Journal of Theoretical and Applied Information Technology
31st July 2024. Vol.102. No. 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# IMPROVED SALP SWARM ALGORITHM FOR TEXT DOCUMENT CLUSTERING

**WALEED ABDELKARIM ABUAIN**[1]

[1] College of Science and Computer Engineering, Yanbu, Taibah University, Al-Madinah Al-Munawwarah, 42353, Saudi Arabia

E-mail: [1]wabuain@taibahu.edu.sa

**ABSTRACT**

Text document clustering (TDC) is a crucial task in text mining that involves dividing a collection of documents into subgroups based on their level of similarity or dissimilarity. A vast amount of information is available on text clustering, and numerous attempts have been made to enhance the learning performance and address the TDC problem. One of the latest swarm algorithms based on population is the SALP Swarm Algorithm (SSA), which has been effectively applied to solve many optimization problems. However, the initial performance of SSA is limited to the exploitation phase, resulting in local optima trapping and a low convergence rate. This study proposes a novel approach to improve the SSA algorithm called the link-based SALP Swarm Algorithm (LBSSA), which enhances the exploitation capability of the original SSA. This involves adding an adjacent operator to the algorithm and utilizing a new aspect of probability, namely the neighborhood selection method (NSM), to improve the searching capability. The effectiveness of LBSSA was evaluated using six different text clustering datasets, demonstrating that the modified SSA combined with NSM significantly improved accuracy, precision, recall, F-measure, purity criterion, and convergence rate. Overall, LBSSA outperformed the original SSA algorithm and other popular clustering techniques such as K-means clustering, Density-based Spatial Clustering of Applications with Noise (DBSCAN), Agglomerative, and optimization algorithms such as Harmony Search (HS), Firefly Algorithm (FFA), BAT algorithm, particle swarm optimization (PSO), and Genetic Algorithm (GA).

**Keywords:** *Data Clustering, Test Clustering, Optimization, SALP Swarm Algorithm*

## 1. INTRODUCTION

In such a broad digital-driven era, and because of enormous technological advancements, the growth of the internet and sophisticated online technologies, such as servers with enormous processing power and massive amounts of data, all comprise a regular occurrence. The International Data Corporation (IDC) has published research predicting that 175 zettabytes of data will be generated globally by 2025 [1]. Massive amounts of data are amassing on mainframes, servers, and cloud computing environments accessible to the public. A sizable portion of this enormous amount of data is conveyed in textual form [2]. Numerous applications of text mining have been presented in the available literature. These applications include the enrichment of search engine consequences, uncontrolled text organization systems, knowledge extraction methods, search and retrieval systems, and textual mining algorithms [3]. Additionally, other techniques were presented for efficiently organizing unstructured text documents [4].

Text document clustering (TDC) is a critical problem in unsupervised learning since it addresses the structure of data partitioning in an unfamiliar area [5]. Additionally, it serves as the foundation for any subsequent learning. This field of text mining permits the organizing of massive quantities of written data. The TDC approach divides a set of documents into groups of related forms that pertain to various subjects or classes [6]. However, these classes are not made a priori. This is because the documents are not pre-categorized according to the intended categories (e.g., politics). This article focuses on partition clustering processes since this clustering methodology aims to split a collection of data objects into a more precise subset comprised of similar clusters using objective function minimization, regardless of the hierarchical structure of the set [7].In recent years, metaheuristic algorithms have made significant efforts to address TDC, owing to the inability of traditional

deterministic techniques to discover globally optimum solutions to solve TDC problem [8]. The bulk of algorithms are derivations of evolutionary algorithms (EAs), survival of the fittest, swarm intelligence (SI), and trajectory-based algorithms (TAs) [9, 10]. Nature is, by and large, the source of inspiration for all metaheuristics (i.e., inspired by biology, ethology, or physics). Their constituents display random behavior (because of random variables) and define many parameters that must be adjusted to the task [11].

The SALP Swarm Algorithm (SSA) algorithm is one of several members of a wider collection of swarm intelligence algorithms developed by [12]. The SSA method is a population-based evolutionary methodology. The SSA algorithm has several advantages over other algorithms. It can strike a delicate balance between supply and demand during a particular search. It just entails the initialization of several factors, as it does not require any mathematical deduction from information. Additionally, it is straightforward, adaptable, scalable, and versatile. As a result, SSA was successfully applied to a wide range of optimization problems, including clustering challenges [13].

The optimization capabilities of the SSA have garnered significant attention since its inception in 2017. Researchers have conducted several studies to enhance and modify its performance, as demonstrated by the work of [14]. However, it is essential to remember that no single optimizer can excel in every problem domain, as dictated by the no-free-lunch theorem. Thus, to effectively address complex inverse problems, it is essential to maintain a diverse range of evolutionary methods, such as SSA, which can adapt to the unique characteristics of various optimization landscapes.

Like other optimization techniques, SSA strives to strike a balance between exploration and exploitation during its search. Exploration allows the algorithm to cover large areas of the search space, often using the randomness of the C1 coefficient. Conversely, exploitation enables SSA to fine-tune its search within promising areas it has discovered. This study, however, zeroes in on the exploitation phase of SSA, which is often underemphasized in traditional implementations. These implementations tend to prioritize the optimal solution, potentially overlooking valuable alternative solutions scattered throughout the search space. This limitation significantly curtails the algorithm's ability to explore the solution landscape, particularly in complex inverse problems.

Consequently, there is a pressing need for more literature on the fine-tuning of the exploitation phase of SSA. This research uncovers that SSA often prioritizes the optimal solution, potentially overlooking valuable alternative solutions within the search space. The proposed modifications hold the promise of rectifying this, enabling SSA to explore alternative solutions and thereby expanding the range of potential solutions for inverse problems. While this approach may not always lead to the absolute global optimum, it promises a more comprehensive understanding of the problem landscape, thereby enhancing the solution space and fostering a sense of optimism about the potential of SSA.

The primary purpose of this paper is to introduce a new and improved version of the SSA, known as the link-based SSA (LBSSA), that specifically addresses the challenges involved in TDC. This enhancement incorporates a unique probability component called the neighborhood selection method (NSM), which helps to identify the best neighboring solutions for continued improvement. By leveraging the local search capabilities of SSA, the LBSSA algorithm effectively balances exploration and exploitation strategies throughout the optimization process. A key aspect of our approach involves using the average distance of documents to their respective cluster centroids (ADDC) as a metric for evaluating the quality of the solution. Through these innovative techniques, LBSSA significantly enhances the effectiveness of SSA for TDC tasks.

The proposed strategy was evaluated using several regularly used datasets that are discussed in the article. The acquired findings were compared to those obtained using proven comparison approaches and algorithms. The experimental results demonstrated that LBSSA outperformed well-known clustering techniques such as K-means clustering technique, Density-based spatial clustering of applications with noise (DBSCAN), and Agglomerative, as well as optimization algorithms such as harmony search (HS), GA, PSO, and SSA, as well as the original SSA.

The remainder of this investigation is divided into the following sections. The second section summarizes past research and publications. The third section discusses the TDC model in detail. The fourth section describes the proposed approach in detail. The fifth section contains an analysis, as well as empirical evidence, demonstrating the proposed method’s efficacy. The sixth section of this study

contains the conclusions and recommendations for additional research.

## 2. RELATED WORK

Clustering is a crucial aspect of text mining, as it enables the seamless organization of large data sets into coherent and thematic clusters [15]. However, conventional techniques like K-means and hierarchical clustering can struggle with complex data structures. SSA is a metaheuristic algorithm that offers exciting new possibilities. Our investigation delves into the current research on SSA in the realm of clustering, providing insights into its evolution, strengths, and potential applications. The trajectory towards achievement commences with a cognizant recognition of the constraints inherent in conventional methodologies employed in the analysis of intricate textual data [16]. In this context, the SSA emerges as a bio-inspired computational paradigm, drawing inspiration from the synchronized locomotion patterns exhibited by SALPs within the marine environment. A comprehensive exploration of scholarly literature unveils a spectrum of modifications and progressions applied to the foundational SSA framework.

As previously articulated, a diverse array of metaheuristic optimization techniques has been employed to address the TDC. Among these approaches, the SSA has emerged as a contemporary swarm-based metaheuristic optimization technique, distinguished by its efficacy, versatility, simplicity, and accessibility. In the work by [17], SSA was adapted for the specific task of tuning a stabilizer in a power system. Comparative analyses with alternative algorithms revealed SSA's superior performance in experimental outcomes.

Recent scholarly endeavors have sought to augment SSA's search mechanism and solution quality through hybridization with other algorithms. In the work by [18] hybridized SSA with the differential evolution algorithm to enhance its capacity for feature exploitation, leveraging the local search proficiency of differential evolution. In addressing feature selection concerns, SSA was hybridized with Particle Swarm Optimization (PSO) as detailed in [19], with a primary objective of fortifying SSA's exploration and exploitation methodologies. Empirical findings suggest the efficacy of the proposed hybrid versions in effectively resolving the targeted issues, particularly in the realm of feature selection.

In the work [20] introduced the Multi-objective SALP Swarm Algorithm (MSSA), designed to address multi-objective real-world problems concurrently. MSSA exhibits a unique capability to optimize multiple criteria, such as compactness and silhouette coefficient, rendering it valuable in scenarios marked by conflicting objectives.

The Adaptive SALP Swarm Algorithm (ASSSA), introduced by [21], represents a dynamic augmentation of the SSA ensemble. Its adaptability, characterized by the dynamic adjustment of crucial parameters based on data characteristics, manifests its efficacy in handling diverse datasets and outperforming its standard counterpart.

However, the focus on SSA signifies merely the inception of a broader exploration into nature-inspired algorithms. Comparative assessments with counterparts such as PSO, GWO and ACO illuminate the nuanced advantages of SSA, particularly in terms of convergence speed and solution quality.

In the work by [22], the Locally Weighted SSA (LWSSA) is introduced as a judicious amalgamation of SSA and a local search mechanism. This amalgamation effectively addresses persistent challenges associated with slow convergence and local optima, as evidenced by its superior performance on benchmark datasets when contrasted with the standard SSA.

## 3. PROBLEM WITH TEXT DOCUMENT CLUSTERING

This section introduces Text Document Clustering (TDC) and formulates it as an optimization problem within the framework of optimal control. The TDC problem can be articulated as follows:

Given a set $Data_{Docs}$ comprising $d$ documents, the objective is to partition these documents into a specified number of $k$ of clustered subsets. In Eq.1 the $Data_{Docs}$ is represented as a vector of a series of documents:

$$Data_{Docs} = (Data_{Docs1}, Data_{Docs2},..., Data_{Docsi},..., Data_{Docsd}) \quad (1)$$

Where $Data_{Docsi}$ denotes the *ith* document, and $Data_{Docsd}$ represents the total number of documents in $Data_{Docs}$.

In Eq.2 each cluster is associated with a cluster centroid ($K_{cent}$), which is defined as a vector of terms with weights:

$$K_{cent} = (k_{centroid1}, k_{centroid2},..., k_{centroidj},...,k_{centroidf}) \quad (2)$$

where $K_{cent}$ signifies the centroid of the $k_{th}$ cluster, $k_{centroidj}$ represents the value of position $j$ in the cluster centroid $k$, and $k_{centroidf}$ denotes the $f^{th}$ term. The weights in $K_{cent}$ reflect the importance of each term in defining the cluster [23].

The Vector Space Model (VSM) is employed to quantify the similarity between documents. Each document $Data_{Docsi}$ is represented as a vector in a high-dimensional space, where each dimension corresponds to a unique term in the document collection. The TF-IDF representation is commonly used to capture the importance of terms in a document.

Cosine similarity, as shown in Eq.3, is then employed to measure the similarity between two document vectors. For documents $Data_{Docsi}$ and $Data_{Docsk}$, the cosine similarity ($sim(Data_{Docsi}, Data_{Docsk})$) is given by:

$$sim(Data_{Docs_i}, Data_{Docs_k}) = \frac{\sum_{j=1}^{|T|} tfidf(Data_{Docs_i}, t_j) \cdot tfidf(Data_{Docs_k}, t_j)}{\sqrt{\sum_{j=1}^{|T|} \left(tfidf(Data_{Docs_i}, t_j)\right)^2} \cdot \sqrt{\sum_{j=1}^{|T|} \left(tfidf(Data_{Docs_k}, t_j)\right)^2}} \quad (3)$$

This similarity measure quantifies the cosine of the angle between the document vectors, providing a metric for clustering algorithms to group similar documents together. Subsequently, clustering algorithms such as K-means or hierarchical clustering can be applied to achieve document partition into clusters.

To determine a partition $k_{centroid} = (k_{centroid1}, k_{centroid2},..,k_{centroidj},...,k_{centroidf})$ that meets the following conditions:

- The items that form a comparable cluster are as similar as feasible. However, the items that comprise the various clusters are as distinct as feasible.
- $\bigcup_{k=1}^{K} k_{cnt} = 0$
- $K_{cnt} \cap K_{cnt'} = \emptyset \; if \; K \neq K'$
- $K_{cnt} \neq \varnothing$

Historically, the degree of resemblance between two documents has been used to indicate the closeness or distance of the target documents [24]. The main objective of the clustering method is to increase intra-cluster similarity while decreasing inter-cluster similarity, as measured by Euclidean distances between data points and cluster centroids.

In the process of text document clustering using an optimization algorithm, several phases need to be conducted.

### 3.1 Preparation of Text Documents

Text preparation methods ought to be conducted to lower the quantity of text terms to facilitate the algorithm's duty. The following stages are classified: 1) tokenization, 2) removing stop words, 3) stemming, and 4) calculating the weighting and document representation of terms.

### 3.2 Clustering Algorithm

Clustering is a crucial aspect of data analysis and machine learning. It entails grouping similar data points based on specific features or characteristics, which can reveal hidden patterns, relationships, and structures within a dataset. This technique yields valuable insights into the underlying organization of data, making it a pivotal element in various domains such as image recognition, customer segmentation, anomaly detection, and more [25].

The Euclidean distance measure is a commonly used objective function in text clustering algorithms, which measures the similarity between documents and the cluster centroid. The objective function aims to minimize the distance between texts within each cluster, with similarity measurements typically expressed in terms of differences or closeness.

TDC algorithms can be broadly classified into two types: hierarchical and partitional. Hierarchical techniques can be divided into divisive (top-down) and agglomerative (bottom-up) approaches. The agglomerative approach analyzes each document individually and merges them into homogeneous groups until further merging becomes impractical. In contrast, divisive clustering groups all texts together and then seek to separate them into smaller, homogeneous groups. The main aim of this study is to partition text documents into distinct clusters, also known as a flat partition. These techniques are usually used with a clustering basis, which assigns each cluster its central point to attract related documents. The ultimate goal of these systems is to accurately distribute large volumes of data into a series of heterogeneous clusters, each containing similar texts.

#### 3.2.1 Solution Representation

The representation of potential solutions is crucial in addressing complex problems. This representation is denoted by the vector $X_x = (x_1, x_2,...,x_d)$, where $d$ signifies the number of documents and the respective values of each variable. The following discussion will delve into the intricacies of this solution representation and explain how each element of the vector contributes to defining solutions in the context of clustering. As shown in

Figure 1, $x_i$ represents the *k-th* choice *( $k \in \{1,...,K\}$)*, where *k* denotes a cluster number and K is the total number of clusters. Each dimension corresponds to a distinct document. In Figure 1, there are a total of 20 documents distributed across five clusters. For example, document 14 belongs to cluster 2, and cluster 2 includes five documents {3,5,8,13,14}.

As depicted in Figure 1, the search space is delineated by all permutations of each document in the set 1, ..., *K*, adhering to the constraint of allocating each document to exactly one cluster (hard clustering). Even in the case of $K = 2$, this poses an NP-hard problem. To intuitively showcase these permutations, each optimization solution is presented as an integer vector comprising *d* elements, where $x_i$ designates the cluster to which the document is assigned.

| ID | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| k | 3 | 3 | 2 | 3 | 2 | 1 | 1 | 2 | 4 | 4 | 5 | 1 | 2 | 2 | 5 | 5 | 4 | 1 | 1 | 1 |

*Figure 1. Solution representation*

### 3.2.2 Objective Function

In the realm of optimization, the objective function is the foundation upon which the entire process is built. It establishes the core aim that the optimization algorithm strives to achieve, by quantifying the system's performance or fitness. Choosing an appropriate objective function is crucial, as it shapes the behavior and outcome of the optimization process [26]. The objective function is a mathematical representation that links the feasible solutions in the search space to real values, representing the quality or desirability of each option. In optimization, the primary objective is typically to either maximize or minimize this scalar value, depending on the nature of the problem at hand. For example, cost minimization problems aim to find the solution that minimizes the overall cost, while profit maximization scenarios aim to find the solution that maximizes the profit [27].

The objective function plays a crucial role in optimization, serving as a measure of success that guides the algorithm towards optimal or near-optimal solutions [28]. Depending on the complexity of the underlying optimization problem, the objective function can range from a simple expression to a complex mathematical model that incorporates various parameters and constraints. In real-world applications, the objective function is derived from a system's performance metrics, considering factors such as efficiency, resource utilization, quality, and other relevant considerations. Throughout the optimization process, the algorithm iteratively explores the solution space, evaluating the objective function at each point to guide its search for optimal or near-optimal solutions. The algorithm's ability to navigate the high-dimensional space defined by the objective function is crucial for efficient and accurate exploration.

Ultimately, the objective function encapsulates the essence of the optimization problem, providing a quantifiable measure of fitness that aligns with predefined optimization goals. Careful formulation and understanding of the objective function are imperative in various fields, ranging from engineering and operations research to machine learning and artificial intelligence, to achieve meaningful and applicable results. It is important to emphasize that the objective function utilized in our work is not arbitrary. Instead, it is rooted in established methodologies and prior research. Our deliberate decision to adopt the same objective function as a previous study (referenced as [4]) ensures consistency and comparability with prior work. This deliberate choice enables a direct and meaningful comparison of results, promoting a comprehensive understanding of the optimization algorithm's performance across various contexts or scenarios.

### 3.3 Validation of Clusters

The validity of the clusters generated is essential for ensuring the reliability of insights derived from such clustering algorithms. This paper delves into the dual aspects of validation (i.e., internal, and external) to provide a comprehensive evaluation framework.

- Internal evaluation: Internal evaluation scrutinizes the clusters' structure, aiming to ascertain the coherence within each cluster. Metrics like the objective function [29], ensure that documents within a cluster exhibit thematic and conceptual similarities. Optimizing internal validity measures enhances the precision of clustering algorithms, facilitating the identification of natural groupings within the text data.
- External evaluation: In contrast, external evaluation steps beyond the clusters to assess their quality using external knowledge sources. This may involve pre-existing document labels, known topic hierarchies, human judgment, F-measure, precision, recall, entropy, purity, error rate, and accuracy [29]. The analysis gains insights into aligning clusters with real-world concepts by comparing clustering results to these external references. External validation solutions crucial semantic coherence and thematic alignment questions, establishing a

connection between clustering outcomes and established knowledge domains.

The synergy between internal and external evaluation forms the foundation of robust text document clustering. Internal measures guide algorithm optimization, ensuring the creation of tightly-knit clusters. External validation, on the other hand, assesses the real-world significance of these clusters, grounding them in established knowledge domains. This interplay enhances the potential of clustering, transforming it from a technical exercise into a potent tool for knowledge discovery and information retrieval.

## 4. PROBLEM WITH TEXT DOCUMENT CLUSTERING

During the exploitation phase of the SSA technique, it is critical to note that the search must not be restricted to only the top (i.e., best) solution. In the optimization field, it is widely recognized that more than focusing on the best solution may hinder the attainment of global optima. As a result, exploring alternative solutions is imperative to avoid prematurely converging on a suboptimal solution. Thus, additional alternatives can be better than the best solution by sharing their own positive features along the way to the global optima. This is the primary disadvantage of SSA, the trajectory of search tends toward the best solution without gaining the benefits of other useful alternatives. It is beneficial in this scenario to search for better alternatives in the search space areas.

Neighborhood selection is a critical stage in any optimization process since it affects both the convergence speed and the quality of the final solution. Three phases may be identified in the proposed NNM. These include populating the solutions with the neighbor information matrix in the first phase. The second part is computing the Link function using Eq.5 The third phase assigns a rank for each solution using the Link matrix; the neighborhood of each solution is then picked based on its rank.

### 4.1 Neighbors and Link Function

$U_i$ denotes matrix neighbors for the solution's population that are comparable to $U_i$. let $sim(U_i,U_j)$ to quantify the pairwise similarity of two solutions, $U_i$ and $U_j$, using Eq.4. The values range from 0 to 1, with a greater value signifying more similarity. If $U_i$ and $U_j$ are within a specific distance of one another, they are considered neighbors.

$$sim(U_i,U_j) \geq \Theta, \; with \; 0 \leq \Theta \leq 1 \qquad (4)$$

Where $\Theta$ is a user-defined threshold for determining how similar two solutions must be to be considered neighbors. When the score of Euclidean distance (*sim*) is equal to 1, a document is compelled to have only identical replies as neighbors. When is set to 0, on the other hand, any pair of solutions become neighbors. The user may choose a value that is appropriate for the application.

A neighbor matrix includes information on the neighbors of each solution in a population. A population of n solutions' neighbor matrix is an $n \times n$ adjacency matrix $M$, in which each entry $M_{i,j}$ denotes either 1 or 0, depending on whether solutions $U_i$ and $U_j$ are neighbors.

The summation is computed using the link function link ($U_i$, $U_j$) described in Eq.5.

$$link(U_i,U_j) = \sum_{m=1}^{n} M_{i,m} \times M_{m,j} \qquad (5)$$

After determining the common neighbors, each pair of solutions are given a rank according to the *link* ($U_i$, $U_j$). Solutions with a greater number of common neighbors have higher rank values; it should be noted that the solutions are given a high rank if they are associated with a greater number of common neighbors; solutions with fewer common neighbors receive lower rank values. Ranking can be used to determine which neighbor has the greatest influence on the particle. The link function makes use of neighbor solutions' information to determine the relationship between two solutions and is often recognized as one of the most efficient methods for determining how near two solutions can be. According to this definition, this approach can be utilized as neighborhoods searching operator within SSA in order to enhance the SSA's local search (exploitation) capabilities for text clustering.

### 4.2 The Proposed TDC Method

To preserve the diversity of the population inside the SSA while not considerably slowing convergence, it has been recommended that the SSA use a NNM (LBSSA). Rather guiding the searching process from only the best solution during the exploitation phase, LBSSA searching process is guiding using both the best solution and the best neighbor solution, as indicated by Eq.6, which NNM selects early on for the goal of increasing variance. LBSSA improved the mechanism for the basic SSA and included the following characteristics:

$$x_i^j = \begin{cases} \lfloor best_{NSj} + c_1 \times ((ub_j - lb_j) \times c_2 + lb_j) Bigr \rfloor & c_3 \geq 0, \\ \lfloor best_{NSj} - c_1 \times ((ub_j - lb_j) \times c_2 + lb_j) Bigr \rfloor & c_3 < 0 \end{cases} \qquad (6)$$

Where $best_{NSj}$ represents the $j^{th}$ variable of the best neighbor solution obtained. The Neighbour

Operator ($N_{operator}$) oversees the exploitation phase of each decision variable i in the solution $x_i$ with a probability range of [0.0%,100%] in order to replace it with *j* from the best/best neighbour solution. When compared to the optimal overall solution and optimal neighbor solution, the LBSSA significantly improves solutions by 50%. It should be noted that the magnitude of moves toward the best neighbor solution using the current solution is determined by the value of $N_{operator}$). The bigger the value of the $N_{operator}$), the more probable it is that the present solution will redirect the decision variable away from the best neighbor option. In comparison, a smaller $N_{operator}$) value suggests a higher chance of selecting the present option, hence shifting the decision variable away from the optimum solution (i.e., original SSA).

## 5. RESULTS AND DISCUSSION

This section describes the experimental design, parameter optimization, benchmark datasets, and assessment metrics for the proposed strategy, as well as comparisons to existing techniques. Additionally, this part addresses the LBSSA's complexity and convergence rate before delving into the findings and analyzing them.

### 2.1 Design of Experiments

This section elaborates on the experimental design of the proposed LBSSA method in TDC. The performance of the proposed method is assessed utilizing text datasets. To ensure the proposed technique was consistent with the results, it was run 30 times with the whole datasets. The number of runs (30) was determined according to the literature in order to provide for appropriate validation of the proposed technique and a fair comparison with all competitors. Each time timhe clustering strategy is used (i.e., local-based algorithms), 100 iterations are performed.; this number of iterations has been proved based on extensive experiments to be adequate for the intensified search process to converge. Additionally, the 1000 iterations of population-based algorithms optimize the diversification search method's convergence. Notably, the clustering technique makes use of all documents in the databases. In other words, there is no usage of a dataset partitioning approach such as k-fold cross-validation. As a result, the clustering algorithms described in [13] do not require any cross-validation.

Traditionally, five external assessment indicators are utilized. External quality measurements include accuracy, precision, recall, F-measure, and purity criteria, while the internal quality measure is computed using the sum of distances (the intra-cluster). To undertake a comparative assessment, the evaluation measures' findings were compared to those of ten leading-edge algorithms, including the original SSA. K-means, DBSCAN Agglomerative, FFA, HS, PSO, and GA, all of which used the same objective function. The experiments were carried out in a Linux environment and implemented using MATLAB version 8.3.0.532. The next subsections provide a full summary of the experimental outcomes.

To have a fair comparison, all methods are performed using the same settings and dataset. Additionally, the initial solution(s) and the number of evolutions are identical.

### 2.2 On the Performance of SSAs, the Effect of Link-based Selection

This section examines the effectiveness of link-based selection on the convergence behavior of SSA. Three distinct SSA variations have been introduced. The first is LBSSA, which stands for SSA with link-based

Selection and sets the $N_{operator}$ to 50%. To refresh your memory, $N_{operator}$ is the percentage of the SSA that employs link-based selection. The findings are compared to 9 benchmark techniques, three of which are based on clustering and six on optimization.

Table 1 summarizes the performance of the revised SSA for six TCD quality clusters. The suggested algorithm seems to have outperformed current methods. Additionally, when employed as an external measurement in DS11, the proposed technique outperformed TCD in terms of error rate. It is on the edge of outperforming all other clustering algorithms across all text datasets (i.e., DS1, DS1, DS3, DS4, DS5, and DS6). In comparison to the prior SSA approach, the suggested LBSSA algorithm significantly increased the cluster quality. Due to the high quality of the clusters, all investigations revealed that the proposed LBSSA outperformed overall comparative algorithm datasets. The performance of LBSSA is compared against other competitors using all datasets.

The experimental results exhibited that the proposed LBSSA algorithm is a promising solution for the TDC problem. The performance of the method is compared against other competitors' algorithms. Table 1 summarizes the findings of six TCD experiments in terms of accuracy, precision, recall, F-measure, and purity. The best outcomes are shown in bold. According to the accuracy metric, LBSSA outperformed SSA in five out of six datasets (DS1, DS2, DS3, DS4, and DS6). The evaluation

measure, which is a widely known standard in the text clustering area, indicated that incorporating NSS in the SSA searching strategy beat both the original SSA method and related algorithms.

LBSSA outperformed SSA and FFA in five of six datasets (DS1, DS2, DS3, DS4, and DS5) using the F-measure (i.e., DS6). According to the measurements, the suggested approach algorithm has proven successful results as well as LBSSA efficient comparative performance. In comparison to other similar algorithms, the LBSSA obtained virtually all of the top F-measure results. This conclusion is drawn in light of the purity measure used to verify the proposed method. The comparison between the original SSA algorithm and other competitors' algorithms showed that LBSSA is a more effective algorithm for handling the text clustering problem. Thus, by incorporating the neighborhood technique during the SSA algorithm's exploitation phase, the local search capability was strengthened. A good mix of exploration and exploitation improved the performance of the proposed SSA method, resulting in accurate clustering.

Additionally, the results indicated that metaheuristic clustering algorithms, such as basic HS, GA, PSO, and SSA, outperformed conventional clustering strategies in the majority of circumstances. This could be explained by the inefficiency of clustering algorithms in searching the desired search space.

### 2.3 Convergence Analysis

The study of convergence is crucial to assessing and comprehending metaheuristic algorithms, as it offers valuable insights into their optimization dynamics. In the context of these algorithms, convergence refers to moving towards an optimal or near-optimal solution over a series of iterations [30]. It is a fundamental metric for evaluating the algorithm's effectiveness, revealing both the pace at which it converges to an optimal solution and the stability of that convergence. Researchers systematically analyze the algorithm's behavior as it refines its solution space over time to explore convergence in metaheuristic algorithms. This examination is essential for understanding the algorithm's performance characteristics, such as its ability to navigate complex search spaces, prevent premature convergence and identify optimal solutions. Additionally, convergence analysis enables comparing and contrasting different metaheuristic algorithms, helping to select the most appropriate approach for specific problem domains.

The rate of convergence to the best solution is a robust indicator for assessing metaheuristic algorithms. The convergence rate of the clustering method to the best solution as defined by the ADDC metric is used to assess it. On the DCD, Figure 2 depicts the convergence behavior of LBSSA and comparable optimization methods (i.e., HS, FFA, PSO, GA, and SSA). The researchers ran each dataset 30 times using randomly generated initializations. The average value was then estimated using each algorithm's convergence characteristics.

Convergence of the original SSA method was faster than that of the LBSSA approach because the SSA algorithm was prone to get trapped in local optima, resulting in premature convergence. However, when compared to other well-known algorithms, LBSSA beat the original SSA approach in terms of performance and clustering quality. The findings revealed that the proposed technique (LBSSA) beat the component algorithms by a large margin in terms of cluster quality.

When compared to rival algorithms, LBSSA produced the optimal outcome. As a consequence, the LBSSA that was anticipated performed beautifully. While the SSA technique initially converges more quickly, it may get trapped in local minima, resulting in premature convergence. The LBSSA objective function (ADDC) values followed a smooth curve from the beginning values to the final optimum solution; no abrupt changes occurred. Additionally, the ADDC of the original SSA method converged prematurely.

Although LBSSA outperforms its competitors in terms of competitiveness, it may take longer than SSA to achieve the optimal solution. Moreover, its performance may be unpredictable and affected by the unique characteristics of each dataset, making it challenging to anticipate how it will behave with new ones. LBSSA may also need help with large datasets or complex problem spaces, and the effectiveness of its parameters requires careful tuning. Lastly, LBSSA's effectiveness may be limited to specific optimization problems, limiting its usefulness across domains.

## 6. CONCLUSION AND SUGGESTIONS FOR THE FUTURE

TDC is a commonly used method for partitioning data structures in unexplored areas by creating clusters. Effective digital document clustering relies heavily on grouping techniques. Trajectory-based algorithms tend to get trapped in local optimal solutions around the search's starting point. The population-based SSA approach is highly

effective in exploring multiple regions of the search space simultaneously, with minimal exploitation costs, to approach the optimal solution.

This paper introduces a unique technique called the LBSSA for resolving text document clustering problems. The proposed technique includes a novel update mechanism that learns from neighboring best solutions and enhances the original SSA's local search capabilities during exploitation. By navigating the search space around candidate solutions, including the global solution, the LBSSA generates many diverse solutions. The LBSSA has successfully partitioned five data and six text benchmark datasets autonomously. The results were evaluated based on five criteria: accuracy, precision, recall, F-measure, purity, and convergence behavior. The LBSSA has proven to be highly efficient.

Based on a comparison with state-of-the-art approaches, including clustering and optimization algorithms, the results indicate that the proposed technique has the potential to achieve global optimal solutions for all tested datasets. Unlike other algorithms that get stuck in local minima, the proposed technique enhances cluster homogeneity and improves the performance of five external indicators while maintaining high convergence rates. In the future, researchers can explore applying the proposed approach in real-world scenarios, such as classifying scientific articles and web documents, topic extraction, and automated essay grading. Furthermore, combining the proposed technique with local search strategies can enhance the exploitation capability and initial solutions during optimization. To ensure the robustness of the proposed clustering technique, it is recommended that future research evaluates datasets beyond those used in this study, including a broader range of data types, sizes, and structures. Additionally, it is crucial to explore the adaptability of the technique to dynamic datasets and interdisciplinary collaborations, address scalability and efficiency concerns, and catalyze further refinement and exploration in the dynamic field of clustering algorithms. Although the proposed technique shows significant improvements, it requires further refinement and exploration to keep up with the latest advancements in the field.

## REFERENCES:

[1] Statista, "Data growth worldwide 2010-2025", 2023. Available: "https://www.statista.com/statistics/871513/worldwide-data-created/".

[2] A. Shamshiri, K. R. Ryu, and J. Y. Park, "Text mining and natural language processing in construction", *Automation in Construction*, vol. 158, 2024, pp. 105200.

[3] A. H. Katrawi, R. Abdullah, M. Anbar, I. AlShourbaji, and A. K. Abasi, "Straggler handling approaches in mapreduce framework: a comparative study", *International Journal of Electrical & Computer Engineering*, vol. 11, no. 1, 2021, pp. 2088-8708.

[4] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, S. N. Makhadmeh, and Z. A. A. Alyasseri, "Linkbased multi-verse optimizer for text documents clustering", *Applied Soft Computing*, vol. 87, 2020, pp. 106002.

[5] G. Venkanna and K. Bharati, "Improved meta-heuristic model for text document clustering by adaptive weighted similarity", *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 31, no. 05, 2023, pp. 749–771.

[6] L. Hirsch, "Document clustering with evolved multi-word search queries where the number of classes is unknown", *Available at SSRN 4552319*, 2023.

[7] O. M. Alyasiri, Y.-N. Cheah, and A. K. Abasi, "Hybrid filter-wrapper text feature selection technique for text classification", *2021 International Conference on Communication & Information Technology (ICICT)*. IEEE, 2021, pp. 80–86.

[8] S. Mehrotra, A. Sharan, and N. Varish, "Improving search result clustering using nature inspired approach", *Multimedia Tools and Applications*, 2024, pp. 1–18.

[9] L. Velasco, H. Guerrero, and A. Hospitaler, "A literature review and critical analysis of metaheuristics recently developed", *Archives of Computational Methods in Engineering*, vol. 31, no. 1, 2024, pp. 125–146.

[10] O.A. Alomari, S.N. Makhadmeh, M.A. Al-Betar, Z.A.A. Alyasseri, I.A. Doush, A.K. Abasi, M.A. Awadallah, and R.A. Zitar, "Gene selection for microarray data classification based on gray wolf optimizer enhanced with triz-inspired operators", *Knowledge-Based Systems*, vol. 223, 2021, pp. 107034.

[11] S. N. Makhadmeh, A. T. Khader, M. A. Al-Betar, S. Naim, Z. A. A. Alyasseri, and A. K. Abasi, "Particle swarm optimization algorithm for power scheduling problem using smart battery", in *2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT)*. IEEE, 2019, pp. 672–677.

[12] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp

swarm algorithm: A bio-inspired optimizer for engineering design problems", *Advances in Engineering Software*, vol. 114, 2017, pp. 163–191.

[13] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, Z. A. A. Alyasseri, and S. N. Makhadmeh, "A novel hybrid multi-verse optimizer with k-means for text documents clustering", *Neural Comput. Appl.*, vol. 32, no. 23, 2020, pp. 17703–17729.

[14] A.K. Abasi, A.T. Khader, M.A. Al-Betar, Z.A.A. Alyasseri, S.N. Makhadmeh, M. Al-laham, and S. Naim, "A hybrid salp swarm algorithm with $\beta$-hill climbing algorithm for text documents clustering", *Evolutionary Data Clustering: Algorithms and Applications*, 2021, pp. 129.

[15] B. Ma and H. Zhuge, "Automatic construction of classification dimensions by clustering texts based on common words", *Expert Systems with Applications*, vol. 238, 2024, pp. 122292.

[16] T. Zhao, Y. Zhu, and X. Xie, "Topology structure optimization of evolutionary hierarchical fuzzy systems", *Expert Systems with Applications*, vol. 238, 2024, pp. 121857.

[17] S. Ekinci and B. Hekimoglu, "Parameter optimization of power system stabilizer via salp swarm algorithm", in *2018 5th International Conference on Electrical and Electronic Engineering (ICEEE)*. IEEE, 2018, pp. 143–147.

[18] M. A. Elaziz, L. Li, K. Jayasena, and S. Xiong, "Multiobjective big data optimization based on a hybrid salp swarm algorithm and differential evolution", *Applied Mathematical Modelling*, 2019.

[19] R. A. Ibrahim, A. A. Ewees, D. Oliva, M. A. Elaziz, and S. Lu, "Improved salp swarm algorithm based on particle swarm optimization for feature selection", *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 8, 2019, pp. 3155–3169.

[20] S. P. Mhatugade, G. M. Kakandikar, O. K. Kulkarni, and V. M. Nandedkar, "Development of a multiobjective salp swarm algorithm for benchmark functions and real-world problems", *Optimization for Engineering Problems*, 2019, pp. 101–130.

[21] S. Kassaymeh, S. Abdullah, M. A. Al-Betar, M. Alweshah, M. Al-Laham, and Z. Othman, "Self-adaptive salp swarm algorithm for optimization problems", *Soft Computing*, vol. 26, no. 18, 2022, pp. 9349–9368.

[22] S. I. Mohammed, N. K. Hussein, O. Haddani, M. Aljohani, M. A. Alkahya, and M. Qaraad, "Finetuned cardiovascular risk assessment: Locally weighted salp swarm algorithm in global optimization", *Mathematics*, vol. 12, no. 2, 2024, pp. 243.

[23] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, M. A. Awadallah, O. A. Alomari, "Text documents clustering using modified multiverse optimizer", *International Journal of Electrical and Computer Engineering*, vol. 10, no. 6, 2020, pp. 6361– 6369.

[24] A. K. Abasi, A. T. Khader, M. A. Al-Betar, S. Naim, S. N. Makhadmeh, and Z. A. A. Alyasseri, "A novel ensemble statistical topic extraction method for scientific publications based on optimization clustering", *Multimedia Tools and Applications*, vol. 80, no. 1, 2021, pp. 37–82.

[25] R. Forsati, M. Mahdavi, M. Shamsfard, and M. R. Meybodi, "Efficient stochastic algorithms for document clustering", *Information Sciences*, vol. 220, 2013, pp. 269–291.

[26] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman, and S. Vimal, "25 years of particle swarm optimization: Flourishing voyage of two decades", *Archives of Computational Methods in Engineering*, vol. 30, no. 3, 2023pp. 1663–1725.

[27] J. Kdela, M. Zale´ sˇak, P. Charv´ at, L. Klime´ s, and T. Mauder, "Assessment of the performance of meta-ˇ heuristic methods used for the inverse identification of effective heat capacity of phase change materials", *Expert Systems with Applications*, vol. 238, 2024, pp. 122373.

[28] M. Ra'ed, N. E. A. Al-qudah, M. S. Jawarneh, and A. Al-Khateeb, "A novel improved lemurs optimization algorithm for feature selection problems", *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 8, 2023, pp. 101704.

[29] E. Rendon, I. Abundez, A. Arizmendi, and E. M. Quiroz, "Internal versus external cluster validation´ indexes", *International Journal of computers and communications*, vol. 5, no. 1, 2011, pp. 27–34.

[30] E. V. Altay, O. Altay, and Y. Ozc¸evik, "A comparative study of metaheuristic optimization algorithms for¨ solving real-world engineering design problems", *CMES-Computer Modeling in Engineering & Sciences*, vol. 139, no. 1, 2024.

*Table 1. Results Of Purity, F-Measure, Precision, Recall, And Accuracy for Six Text Clustering Datasets.*

| | | Optimization Algorithms | | | | | | | Clustering Techniques | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **Measure** | **LBSSA** | **SSA** | **FFA** | **BAT** | **HS** | **PSO** | **GA** | **DBSCAN** | **K-means** | **Agglomerative** |
| DS1 | Accuracy | 0.5091 | 0.4530 | 0.3605 | 0.4120 | 0.4369 | 0.4256 | 0.3442 | 0.3796 | 0.4152 | 0.4098 |
| | Precision | 0.5974 | 0.5672 | 0.5625 | 0.5498 | 0.5512 | 0.4032 | 0.3996 | 0.3174 | 0.3739 | 0.3384 |
| | Recall | 0.5545 | 0.4762 | 0.4644 | 0.6535 | 0.4629 | 0.4842 | 0.5155 | 0.4054 | 0.3859 | 0.4654 |
| | F-measure | 0.5742 | 0.5229 | 0.5162 | 0.4395 | 0.5042 | 0.3009 | 0.3939 | 0.2841 | 0.3294 | 0.3763 |
| | Purity | 0.6241 | 0.5615 | 0.5566 | 0.4780 | 0.5472 | 0.4154 | 0.3673 | 0.3864 | 0.3887 | 0.4283 |
| | Rank | 1 | 2 | 5 | 3 | 4 | 6 | 7 | 10 | 9 | 8 |
| DS2 | Accuracy | 0.5271 | 0.4201 | 0.3891 | 0.4713 | 0.3833 | 0.2897 | 0.3012 | 0.2830 | 0.3584 | 0.3425 |
| | Precision | 0.5588 | 0.4539 | 0.4171 | 0.3944 | 0.4119 | 0.3399 | 0.3628 | 0.2889 | 0.3407 | 0.3184 |
| | Recall | 0.4078 | 0.3925 | 0.3663 | 0.3671 | 0.3621 | 0.2959 | 0.2931 | 0.2794 | 0.3448 | 0.3068 |
| | F-measure | 0.4727 | 0.4171 | 0.3918 | 0.3689 | 0.3903 | 0.3003 | 0.2770 | 0.2869 | 0.3405 | 0.2916 |
| | Purity | 0.4737 | 0.4505 | 0.4168 | 0.4322 | 0.4133 | 0.3134 | 0.3220 | 0.2818 | 0.3931 | 0.2893 |
| | Rank | 1 | 2 | 4 | 3 | 5 | 9 | 7 | 10 | 6 | 8 |
| DS3 | Accuracy | 0.5037 | 0.4693 | 0.4421 | 0.3878 | 0.4228 | 0.3565 | 0.3139 | 0.2845 | 0.3558 | 0.3158 |
| | Precision | 0.5635 | 0.5207 | 0.4963 | 0.4011 | 0.4870 | 0.4163 | 0.3490 | 0.2992 | 0.3992 | 0.4300 |
| | Recall | 0.6495 | 0.4592 | 0.4273 | 0.3776 | 0.4191 | 0.3230 | 0.2817 | 0.3029 | 0.3563 | 0.2847 |
| | F-measure | 0.6006 | 0.4894 | 0.4596 | 0.4170 | 0.4475 | 0.4257 | 0.2703 | 0.3829 | 0.3974 | 0.3552 |
| | Purity | 0.4880 | 0.5626 | 0.5344 | 0.5215 | 0.5187 | 0.4773 | 0.3596 | 0.4518 | 0.4606 | 0.3920 |
| | Rank | 1 | 2 | 3 | 5 | 4 | 6 | 10 | 9 | 7 | 8 |
| DS4 | Accuracy | 0.5864 | 0.4824 | 0.4520 | 0.4728 | 0.4410 | 0.3387 | 0.3845 | 0.3491 | 0.4023 | 0.3587 |
| | Precision | 0.5150 | 0.4750 | 0.4445 | 0.4607 | 0.4365 | 0.3505 | 0.3988 | 0.4030 | 0.3568 | 0.3426 |
| | Recall | 0.5467 | 0.4622 | 0.4329 | 0.5383 | 0.4218 | 0.3678 | 0.4362 | 0.4082 | 0.3355 | 0.3278 |
| | F-measure | 0.5316 | 0.4799 | 0.4463 | 0.4356 | 0.4356 | 0.3488 | 0.3933 | 0.3943 | 0.4091 | 0.3484 |
| | Purity | 0.5325 | 0.6233 | 0.5998 | 0.4289 | 0.5869 | 0.4651 | 0.5842 | 0.5810 | 0.5270 | 0.4653 |
| | Rank | 1 | 2 | 3 | 4 | 5 | 9 | 6 | 7 | 8 | 10 |
| DS5 | Accuracy | 0.5559 | 0.5338 | 0.5034 | 0.6267 | 0.5084 | 0.4829 | 0.4988 | 0.4636 | 0.4733 | 0.4695 |
| | Precision | 0.5853 | 0.5319 | 0.5013 | 0.5414 | 0.5008 | 0.4422 | 0.4567 | 0.4788 | 0.4713 | 0.4234 |
| | Recall | 0.5201 | 0.4572 | 0.4274 | 0.4376 | 0.4260 | 0.4179 | 0.3872 | 0.3970 | 0.3303 | 0.3217 |
| | F-measure | 0.5632 | 0.4905 | 0.4621 | 0.4655 | 0.4618 | 0.4399 | 0.3898 | 0.4658 | 0.4253 | 0.3932 |
| | Purity | 0.4021 | 0.6152 | 0.5809 | 0.5211 | 0.5863 | 0.5269 | 0.5714 | 0.3807 | 0.5588 | 0.4909 |
| | Rank | 2 | 1 | 5 | 3 | 4 | 6 | 7 | 9 | 8 | 10 |
| DS6 | Accuracy | 0.7567 | 0.7296 | 0.7111 | 0.6049 | 0.6838 | 0.5761 | 0.5507 | 0.6004 | 0.6590 | 0.6123 |
| | Precision | 0.6870 | 0.7260 | 0.7032 | 0.5928 | 0.6716 | 0.5855 | 0.6013 | 0.5468 | 0.6677 | 0.6412 |
| | Recall | 0.7563 | 0.7106 | 0.6986 | 0.5744 | 0.6616 | 0.5457 | 0.5347 | 0.5590 | 0.6827 | 0.6322 |
| | F-measure | 0.7019 | 0.7218 | 0.6993 | 0.5995 | 0.6666 | 0.5548 | 0.5319 | 0.5969 | 0.6698 | 0.6563 |
| | Purity | 0.7620 | 0.7027 | 0.6808 | 0.5556 | 0.6508 | 0.5783 | 0.5343 | 0.5884 | 0.5882 | 0.6148 |
| | Rank | 1 | 2 | 3 | 7 | 4 | 9 | 10 | 8 | 5 | 6 |

Journal of Theoretical and Applied Information Technology
31st July 2024. Vol.102. No. 14

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

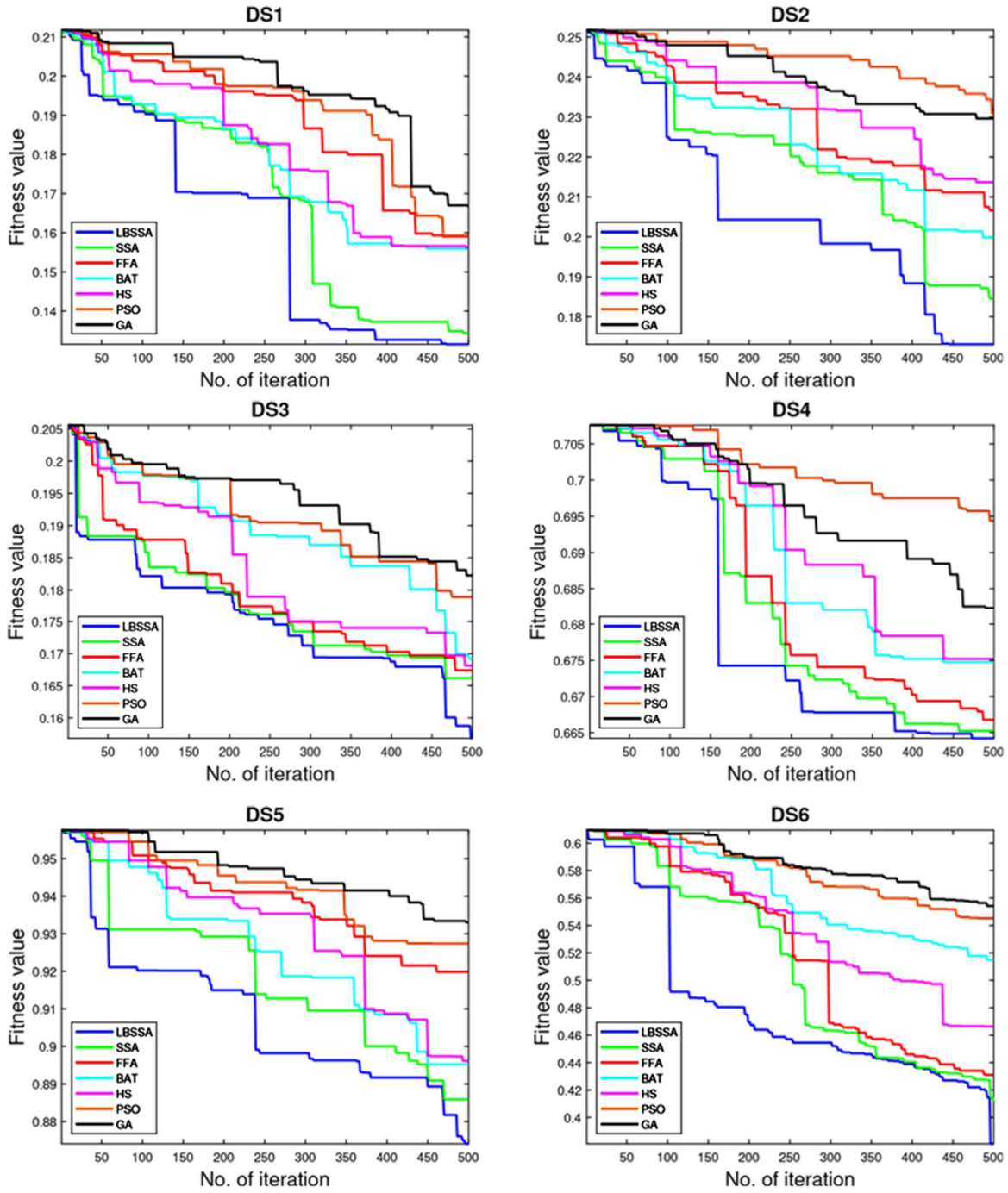


*Figure 2. Optimization Algorithms for TDC Convergence Rate Across Six Diverse Datasets.*