. 2017, doi: 10.12785/IJCDS/060203.

[19] A. Dhiman and D. Toshniwal, "An approximate model for event detection from Twitter data," *IEEE Access*, vol. 8, pp. 122168–122184, 2020, doi: 10.1109/ACCESS.2020.3007004.

[20] A. H. Hossny, L. Mitchell, N. Lothian, and G. Osborne, "Feature selection methods for event detection in Twitter: A text mining approach," *Social Netw. Anal. Mining*, vol. 10, no. 1, pp. 1–15, Dec. 2020, doi: 10.1007/S13278-020-00658-3.

[21] T. M. Beigh, S. Upadhyaya, and G. Gopal, "Event identification in social news streams using keyword analysis," *Int. Res. J. Eng. Technol.*, vol. 3, no. 5, pp. 1781–1786, 2016, doi: 10.1109/ICISS.2010.5654957.

[22] X. Dai and Y. Sun, "Event identification within news topics," in *Proc. Int. Conf. Intell. Comput. Integr. Syst.*, Oct. 2010, pp. 498–502, doi: 10.1109/ICISS.2010.5654957.

[23] L. Hu, B. Zhang, L. Hou, and J. Li, "Adaptive online event detection in news streams," *Knowl.-Based Syst.*, vol. 138, pp. 105–112, Dec. 2017, doi: 10.1016/j.knosys.2017.09.039.

[24] H. Nanba, R. Saito, A. Ishino, and T. Takezawa, "Automatic extraction of event information from newspaper articles and web pages," in *Proc. Int. Conf. Asian Digit. Libraries*, 2013, pp. 171–175, doi: 10.1007/978-3-319-03599-4_21.

[25] A. Edouard, E. Cabrio, S. Tonelli, and N. Le Thanh, "Graph-based event extraction from Twitter," in *Proc. Recent Adv. Natural Lang. Process. Meet Deep Learn.*, Nov. 2017, pp. 222–230, doi: 10.26615/978-954-452-049-6_031.

[26] B. S. Harish and M. B. Revanasiddappa, "A comprehensive survey on various feature selection methods to categorize text documents," *Int. J. Comput. Appl.*, vol. 164, no. 8, pp. 1–7, Apr. 2017, doi: 10.5120/IJCA2017913711.

[27] A. Jovic, K. Brkic, and N. Bogunovic, "A review of feature selection methods with applications," in *Proc. 38th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, 2015, pp. 1200–1205, doi: 10.1109/MIPRO.2015.7160458.

[28] K. K. Bharti and P. K. Singh, "A survey on filter techniques for feature selection in text mining," in *Proc. 2nd Int. Conf. Soft Comput. Problem Solving (SocProS)*, Dec. 2012, pp. 1545–1559, doi: 10.1007/978-81-322-1602-5_154.

[29] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Comput. Appl.*, vol. 25, nos. 3–4, pp. 663–681, Sep. 2014, doi: 10.1007/S00521-013-1525-5.

[30] H. Banka and S. Dara, "A Hamming distance based binary particle swarm optimization (HDBPSO) algorithm for high dimensional feature selection, classification and validation," *Pattern Recognit. Lett.*, vol. 52, pp. 94–100, Jan. 2015.

[31] E. Rashedi, H. Nezamabadi-Pourand, and S. Saryazdi, "BGSA: Binary gravitational search algorithm," *Natural Comput.*, vol. 9, no. 3, pp. 727–745, Sep. 2010, doi: 10.1007/S11047-009-9175-3.

[32] M. A. Tawhid and K. B. Dsouza, "Hybrid binary dragonfly enhanced particle swarm optimization algorithm for solving feature selection problems," *Math. Found. Comput.*, vol. 1, no. 2, pp. 181–200, 2018, doi: 10.3934/MFC.2018009.

[33] D. Rodrigues, L. A. M. Pereira, T. N. S. Almeida, J. P. Papa, A. N. Souza, C. C. O. Ramos, and X.-S. Yang, "BCS: A binary cuckoo search algorithm for feature selection," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2013, pp. 465–468, doi: 10.1109/ISCAS.2013.6571881.

[34] A.-C. Enache, V. Sgarciu, and A. Petrescu-Nita, "Intelligent feature selection method rooted in binary bat algorithm for intrusion detection," in *Proc. IEEE 10th Jubilee Int. Symp. Appl. Comput. Intell. Informat.*, May 2015, pp. 517–521, doi: 10.1109/SACI.2015.7208259.

[35] R. R. Rajalaxmi and A. Ramesh, "Binary bat approach for effective spam classification in online social networks," *Austral. J. Basic Appl. Sci.*, vol. 8, no. 18, pp. 383–388, 2014.

[36] J. Sharma and B. Annappa, "Community detection using meta-heuristic approach: Bat algorithm variants," in *Proc. 9th Int. Conf. Contemp. Comput. (IC3)*, Aug. 2016, pp. 1–7, doi: 10.1109/IC3.2016.7880209.

[37] A. A. Akinyelu and A. O. Adewumi, "On the performance of cuckoo search and bat algorithms based instance selection techniques for SVM speed optimization with application to e-fraud detection," *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 3, pp. 1348–1375, 2018, doi: 10.3837/TIIS.2018.03.021.

[38] A.-C. Enache and V. Sgarciu, "Anomaly intrusions detection based on support vector machines with bat algorithm," in *Proc. 18th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Oct. 2014, pp. 856–861, doi: 10.1109/ICSTCC.2014.6982526.

[39] T. Jayabarathi, T. Raghunathan, and A. H. Gandomi, "The bat algorithm, variants and some practical engineering applications: A review," in *Nature-Inspired Algorithms and Applied Optimization*. Cham, Switzerland: Springer, 2018, pp. 313–330.

[40] I. Fister, I. Fister, X.-S. Yang, S. Fong, and Y. Zhuang, "Bat algorithm: Recent advances," in *Proc. IEEE 15th Int. Symp. Comput. Intell. Informat. (CINTI)*, Nov. 2014, pp. 163–167, doi: 10.1109/CINTI.2014.7028669.

[41] X. Huang, X. Zeng, and R. Han, "Dynamic inertia weight binary bat algorithm with neighborhood search," *Comput. Intell. Neurosci.*, vol. 2017, pp. 1–15, May 2017, doi: 10.1155/2017/3235720.

[42] W. Z. Al-Dyani, A. H. Yahya, and F. K. Ahmad, "Challenges of event detection from social media streams," *Int. J. Eng. Technol.*, vol. 7, no. 2, pp. 72–75, 2018, doi: 10.14419/IJET.V7I2.15.11217.

[43] W. Z. Al-Dyani, F. K. Ahmad, and S. S. Kamaruddin, "Binary bat algorithm for text feature selection in news events detection model using Markov clustering," *Cogent Eng.*, vol. 9, no. 1, Dec. 2022, Art. no. 2010923, doi: 10.1080/23311916.2021.2010923.

[44] Q. Chen, X. Guo, and H. Bai, "Semantic-based topic detection using Markov decision processes," *Neurocomputing*, vol. 242, pp. 40–50, Jun. 2017, doi: 10.1016/j.neucom.2017.02.020.

[45] B. Manaskasemsak, B. Chinthanet, and A. Rungsawang, "Graph clustering-based emerging event detection from Twitter data stream," in *Proc. 5th Int. Conf. Netw., Commun. Comput. (ICNCC)*, 2016, pp. 37–41, doi: 10.1145/3033288.3033312.

[46] M. T. Altuncu, E. Mayer, S. N. Yaliraki, and M. Barahona, "From free text to clusters of content in health records: An unsupervised graph partitioning approach," *Appl. Netw. Sci.*, vol. 4, no. 1, pp. 1–23, Dec. 2019, doi: 10.1007/S41109-018-0109-9.

[47] M. T. Altuncu, S. N. Yaliraki, and M. Barahona, "Content-driven, unsupervised clustering of news articles through multiscale graph partitioning," 2018, *arXiv:1808.01175*.

[48] N. Azam, M. Abulaish, and N. A.-H. Haldar, "Twitter data mining for events classification and analysis," in *Proc. 2nd Int. Conf. Soft Comput. Mach. Intell. (ISCMI)*, Nov. 2015, pp. 79–83, doi: 10.1109/ISCMI.2015.33.

[49] M. Abulaish, S. Sharma, and M. Fazil, "A multi-attributed graph-based approach for text data modeling and event detection in Twitter," in *Proc. 11th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2019, pp. 703–708, doi: 10.1109/COMSNETS.2019.8711451.

[50] K. G. Dhal and S. Das, "A dynamically adapted and weighted bat algorithm in image enhancement domain," *Evolving Syst.*, vol. 10, no. 2, pp. 1–19, 2018, doi: 10.1007/S12530-018-9216-1.

[51] J. Sheng, C. Liu, L. Chen, B. Wang, and J. Zhang, "Research on community detection in complex networks based on internode attraction," *Entropy*, vol. 22, no. 12, p. 1383, Dec. 2020, doi: 10.3390/E22121383.

[52] C. E. M. Barbosa and G. C. Vasconcelos, "Eight bio-inspired algorithms evaluated for solving optimization problems," in *Proc. Int. Conf. Artif. Intell. Soft Comput.*, 2018, pp. 290–301, doi: 10.1007/978-3-319-91253-0_28.

[53] W. H. Bangyal, J. Ahmad, H. Tayyab, and S. Pervaiz, "An overview of mutation strategies in bat algorithm," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 8, pp. 523–534, 2018, doi: 10.14569/IJACSA.2018.090866.

[54] H. M. Zawbaa, E. Emary, C. Grosan, and V. Snasel, "Large-dimensionality small-instance set feature selection: A hybrid bio-inspired heuristic approach," *Swarm Evol. Comput.*, vol. 42, pp. 29–42, Oct. 2018.

[55] L. Brezočnik, I. Fister, and V. Podgorelec, "Swarm intelligence algorithms for feature selection: A review," *Appl. Sci.*, vol. 8, no. 9, p. 1521, Sep. 2018, doi: 10.3390/APP8091521.

[56] M. Sharma and P. Kaur, "A comprehensive analysis of nature-inspired meta-heuristic techniques for feature selection problem," *Arch. Comput. Methods Eng.*, vol. 28, no. 3, pp. 1103–1127, May 2021.

[57] S. Maza and M. Touahria, "Feature selection algorithms in intrusion detection system: A survey," *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 10, pp. 5079–5099, 2018.

[58] M. Rostami, K. Berahmand, E. Nasiri, and S. Forouzandeh, "Review of swarm intelligence-based feature selection methods," *Eng. Appl. Artif. Intell.*, vol. 100, Apr. 2021, Art. no. 104210.

[59] V. Yasaswini and S. Baskaran, "An optimization of feature selection for classification using modified bat algorithm," in *Advanced Computing and Intelligent Technologies*, M. Bianchini, V. Piuri, S. Das, and R. N. Shaw, Eds. Singapore: Springer, 2022, pp. 389–399.

[60] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "A new feature selection method to improve the document clustering using particle swarm optimization algorithm," *J. Comput. Sci.*, vol. 25, pp. 456–466, Mar. 2017, doi: 10.1016/j.jocs.2017.07.018.

[61] L. M. Abualigah, A. T. Khader, and M. A. Al-Betar, "Unsupervised feature selection technique based on genetic algorithm for improving the text clustering," in *Proc. 7th Int. Conf. Comput. Sci. Inf. Technol. (CSIT)*, Jul. 2016, pp. 1–6, doi: 10.1109/CSIT.2016.7549453.

[62] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, S. N. Makhadmeh, and Z. A. A. Alyasseri, "An improved text feature selection for clustering using binary grey wolf optimizer," in *Proc. 11th Nat. Tech. Seminar Unmanned Syst. Technol.*, 2021, pp. 503–516, doi: 10.1007/978-981-15-5281-6_34.

[63] L. M. Abualigah, A. T. Khader, M. A. Al-Betar, and M. A. Awadallah, "A krill herd algorithm for efficient text documents clustering," in *Proc. IEEE Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, May 2016, pp. 67–72, doi: 10.1109/ISCAIE.2016.7575039.

[64] A. Dhar, N. S. Dash, and K. Roy, "Efficient feature selection based on modified Cuckoo search optimization problem for classifying web text documents," in *Proc. Int. Conf. Recent Trends Image Process. Pattern Recognit.*, 2018, pp. 640–651, doi: 10.1007/978-981-13-9187-3_57.

[65] M. Tubishat, M. Alswaitti, S. Mirjalili, M. A. Al-Garadi, M. T. Alrashdan, and T. A. Rana, "Dynamic butterfly optimization algorithm for feature selection," *IEEE Access*, vol. 8, pp. 194303–194314, 2020, doi: 10.1109/ACCESS.2020.3033757.

[66] M. M. Mafarja, D. Eleyan, I. Jaber, A. Hammouri, and S. Mirjalili, "Binary dragonfly algorithm for feature selection," in *Proc. Int. Conf. New Trends Comput. Sci. (ICTCS)*, Oct. 2017, pp. 12–17, doi: 10.1109/ICTCS.2017.43.

[67] L. Zhang, K. Mistry, C. P. Lim, and S. C. Neoh, "Feature selection using firefly optimization for classification and regression models," *Decis. Support Syst.*, vol. 106, pp. 64–85, Feb. 2018, doi: 10.1016/j.dss.2017.12.001.

[68] M. J. Meena, K. R. Chandran, A. Karthik, and A. V. Samuel, "An enhanced ACO algorithm to select features for text categorization and its parallelization," *Expert Syst. Appl.*, vol. 39, no. 5, pp. 5861–5871, Apr. 2012, doi: 10.1016/j.eswa.2011.11.081.

[69] K. K. Bharti and P. K. Singh, "Chaotic gradient artificial bee colony for text clustering," *Soft Comput.*, vol. 20, no. 3, pp. 1113–1126, Mar. 2016, doi: 10.1007/S00500-014-1571-7.

[70] M. Tubishat, M. A. M. Abushariah, N. Idris, and I. Aljarah, "Improved whale optimization algorithm for feature selection in Arabic sentiment analysis," *Int. J. Speech Technol.*, vol. 49, no. 5, pp. 1688–1707, May 2019, doi: 10.1007/S10489-018-1334-8.

[71] D. Gupta, J. Arora, U. Agrawal, A. Khanna, and V. H. C. de Albuquerque, "Optimized binary bat algorithm for classification of white blood cells," *Measurement*, vol. 143, pp. 180–190, Sep. 2019, doi: 10.1016/j.measurement.2019.01.002.

[72] R. Ramasamy and S. S. Rani, "Modified binary bat algorithm for feature selection in unsupervised learning," *Int. Arab J. Inf. Technol.*, vol. 15, no. 6, pp. 1060–1067, 2018.

[73] M. W. U. Alam, "Improved binary bat algorithm for feature selection," M.S. thesis, Åbo Akademi Univ., Turku, Finland, 2019.

[74] N. Kushwaha and M. Pant, "Link based BPSO for feature selection in big data text clustering," *Future Gener. Comput. Syst.*, vol. 82, pp. 190–199, May 2018, doi: 10.1016/j.future.2017.12.005.

[75] S. Arora and P. Anand, "Binary butterfly optimization approaches for feature selection," *Expert Syst. Appl.*, vol. 116, pp. 147–160, Feb. 2019, doi: 10.1016/j.eswa.2018.08.051.

[76] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary ant lion approaches for feature selection," *Neurocomputing*, vol. 213, pp. 54–65, Nov. 2016, doi: 10.1016/j.neucom.2016.03.101.

[77] T. Thaher, A. A. Heidari, M. Mafarja, J. S. Dong, and S. Mirjalili, "Binary Harris Hawks optimizer for high-dimensional, low sample size feature selection," in *Evolutionary Machine Learning Techniques*. Singapore: Springer, 2020, pp. 251–272.

[78] S. Kashef and H. Nezamabadi-pour, "A new feature selection algorithm based on binary ant colony optimization," in *Proc. 5th Conf. Inf. Knowl. Technol.*, May 2013, pp. 50–54, doi: 10.1109/IKT.2013.6620037.

[79] H. Xu, S. Yu, J. Chen, and X. Zuo, "An improved firefly algorithm for feature selection in classification," *Wireless Pers. Commun.*, vol. 102, no. 4, pp. 2823–2834, Oct. 2018, doi: 10.1007/S11277-018-5309-1.

[80] H. Faris, M. M. Mafarja, A. A. Heidari, I. Aljarah, A.-Z. Ala'M, S. Mirjalili, and H. Fujita, "An efficient binary salp swarm algorithm with crossover scheme for feature selection problems," *Knowl.-Based Syst.*, vol. 154, pp. 43–67, Aug. 2018.

[81] D. Rodrigues, X. S. Yang, A. N. De Souza, and J. P. Papa, "Binary flower pollination algorithm and its application to feature selection," in *Recent Advances in Swarm Intelligence and Evolutionary Computation*. Cham, Switzerland: Springer, 2015, pp. 85–100.

[82] A. G. Hussien, A. E. Hassanien, E. H. Houssein, S. Bhattacharyya, and M. Amin, "S-shaped binary whale optimization algorithm for feature selection," in *Recent Trends in Signal and Image Processing*. Singapore: Springer, 2019, pp. 79–87.

[83] D. Rodrigues, L. A. M. Pereira, J. P. Papa, and S. A. T. Weber, "A binary krill herd approach for feature selection," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 1407–1412, doi: 10.1109/ICPR.2014.251.

[84] M. A. Tawhid and K. B. Dsouza, "Hybrid binary bat enhanced particle swarm optimization algorithm for solving feature selection problems," *Appl. Comput. Informat.*, vol. 16, no. 1/2, pp. 117–136, Apr. 2018, doi: 10.1016/j.aci.2018.04.001.

[85] A. S. S. Rani and R. R. Rajalaxmi, "Unsupervised feature selection using binary bat algorithm," in *Proc. 2nd Int. Conf. Electron. Commun. Syst. (ICECS)*, Feb. 2015, pp. 451–456, doi: 10.1109/ECS.2015.7124945.

[86] K. K. Bharti and P. K. Singh, "Hybrid dimension reduction by integrating feature selection with feature extraction method for text clustering," *Expert Syst. Appl.*, vol. 42, no. 6, pp. 3105–3114, Apr. 2015, doi: 10.1016/j.eswa.2014.11.038.

[87] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neuro Comput.*, vol. 260, pp. 302–312, Oct. 2017, doi: 10.1016/j.neucom.2017.04.053.

[88] Y. Bin, Y. Lu, K. Zhu, G. Yang, J. Liu, and H. Yin, "Feature selection based on modified bat algorithm," *IEICE Trans. Inf. Syst.*, vol. 100, no. 8, pp. 1860–1869, 2017, doi: 10.1587/transinf.2016EDP7471.

[89] K. Sörensen, "Metaheuristics—The metaphor exposed," *Int. Trans. Oper. Res.*, vol. 22, no. 1, pp. 3–18, Jan. 2015, doi: 10.1111/ITOR.12001.

[90] R. Y. M. Nakamura, L. A. M. Pereira, K. A. Costa, D. Rodrigues, J. P. Papa, and X.-S. Yang, "BBA: A binary bat algorithm for feature selection," in *Proc. 25th SIBGRAPI Conf. Graph., Patterns Images*, Aug. 2012, pp. 291–297, doi: 10.1109/SIBGRAPI.2012.47.

[91] D. Rodrigues, L. A. M. Pereira, R. Y. M. Nakamura, K. A. P. Costa, X.-S. Yang, A. N. Souza, and J. P. Papa, "A wrapper approach for feature selection based on bat algorithm and optimum-path forest," *Expert Syst. Appl.*, vol. 41, no. 5, pp. 2250–2258, Apr. 2014, doi: 10.1016/j.eswa.2013.09.023.

[92] H. Chen, Q. Hou, L. Han, Z. Hu, Z. Ye, J. Zeng, and J. Yuan, "Distributed text feature selection based on bat algorithm optimization," in *Proc. 10th IEEE Int. Conf. Intell. Data Acquisition Adv. Comput. Syst., Technol. Appl. (IDAACS)*, Sep. 2019, pp. 75–80, doi: 10.1109/IDAACS.2019.8924308.

[93] A.-C. Enache and V. Sgârciu, "Enhanced intrusion detection system based on bat algorithm-support vector machine," in *Proc. 11th Int. Conf. Secur. Cryptogr.*, 2014, pp. 184–189, doi: 10.5220/0005015501840189.

[94] A. M. Taha, A. Mustapha, and S.-D. Chen, "Naive Bayes-guided bat algorithm for feature selection," *Sci. World J.*, vol. 2013, pp. 1–9, Dec. 2013, doi: 10.1155/2013/325973.

[95] H. Doreswamy and U. M. Salma, "A binary bat inspired algorithm for the classification of breast cancer data," *Int. J. Soft Comput. Artif. Intell. Appl.*, vol. 5, no. 2, pp. 1–21, 2016, doi: 10.5121/IJSCAI.2016.5301.

[96] O. A. Alomari, A. T. Khader, M. A. Al-Betar, and L. M. Abualigah, "Gene selection for cancer classification by combining minimum redundancy maximum relevancy and bat-inspired algorithm," *Int. J. Data Min. Bioinform.*, vol. 19, no. 1, pp. 32–51, 2017, doi: 10.1504/IJDMB.2017.10009480.
[97] A.-C. Enache and V. Sgarciu, "A feature selection approach implemented with the binary bat algorithm applied for intrusion detection," in *Proc. 38th Int. Conf. Telecommun. Signal Process. (TSP)*, Jul. 2015, pp. 11–15.
[98] A.-C. Enache and V. Sgarciu, "An improved bat algorithm driven by support vector machines for intrusion detection," in *Proc. Comput. Intell. Secur. Inf. Syst. Conf.*, 2015, pp. 41–51.
[99] A.-C. Enache, V. Sgarciu, and M. Togan, "Comparative study on feature selection methods rooted in swarm intelligence for intrusion detection," in *Proc. 21st Int. Conf. Control Syst. Comput. Sci. (CSCS)*, May 2017, pp. 239–244, doi: 10.1109/CSCS.2017.40.
[100] X. Xie, X. Qin, Q. Zhou, Y. Zhou, T. Zhang, R. Janicki, and W. Zhao, "A novel test-cost-sensitive attribute reduction approach using the binary bat algorithm," *Knowl.-Based Syst.*, vol. 186, Dec. 2019, Art. no. 104938, doi: 10.1016/J.KNOSYS.2019.104938.
[101] S. Taghian, M. H. Nadimi-Shahraki, and H. Zamani, "Comparative analysis of transfer function-based binary metaheuristic algorithms for feature selection," in *Proc. Int. Conf. Artif. Intell. Data Process. (IDAP)*, Sep. 2018, pp. 1–6.
[102] G. Li and C. Le, "Hybrid binary bat algorithm with cross-entropy method for feature selection," in *Proc. 4th Int. Conf. Control Robot. Eng. (ICCRE)*, Apr. 2019, pp. 165–169.
[103] O. S. Qasim and Z. Y. Algamal, "Feature selection using different transfer functions for binary bat algorithm," *Int. J. Math., Eng. Manage. Sci.*, vol. 5, no. 4, pp. 697–706, Aug. 2020.
[104] R. Yaghoubzadeh, S. R. Kamel, H. Barzgar, and B. M. San'ati, "The use of the binary bat algorithm in improving the accuracy of breast cancer diagnosis," *Multidisciplinary Cancer Invest.*, vol. 5, no. 1, pp. 1–7, Jan. 2021.
[105] A. Seetharaman and A. C. Sundersingh, "Gene selection and classification using correlation feature selection based binary bat algorithm with greedy crossover," *Concurrency Comput., Pract. Exper.*, vol. 34, no. 5, p. e6718, Feb. 2022.
[106] K. Chatra, V. Kuppili, D. R. Edla, and A. K. Verma, "Cancer data classification using binary bat optimization and extreme learning machine with a novel fitness function," *Med. Biol. Eng. Comput.*, vol. 57, no. 12, pp. 2673–2682, Dec. 2019.
[107] G. Janani, S. P. Rajamohana, M. Anitha, and K. Umamaheswari, "Feature selection using hybrid approach for opinion spam detection," *Int. J. Eng. Sci.*, vol. 6, no. 9, pp. 61–72, 2017.
[108] S. Maheswari and K. Arunesh, "Unsupervised binary BAT algorithm based network intrusion detection system using enhanced multiple classifiers," in *Proc. Int. Conf. Smart Electron. Commun. (ICOSEC)*, Sep. 2020, pp. 885–889.
[109] N. Saleem, K. Zafar, and A. Sabzwari, "Enhanced feature subset selection using niche based bat algorithm," *Computation*, vol. 7, no. 3, p. 49, Sep. 2019.
[110] K. Atefi, H. Hashim, and T. Khodadadi, "A hybrid anomaly classification with deep learning (DL) and binary algorithms (BA) as optimizer in the intrusion detection system (IDS)," in *Proc. 16th IEEE Int. Colloq. Signal Process. Appl. (CSPA)*, Feb. 2020, pp. 29–34.
[111] S. Akila and S. A. Christe, "A wrapper based binary bat algorithm with greedy crossover for attribute selection," *Expert Syst. Appl.*, vol. 187, Jan. 2022, Art. no. 115828.
[112] Y. Upadhyaya, S. Sachin, A. Tripathi, and R. Jain, "Comparative analysis of feature selection in epilepsy seizure recognition using cuckoo, gravitational search and bat algorithm," *Int. J. Inf. Syst. Manag. Sci.*, vol. 1, no. 1, pp. 1–8, 2018.
[113] A. M. Taha, S.-D. Chen, and A. Mustapha, "Bat algorithm based hybrid filter-wrapper approach," *Adv. Oper. Res.*, vol. 2015, pp. 1–5, Sep. 2015, doi: 10.1155/2015/961494.
[114] A.-C. Enache and V. Sgarciu, "Anomaly intrusions detection based on support vector machines with an improved bat algorithm," in *Proc. 20th Int. Conf. Control Syst. Comput. Sci.*, May 2015, pp. 317–321, doi: 10.1109/CSCS.2015.12.
[115] B. Raman and T. R. Ioerger, "Instance-based filter for feature selection," *J. Mach. Learn. Res.*, vol. 1, no. 3, pp. 1–23, 2002.
[116] Z. Zhu, Y.-S. Ong, and M. Dash, "Markov blanket-embedded genetic algorithm for gene selection," *Pattern Recognit.*, vol. 40, no. 11, pp. 3236–3248, Nov. 2007.
[117] J.-J. Xing, Y.-F. Liu, Y.-Q. Li, H. Gong, and Y.-P. Zhou, "QSAR classification model for diverse series of antimicrobial agents using classification tree configured by modified particle swarm optimization," *Chemometric Intell. Lab. Syst.*, vol. 137, pp. 82–90, Oct. 2014.
[118] N. Khatri, V. Lather, and A. K. Madan, "Diverse classification models for anti-hepatitis C virus activity of thiourea derivatives," *Chemometric Intell. Lab. Syst.*, vol. 140, pp. 13–21, Jan. 2015.
[119] Y. Li, Y. Kong, M. Zhang, A. Yan, and Z. Liu, "Using support vector machine (SVM) for classification of selectivity of H1N1 neuraminidase inhibitors," *Mol. Informat.*, vol. 35, nos. 3–4, pp. 116–124, Apr. 2016.
[120] A. H. Lashkari, A. Seo, G. D. Gil, and A. Ghorbani, "CIC-AB: Online ad blocker for browsers," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2017, pp. 1–7.
[121] M. Ott, C. Cardie, and J. T. Hancock, "Negative deceptive opinion spam," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2013, pp. 497–501.
[122] S. H. Rezatofighi and H. Soltanian-Zadeh, "Automatic recognition of five types of white blood cells in peripheral blood," *Computerized Med. Imag. Graph.*, vol. 35, no. 4, pp. 333–343, Jun. 2011.
[123] Y. Saeys, I. Inza, and P. Larrañaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, Oct. 2007.
[124] B. Zhang, H. Yuan, L. Sun, J. Shi, Z. Ma, and L. Zhou, "A two-stage framework for bat algorithm," *Neural Comput. Appl.*, vol. 28, no. 9, pp. 2605–2619, Sep. 2017, doi: 10.1007/S00521-016-2192-0.
[125] A. Chakri, R. Khelif, M. Benouaret, and X.-S. Yang, "New directional bat algorithm for continuous optimization problems," *Expert Syst. Appl.*, vol. 69, pp. 159–175, Mar. 2017, doi: 10.1016/j.eswa.2016.10.050.
[126] K. Bharti and P. K. Singh, "Opposition chaotic fitness mutation based adaptive inertia weight BPSO for feature selection in text clustering," *Appl. Soft Comput.*, vol. 43, pp. 20–34, Jun. 2016, doi: 10.1016/j.asoc.2016.01.019.
[127] R. S. Parpinelli, G. F. Plichoski, R. S. Da Silva, and P. H. Narloch, "A review of techniques for online control of parameters in swarm intelligence and evolutionary computation algorithms," *Int. J. Bio-Inspired Comput.*, vol. 13, no. 1, pp. 1–20, 2019, doi: 10.1504/IJBIC.2019.097731.
[128] I. Fister, Jr., I. Fister, and X. S. Yang, "Towards the development of a parameter-free bat algorithm," in *Proc. 2nd Student Comput. Sci. Res. Conf.*, 2015, pp. 31–34.
[129] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 124–141, Jul. 1999, doi: 10.1109/4235.771166.
[130] J. Zhang, W.-N. Chen, Z.-H. Zhan, W.-J. Yu, Y.-L. Li, N. Chen, and Q. Zhou, "A survey on algorithm adaptation in evolutionary computation," *Frontiers Electr. Electron. Eng.*, vol. 7, no. 1, pp. 16–31, Mar. 2012, doi: 10.1007/S11460-012-0192-0.
[131] C. Wang, S. Zhou, Y. Gao, and C. Liu, "A self-adaptive bat algorithm for the truck and trailer routing problem," *Eng. Comput.*, vol. 35, no. 1, pp. 108–135, Mar. 2018, doi: 10.1108/EC-11-2016-0408.
[132] A. Aleti and I. Moser, "Studying feedback mechanisms for adaptive parameter control in evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2013, pp. 3117–3124, doi: 10.1109/CEC.2013.6557950.
[133] I. N. Trivedi, J. Pradeep, J. Narottam, K. Arvind, and L. Dilip, "Novel adaptive whale optimization algorithm for global optimization," *Indian J. Sci. Technol.*, vol. 9, no. 38, pp. 319–326, Aug. 2016, doi: 10.17485/ijst/2016/v9i38/101939.
[134] N. Boukhari, F. Debbat, N. Monmarché, and M. Slimane, "A study on self-adaptation in the evolutionary strategy algorithm," in *Proc. IFIP Int. Conf. Comput. Intell. Appl.*, 2018, pp. 150–160, doi: 10.1007/978-3-319-89743-1_14.
[135] M. G. P. de Lacerda, "Out-of-the-box parameter control for evolutionary and swarm-based algorithms with distributed reinforcement learning," M.S. thesis, Univ. Federal de Pernambuco, 2021.
[136] M. G. P. de Lacerda, L. F. de Araujo Pessoa, F. B. de Lima Neto, T. B. Ludermir, and H. Kuchen, "A systematic literature review on general parameter control for evolutionary and swarm-based algorithms," *Swarm Evol. Comput.*, vol. 60, Feb. 2021, Art. no. 100777.
[137] A. H. Gandomi and X.-S. Yang, "Chaotic bat algorithm," *J. Comput. Sci.*, vol. 5, no. 2, pp. 224–232, Mar. 2014, doi: 10.1016/j.jocs.2013.10.002.
[138] M. R. Ramli, Z. A. Abas, M. I. Desa, Z. Z. Abidin, and M. B. Alazzam, "Enhanced convergence of bat algorithm based on dimensional and inertia weight factor," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 31, no. 4, pp. 452–458, Oct. 2019, doi: 10.1016/j.jksuci.2018.03.010.

[139] X.-X. Ma and J.-S. Wang, "Optimized parameter settings of binary bat algorithm for solving function optimization problems," *J. Electr. Comput. Eng.*, vol. 2018, pp. 1–9, May 2018.

[140] J. Perez, F. Valdez, and O. Castillo, "Modification of the bat algorithm using fuzzy logic for dynamical parameter adaptation," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 464–471, doi: 10.1109/CEC.2015.7256926.

[141] J. Perez, F. Valdez, O. Castillo, and O. Roeva, "Bat algorithm with parameter adaptation using interval type-2 fuzzy logic for benchmark mathematical functions," in *Proc. IEEE 8th Int. Conf. Intell. Syst. (IS)*, Sep. 2016, pp. 120–127, doi: 10.1109/IS.2016.7737409.

[142] I. Fister, S. Fong, and J. Brest, "A novel hybrid self-adaptive bat algorithm," *Sci. World J.*, vol. 2014, pp. 709–738, Apr. 2014, doi: 10.1155/2014/709738.

[143] X. Liu and D. Qi, "A self-adaptive bat algorithm for camera calibration," in *Proc. 2nd Int. Conf. Adv. Mech. Eng. Ind. Informat. (AMEII)*, 2016, pp. 1–7.

[144] A. Baziar, A. Kavoosi-Fard, and J. Zare, "A novel self adaptive modification approach based on bat algorithm for optimal management of renewable MG," *J. Intell. Learn. Syst. Appl.*, vol. 5, no. 1, p. 11, 2013, doi: 10.4236/JILSA.2013.51002.

[145] S. Lyu, Z. Li, Y. Huang, J. Wang, and J. Hu, "Improved self-adaptive bat algorithm with step-control and mutation mechanisms," *J. Comput. Sci.*, vol. 30, pp. 65–78, Jan. 2019, doi: 10.1016/j.jocs.2018.11.002.

[146] H. Afrabandpey, M. Ghaffari, A. Mirzaei, and M. Safayani, "A novel bat algorithm based on chaos for optimization tasks," in *Proc. Iranian Conf. Intell. Syst. (ICIS)*, Feb. 2014, pp. 1–6, doi: 10.1109/IranianCIS.2014.6802527.

[147] R. Jordehi, "Chaotic bat swarm optimisation (CBSO)," *Appl. Soft Comput.*, vol. 26, no. 1, pp. 523–530, Jan. 2015, doi: 10.1016/j.asoc.2014.10.010.

[148] X. Cai, L. Wang, Q. Kang, and Q. Wu, "Bat algorithm with Gaussian walk," *Int. J. Bio-Inspired Comput.*, vol. 6, no. 3, pp. 166–174, 2014, doi: 10.1504/IJBIC.2014.062637.

[149] B. H. Nguyen, B. Xue, and M. Zhang, "A survey on swarm intelligence approaches to feature selection in data mining," *Swarm Evol. Comput.*, vol. 54, May 2020, Art. no. 100663.

[150] S. Cheng, B. Liu, Y. Shi, Y. Jin, and B. Li, "Evolutionary computation and big data: Key challenges and future directions," in *Proc. Int. Conf. Data Mining Big Data*, 2016, pp. 3–14.

[151] B. H. Nguyen, B. Xue, and P. Andreae, "A novel binary particle swarm optimization algorithm and its applications on knapsack and feature selection problems," in *Intelligent and Evolutionary Systems*. Cham, Switzerland: Springer, 2017, pp. 319–332.

[152] D. H. Wolper and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997, doi: 10.1109/4235.585893.

[153] J. Yousef, A. Youssef, and A. Keshk, "A hybrid swarm intelligence based feature selection algorithm for high dimensional datasets," *Int. J. Comput. Inf.*, vol. 8, no. 1, pp. 67–86, 2021.

[154] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO)*. Berlin, Germany: Springer, 2010, pp. 65–74, doi: 10.1007/978-3-642-12538-6_6.

[155] Y. Yuan, X. Wu, P. Wang, and X. Yuan, "Application of improved bat algorithm in optimal power flow problem," *Appl. Intell.*, vol. 48, no. 8, pp. 2304–2314, Aug. 2018, doi: 10.1007/S10489-017-1081-2.

[156] Y. Zhou, L. Li, and M. Ma, "A complex-valued encoding bat algorithm for solving 0–1 knapsack problem," *Neural Process. Lett.*, vol. 44, no. 2, pp. 407–430, Oct. 2016, doi: 10.1007/S11063-015-9465-Y.

[157] E. Emary, W. Yamany, and A. E. Hassanien, "New approach for feature selection based on rough set and bat algorithm," in *Proc. 9th Int. Conf. Comput. Eng. Syst. (ICCES)*, Dec. 2014, pp. 346–353, doi: 10.1109/ICCES.2014.7030984.

[158] D. Zhao and Y. He, "Chaotic binary bat algorithm for analog test point selection," *Anal. Integr. Circuits Signal Process.*, vol. 84, no. 2, pp. 201–214, Aug. 2015, doi: 10.1007/S10470-015-0548-5.

[159] W. Z. Al-Dyani, F. K. Ahmad, and S. S. Kamaruddin, "Improvements of bat algorithm for optimal feature selection: A systematic literature review," *Intell. Data Anal.*, vol. 26, no. 1, pp. 5–31, Jan. 2022, doi: 10.3233/IDA-205455.

[160] S. Sabba and S. Chikhi, "A discrete binary version of bat algorithm for multidimensional knapsack problem," *Int. J. Bio-Inspired Comput.*, vol. 6, no. 2, pp. 140–152, 2014, doi: 10.1504/IJBIC.2014.060598.

[161] R. Jensi and G. W. Jiji, "MBA-LF: A new data clustering method using modified bat algorithm and Levy flight," *ICTACT J. Soft Comput.*, vol. 6, no. 1, pp. 1093–1101, 2015, doi: 0.21917/IJSC.2015.0151.

[162] Z. Ye, X. Hou, X. Zhang, and J. Yang, "Application of bat algorithm for texture image classification," *Int. J. Intell. Syst. Appl.*, vol. 10, no. 5, pp. 42–50, May 2018, doi: 10.5815/IJISA.2018.05.05.

[163] I. Messaoudi and N. Kamel, "A multi-objective bat algorithm for community detection on dynamic social networks," *Int. J. Speech Technol.*, vol. 49, no. 6, pp. 2119–2136, Jun. 2019, doi: 10.1007/S10489-018-1386-9.

[164] O. Abdel-Raouf, M. Abdel-Baset, and I. El-Henawy, "An improved chaotic bat algorithm for solving integer programming problems," *Int. J. Mod. Educ. Comput. Sci.*, vol. 6, no. 8, p. 18, 2014, doi: 10.5815/IJMECS.2014.08.03.

[165] J. Li, Z. Zhao, R. Li, H. Zhang, and T. Zhang, "Ai-based two-stage intrusion detection for software defined IoT networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2093–2102, Nov. 2018.

[166] R. Hassanian-Esfahani and M.-J. Kargar, "A survey on web news retrieval and mining," in *Proc. 2nd Int. Conf. Web Res. (ICWR)*, Apr. 2016, pp. 90–101, doi: 10.1109/ICWR.2016.7498452.

[167] X. Yang and X. He, "Bat algorithm: Literature review and applications," *Int. J. Bio-Inspired Com.*, vol. 5, pp. 141–149, Aug. 2013, doi: 10.1504/IJBIC.2013.055093.

[168] M. Chawla and M. Duhan, "Bat algorithm: A survey of the state-of-the-art," *Appl. Artif. Intell.*, vol. 29, no. 6, pp. 617–634, 2015.

[169] S. Induja and V. P. Eswaramurthy, "Bat algorithm: An overview and its applications," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 5, no. 1, pp. 448–451, 2016.

[170] S. Sharma, A. K. Luhach, and K. Jyoti, "Research and analysis of advancements in bat algorithm," in *Proc. 3rd Int. Conf. Comput. Sustain. Global Develop. (INDIACom)*, 2016, pp. 2391–2396.

[171] S. L. Yadav and M. Phogat, "A review on bat algorithm," *Int. J. Comput. Sci. Eng.*, vol. 5, no. 7, pp. 39–43, Jul. 2017, doi: 10.26438/ijcse/v5i7.3943.

[172] W. Kongkaew, "Bat algorithm in discrete optimization: A review of recent applications," *Songklanakarin J. Sci. Technol.*, vol. 39, no. 5, pp. 641–650, 2017.

[173] F. Liu, X. Yan, and Y. Lu, "Feature selection for image steganalysis using binary bat algorithm," *IEEE Access*, vol. 8, pp. 4244–4249, 2020.

[174] S. Deshpande, M. Doke, A. Deshpande, and A. N. Chaudhari, "Expert system for retrieval of documents using evolutionary approaches incorporating clustering," in *Proc. Int. Conf. Electron., Commun. Aerosp. Technol. (ICECA)*, Apr. 2017, pp. 414–418.

[175] O. A. Alomari, A. Tajudin, K. Mohammed, and A. A. Mohammed, "A novel gene selection method using modified MRMR and hybrid bat-inspired algorithm with β-hill climbing," *Appl. Intell.*, vol. 48, pp. 4429–4447, Nov. 2018, doi: 10.1007/S10489-018-1207-1.

[176] P. V. B. Reddy, S. P. Nandyala, and J. S. Devi, "Speaker verification with optimized feature subset using MOBA," in *Proc. IEEE Distrib. Comput., VLSI, Electr. Circuits Robot. (DISCOVER)*, Aug. 2016, pp. 101–106, doi: 10.1109/DISCOVER.2016.7806242.

[177] R. M. Rizk-Allah and A. E. Hassanien, "New binary bat algorithm for solving 0–1 knapsack problem," *Complex Intell. Syst.*, vol. 4, no. 1, pp. 31–53, Mar. 2018, doi: 10.1007/S40747-017-0050-Z.

[178] S. Yılmaz and E. U. Küçüksille, "A new modification approach on bat algorithm for solving optimization problems," *Appl. Soft Comput.*, vol. 28, pp. 259–275, Mar. 2015, doi: 10.1016/j.asoc.2014.11.029.

[179] G.-G. Wang, M. Lu, and X.-J. Zhao, "An improved bat algorithm with variable neighborhood search for global optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 1773–1778, doi: 10.1109/CEC.2016.7744003.

[180] X. Cai, X.-Z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *Int. J. Bio-Inspired Comput.*, vol. 8, no. 4, pp. 205–214, 2016, doi: 10.1504/IJBIC.2016.078666.

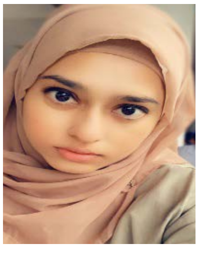
[181] K. Lei and C. Pu, "Complex optimization problems using highly efficient particle swarm optimizer," *Telkomnika*, vol. 12, no. 4, pp. 1023–1030, 2014, doi: 10.12928/TELKOMNIKA.v12i4.535.

[182] S. M. Van Dongen, "Graph clustering by flow simulation," Ph.D. thesis, Univ. Utrecht, Utrecht, The Netherlands, 2000.

[183] A. Bustamam, M. S. Wisnubroto, and D. Lestari, "Analysis of protein-protein interaction network using Markov clustering with pigeon-inspired optimization algorithm in HIV (human immunodeficiency virus)," in *Proc. AIP Conf.*, 2018, p. 20229, doi: 10.1063/1.5064226.

[184] A. Azad, G. A. Pavlopoulos, C. A. Ouzounis, N. C. Kyrpides, and A. Buluç, "HipMCL: A high-performance parallel implementation of the Markov clustering algorithm for large-scale networks," *Nucleic Acids Res.*, vol. 46, no. 6, p. e33, Apr. 2018, doi: 10.1093/nar/gkx1313.
[185] F. Ahmed and M. Abulaish, "An MCL-based approach for spam profile detection in online social networks," in *Proc. IEEE 11th Int. Conf. Trust, Secur. Privacy Comput. Commun.*, Jun. 2012, pp. 602–608, doi: 10.1109/TrustCom.2012.83.
[186] E. I. Setiawan, C. P. Susanto, J. Santoso, S. Sumpeno, and M. H. Purnomo, "Preliminary study of spam profile detection for social media using Markov clustering: Case study on Javanese people," in *Proc. Int. Comput. Sci. Eng. Conf. (ICSEC)*, 2016, pp. 1–4, doi: 10.1007/978-3-319-60000-0_5.
[187] T. Hospedales, S. Gong, and T. Xiang, "A Markov clustering topic model for mining behaviour in video," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1165–1172, doi: 10.1109/ICCV.2009.5459342.
[188] Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, "Learning Markov clustering networks for scene text detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 6936–6944, doi: 10.1109/CVPR.2018.00725.
[189] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, 2010, doi: 10.1016/j.physrep.2009.11.002.
[190] L. M. Abualigah and A. T. Khader, "Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering," *J. Supercomput.*, vol. 73, no. 11, pp. 4773–4795, 2017, doi: 10.1109/CSIT.2016.7549453.
[191] L. M. Abualigah, A. T. Khader, M. A. AlBetar, and E. S. Hanandeh, "Unsupervised text feature selection technique based on particle swarm optimization algorithm for improving the text clustering," in *Proc. 1st EAI Int. Conf. Comput. Sci. Eng.*, 2017, p. 169, doi: 10.4108/EAI.27-2-2017.152282.
[192] X. Huang, X. Zhang, Y. Ye, S. Deng, and X. Li, "A topic detection approach through hierarchical clustering on concept graph," *Appl. Math. Inf. Sci.*, vol. 7, no. 6, p. 2285, 2013, doi: 10.12785/AMIS/070619.
[193] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, no. 6, p. 66133, Jun. 2004, doi: 10.1103/PhysRevE.69.066133.
[194] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh, "Hybrid clustering analysis using improved krill herd algorithm," *Appl. Intell.*, vol. 48, no. 11, pp. 4047–4071, 2018, doi: 10.3390/A10020056.
[195] S.-S. Hong, W. Lee, and M.-M. Han, "The feature selection method based on genetic algorithm for efficient of text clustering and text classification," *Int. J. Adv. Soft Comput. Appl.*, vol. 7, no. 1, pp. 22–40, 2015, [Online]. Available: https://scholarworks.bwise.kr/gachon/handle/2020.sw.gachon/11004
[196] R. M. Rizk-Allah, A. E. Hassanien, M. Elhoseny, and M. Gunasekaran, "A new binary salp swarm algorithm: Development and application for optimization tasks," *Neural Comput. Appl.*, vol. 31, no. 5, pp. 1641–1663, May 2019.
[197] E. Emary, H. M. Zawbaa, and A. E. Hassanien, "Binary grey wolf optimization approaches for feature selection," *Neurocomputing*, vol. 172, pp. 371–381, Jan. 2016.
[198] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 80, no. 5, pp. 1–12, Nov. 2009, doi: 10.1103/PhysRevE.80.056117.
[199] S. M. Szilágyi and L. Szilágyi, "A fast hierarchical clustering algorithm for large-scale protein sequence data sets," *Comput. Biol. Med.*, vol. 48, pp. 94–101, May 2014, doi: 10.1016/j.compbiomed.2014.02.016.

WAFA ZUBAIR AL-DYANI received the bachelor's degree (Hons.) in computer science from Hadhramout University, Yemen, in 2008, the M.S. degree in information technology (IT) from the Universiti Utara Malaysia (UUM), in 2016, and the Ph.D. degree in IT (text mining) from the School of Computing, UUM. She is currently a Lecturer with the Department of Computer Science, College of Computers and Information Technology, Hadhramout University, Hadhramout, Yemen. Her research interests include text mining, natural language processing, machine learning, and optimization algorithms (bat algorithm).



FARZANA KABIR AHMAD received the bachelor's (Hons.) and master's degrees in computer science from the Universiti Sains Malaysia, in 2003 and 2005, respectively, and the Ph.D. degree in computer science (bioinformatics) from the Universiti Teknologi Malaysia, in 2012. She is currently a Senior Lecturer at the School of Computing, Universiti Utara Malaysia, Malaysia. Her doctoral work involves the development of a synergy network for breast cancer progression. Her main research interests include machine learning and data mining projects that seek hidden information from huge, complex datasets, and finally generate/built models to ease the human decision-making process. At the moment, most of her research is related to events detection, fake news detection, and predictive modeling main in social tension environment studies. She is engaging with big data analytics, bio-inspired algorithms, and text mining-based research.

SITI SAKIRA KAMARUDDIN received the Diploma degree in computer science from the Universiti Putra Malaysia (UPM), Serdang, Selangor, in 1990, the bachelor's and master's degrees in computer science from the Universiti Teknologi Malaysia (UTM), Skudai, Johor, in 1995 and 1998, respectively, and the Ph.D. degree in science and system management from the Universiti Kebangsaan Malaysia, Bangi, Selangor, in 2011. She is currently an Associate Professor with the School of Computing, Universiti Utara Malaysia (UUM), Sintok, Kedah. She has published various papers in scientific journals and international conferences in the area of computational intelligence, text mining, natural language processing, social media analytics, and data mining.