APPLIED ARTIFICIAL INTELLIGENCE
2024, VOL. 38, NO. 1, e2335098 (27 pages)
https://doi.org/10.1080/08839514.2024.2335098

An Adapted Ant Colony Optimization for Feature Selection

Duygu Yilmaz Eroglu[a] and Umut Akcan[b]

[a]Department of Industrial Engineering, Bursa Uludag University, Bursa, Turkey; [b]Graduate School of Natural and Applied Sciences, Industrial Engineering Program, Bursa Uludag University, Bursa, Turkey

ABSTRACT
As information technologies evolve, they generate vast and ever-expanding datasets. This wealth of high-dimensional data presents challenges, including increased computational demands and difficulties in extracting valuable insights. The aim of feature selection is to address this complexity by reducing data dimensions with minimal information loss. Our proposed feature selection approach, the Feature Selection via Ant Colony Optimization algorithm, employs heuristic distance directly in its probability function, instead of using its inverse. The algorithm bypasses the need for sub-attribute sets, running multiple iterations to create a frequency order list from the collected routes, which informs feature importance. The efficacy of this technique has been validated through comparative experiments with other methods from scientific literature. To ensure fairness, these experiments used identical datasets, data partitioning strategies, classifiers, and performance metrics. Initially, the algorithm was compared with fifteen different algorithms, and subsequently benchmarked against three selected methods. The impact of feature selection on classification performance was statistically verified through comparisons before and after the feature selection process. Convergence performance of the proposed method has also been evaluated. Our findings robustly support the efficacy of the introduced approach in managing complex, multidimensional data effectively.

ARTICLE HISTORY
Received 25 August 2023
Revised 13 February 2024
Accepted 9 March 2024

Introduction

As data volumes rapidly expand, innovative techniques stemming from machine learning, statistics, and data mining are crucial for processing and analyzing this burgeoning big data. High-dimensional data, while rich in complex and varied information, correspondingly escalates computational, processing, and storage demands across diverse disciplines. Challenges abound, from identifying key features and detecting anomalies to other drawbacks associated with high-dimensional data structures. Despite these hurdles, more sophisticated methods are being devised that efficiently address these problems, taking the entirety of dimensional space into

CONTACT Duygu Yilmaz Eroglu duygueroglu@uludag.edu.tr Department of Industrial Engineering, Bursa Uludag University, Gorukle Campus, Bursa 16059, Turkey

account. Data preprocessing represents a critical, albeit time-intensive, phase in the data mining process. Yet, for enhanced performance, researchers must thoroughly comprehend and pinpoint not just issues related to the data but also understand how these issues intertwine with knowledge-based methodologies (Zhang, Zhang, and Yang 2003). Feature selection stands as a prevalent preprocessing technique in machine learning, which involves selecting the most pertinent features for model development, a step that becomes particularly crucial when navigating a space with a high number of dimensions (Eroglu and Kilic 2017). Feature selection is like trying to solve a tricky puzzle because finding the best mix of features isn't straightforward and can be quite time-consuming. Traditional methods tackle this problem by scoring each feature based on how the data looks or behaves, ranking them, and then picking them out one at a time. But these methods can miss out on how different features affect each other, which means they can't promise to always find the best possible mix (Yan and Yang 2015). To handle the issue of these methods taking too long, certain algorithms that aim for a good-enough set of features more quickly have been developed – these are known as approximation algorithms, and there are four types: filter, wrapper, embedded, and hybrid (Rostami et al. 2021). Filter methods use simple data checks, which are quick and don't require much computing power, potentially saving time and cost. Wrapper methods, however, use a trial-and-error process with a learning algorithm that learns as it goes, which can get more accurate results but at the cost of using more computing resources. Embedded methods incorporate the feature selection right into the model-building process (Palma-Mendoza et al. 2019). Hybrids mix the quick and simple filters with the more thorough wrapper methods, trying to get the best mix of speed and accuracy. They start by quickly cutting down the list of features using the filter method. In the second phase, the selection of the best subset of features from the already narrowed-down list is completed using the wrapper method. This two-step process in hybrid approaches helps mitigate the risk of accidentally discarding valuable features, an issue that may arise if only a filter method is employed (Unler, Murat, and Babu Chinnam 2011). In these hybrid models, metaheuristic algorithms often perform the initial filtering. Metaheuristics stand out as a particularly effective strategy for identifying the optimal set of features quickly, and there is extensive research on this topic. Recent comprehensive reviews have been conducted on metaheuristic algorithms for selecting features across multiple classes (Akinola et al. 2022) and on a decade's worth of research in this field (Agrawal et al. 2021). Swarm intelligence (SI) algorithms, inspired by the collective behaviors of animals such as ants, bees, and birds, fall under the umbrella of metaheuristics and are known for their effective handling of feature selection. These algorithms harness the concept of group intelligence to develop a self-organizing system focused on achieving goals, like finding food sources (Bindu and Sabu 2020). One prominent SI algorithm, the Ant Colony Optimization (ACO), is applied to a variety of challenges, ranging from detecting irregular driving patterns (Huang et al. 2023) to applications in mechanical design, manufacturing, and automation (Qiu and Huang 2023). Its effectiveness in feature selection has also been documented and validated multiple times in previous studies (Kashef and Nezamabadi-Pour 2015; Wang et al. 2022). Nevertheless, it has been shown that making innovative changes, even minor ones, to the fundamentals of these algorithms can lead to improved outcomes, as has been repeatedly proven in research.

In the study outlined, a novel hybrid approach for feature selection is proposed, termed Feature Selection via Ant Colony Optimization (FSvACO). The proposed method deviates from standard research in two main ways when applying the ant colony algorithm. First, it skips the commonly used "inverse multiplicative property" for generating heuristic distance in the algorithm's probability function. Second, instead of adhering to the conventional view that paths are attribute subsets or selecting the most pheromone-laden paths, this approach runs the algorithm multiple times to create a frequency-based ranking of paths.

The developed algorithm has undergone validation through a series of four distinct evaluations:

(1) The initial comparison involves the algorithm's performance relative to fifteen documented methods in scientific literature, using four common datasets. At this stage, all methods apply k-fold cross-validation, employ the SVM classifier, and utilize the accuracy metric for comparison.
(2) In the second comparison, the algorithm is benchmarked against three other algorithms over eight common datasets. Here, a two-thirds training to one-third test data partitioning method is the standard approach. The algorithms run five times using an SVM classifier, and the resulting accuracy values are compared.
(3) The third comparison calculates the f-score values for two classifiers (KNN and SVM) with and without the implementation of feature selection. Both the original and the preprocessed datasets – the latter refined by FSvACO to select the most relevant features – use a two-thirds training and one-third test set split for evaluation.
(4) Lastly, studies regarding convergence performance have been conducted.

The validation of our algorithm through extensive comparisons emphasizes its substantial contribution to the domain of feature selection techniques within high-dimensional data analysis.

Literature Review on Ant Colony Optimization for Feature Selection

Solorio-Fernández, Ariel Carrasco-Ochoa, and Fco Martínez-Trinidad (2020), provides a comprehensive review of unsupervised feature selection methods, including the most relevant and recent developments. All the compared algorithms are also compared with our study, considering the common data-sets. The brief information about these algorithms in the literature are as follows; The SVD-Entropy method, proposed in Varshavsky et al. (2006), uses information-based univariate feature selection to select features with the best data representation. This basis measures the entropy of the original data matrix through its singular values. When entropy is low, distinct clusters form since the data matrix spectrum isn't evenly distributed. High entropy shows a uniformly distributed spectrum, leading to indistinct clusters. Another method is Laplacian Score (LS) proposed by He, Cai, and Niyogi (2005). The proposed filter method for feature selection is independent of any learning algorithm. It is based on the observation that data points from the same class are often close to each other. The importance of a feature is evaluated by its power of locality preserving, or Laplacian score. Another method, Spectral feature selection (SPEC) is developed by Zhao and Liu (2007). The proposed algorithm works by using spectral graph theory to determine the relevance of a feature based on its consistency with the structure of the graph induced from the similarity set. The proposed method is able to generate families of algorithms for both supervised and unsupervised feature selection. Another algorithm, Unsupervised Spectral Feature Selection Method for mixed data (US-FSM) is developed by Solorio-Fernández, Fco Martínez-Trinidad, and Ariel Carrasco-Ochoa (2017). US-FSM evaluates features by studying the alterations in the spectral distribution (known as spectral gaps) of the first significant eigenvalues of the Normalized Laplacian matrix when each feature is individually removed from the full feature set. The features are then arranged in descending order based on their respective spectral gap values. Another methodology, Feature Selection using Feature Similarity (FSFS), introduced by Mitra, Murthy, and Sankar (2002). This approach uses a statistical measure called maximal information compression index to eliminate feature redundancy. The method involves partitioning features into clusters so that similar features are grouped together, and dissimilar features are separated. RRFS (Relevance Redundancy Feature Selection) is an algorithm, proposed by Ferreira and Figueiredo (2012), used for feature selection that removes redundant features to obtain a more adequate subset of features for learning problems. It is used to reduce the number of features, thus directly targeting the curse of dimensionality problem, often to obtain better performing classifiers. Another study, proposed by Yang et al. (2011), presents an unsupervised feature selection algorithm, UDFS (Unsupervised Discriminative Feature Selection algorithm) which is able to select discriminative features in batch mode. The algorithm is validated by comparing with the existing unsupervised feature selection algorithms. The other proposed methodology, NDFS (Nonnegative Discriminative Feature Selection), proposed by Li et al. (2012) is a general approach for spectral analysis-based feature selection. It seeks to select discriminative features for unsupervised learning by utilizing the cluster labels based on the data structure. It learns the scaled cluster indicator matrix F and the feature selection matrix W simultaneously. Tabakhi, Moradi, and Akhlaghian (2014) proposed unsupervised feature selection method based on ant colony optimization (UFSACO). It uses the inverse of the similarity between features as the heuristic information and a desirability measure, pheromone, which is associated with the features and is updated by ants gradually. The performance of the method is evaluated over different classifiers using the classification error rate as the performance measure. MGSACO (Multi-objective Gene Selection Algorithm based on Ant Colony Optimization) (Tabakhi et al. 2015) is a filter-based gene selection method that uses the relevance and redundancy analyses in the gene selection process. It uses the search strategy of the ant colony optimization algorithm to solve the gene selection problem, and the inverse of the similarity between genes is used as the heuristic information which is assigned to the graph edges. The fitness value of the solution is computed by taking the relevance of each gene in the subset. Another method, Dual Self-Representation and Manifold Regularization (DSRMR) proposed by Tang et al. (2018), takes the local geometrical structure of the data into account and uses an $l_{2,1}$-norm and a sample self-representation term to learn the similarity graph automatically. LLC-fs (Locality Constrained Linear Coding with Feature Selection) proposed by Zeng and Cheung (2010) is an approach for feature selection and kernel learning for local learning-based clustering. It uses a regularization framework to achieve sparse feature weights or kernel combination coefficients, which helps to better understand the problem by focusing on only a few dominant features or kernels that most contribute to the task. Guo and Zhu (2018) developed Dependence Guided Unsupervised Feature Selection (DGUFS). The algorithm is designed to select features and partition data in a joint manner. It uses $l_{2,0}$-norm equality constraints and two dependence guided terms to enhance the interdependence between original data, cluster labels, and selected features. Another proposed methodology (Li, Lu, and Wu 2006) is a hybrid approach based on ranking features according to their relevance to clustering using a new ranking index which belongs to exponential entropy. The hybrid method uses a modified Fuzzy Feature Evaluation Index with a new method to calculate the feature weight and a wrapper method to select a compact feature subset from the candidate feature set based on the clustering performance. LS-WNCH-BE (Laplacian Score Weighted Normalized Calinski – Harabasz Backward Elimination) is proposed by Solorio-Fernández, Ariel Carrasco-Ochoa, and Fco Martínez-

Trinidad (2016). The developed hybrid filter-wrapper method for unsupervised feature selection combines spectral feature selection using the Laplacian Score ranking and a modified Calinski-Harabasz index. The method has a reasonable compromise between quality and performance and can select relevant features while eliminating those that do not contribute to improving the quality of the clusters.

Key studies in the scholarly literature that utilize the ant colony algorithm for feature selection include the following. Al-Ani (2007) applied a mutual information measure to compute heuristic distance and updated pheromones with a formula that included mean square error. Sivagaminathan and Ramakrishnan (2007) introduced a process that combines ACO with artificial neural networks to conduct feature selection; this method updates pheromone levels based on feedback from the neural networks. Kanan and Faez (2008) employed this approach within a facial recognition system. In the research of Deriche (2009), the impact of various local feature selection criteria through ACO was analyzed, with a specific focus on comparing the Fisher criterion to mutual information measures in the context of mean square error. Aghdam, Ghasem-Aghaee, and Ehsan Basiri (2009) leveraged ant colony optimization (ACO) for feature selection in text mining to address issues with high-dimensional data. Min and Fangfang (2010) developed a dual-method approach that utilized filter and wrapper techniques. They employed several statistical methods, including correlative feature selection, relief, Mahalanobis distance, multivariate correlation, and mutual information for filtering. For the wrapping phase, they tested adaptive genetic algorithms, chaotic binary particle swarm optimization, and a clonal selection algorithm, ultimately favoring the latter due to its superior performance in global optimization. Ali and Shahzad (2012) employed a measure known as symmetric uncertainty, along with entropy, to ascertain the heuristic distance among features. Wald, Khoshgoftaar, and Napolitano (2013) investigated whether using the same learning model both inside and outside the wrapper method affected the feature selection process. Saraç and Ayşe Özel (2014) applied ACO to select features for web page classification, utilizing the training set's document frequency to guide heuristic distance. Hamed, Dara, and Kremer (2014) introduced an embedded method utilizing support vector machines for selecting features. Kashef and Nezamabadi-Pour (2015) suggested a feature selection technique combining the ant colony algorithm with a binary selection method, which distinguishes whether a feature is present or absent, with ants selecting nodes as they explore the features, guided by various statistical assessments for the binary paths. Mohammed et al. (2016) developed an ACO-based feature selection approach aimed at simplifying the classification of endoscopy images into bleeding, non-bleeding, and non-informative categories. To penalize misclassification heavily in their evaluation function, they incorporated sensitivity and accuracy rates into a new assessment formula. Fahrudin, Syarif, and Ridho Barakbah (2016) introduced an ant colony optimization algorithm tailored for feature selection in microarray analysis. Microarrays, which are instrumental in detecting nucleic acid sequences within cells, facilitate the simultaneous analysis of numerous samples. During the selection process, the team employed three distinct algorithms: a genetic algorithm, particle swarm optimization, and ant colony optimization, suggesting that the latter may be particularly effective for high-dimensional datasets. Shunmugapriya and Kanmani (2017) designed a hybrid algorithm that amalgamates elements of both ant colony and artificial bee colony algorithms. In their model, the bee components utilize attribute subsets identified by the ant mechanisms. Ghosh et al. (2020) devised a combined wrapper-filter feature selection strategy based on ant colony algorithms, utilizing cosine similarity for ant colony operations. They evaluated feature subsets through a function that calculates accuracy and the quantity of unused features. Meanwhile, Paniri, Bagher Dowlatshahi, and Nezamabadi-Pour (2020) developed a feature selection technique using ant colony optimization for multi-label classification challenges. Their approach utilized the Pearson correlation coefficient and cosine similarity while accounting for the associations between feature sets and class labels in multi-label contexts.

Preliminaries

Ant Colony Algorithm (ACO)

Introduced by Dorigo (1992), ant colony optimization is an algorithm inspired by the foraging behavior observed in ants. These insects form colonies and communicate indirectly by laying down pheromones, which serve as trails leading to food sources for other ants. When an ant traverses a route to food, it leaves behind a pheromone trail; higher concentrations of this chemical attract more ants, optimizing the food-gathering process (Deneubourg et al. 1990). The ACO algorithm adopts this natural strategy, utilizing a form of simulated pheromone to mark promising paths in the search space for various optimization problems. As the algorithm iterates, pheromone levels accumulate along better solutions, guiding subsequent search efforts (Dorigo, Birattari, and Stutzle 2006; Garnier, Gautrais, and Theraulaz 2007).

Ant Colony Algorithm Parameters

- Number of Ants: Specifies how many artificial ants are used.
- Number of Iterations: Number of algorithm cycles. Can also be considered as the number of colonies.
- Alpha(α): Determines the weight given to the pheromone trail.
- Beta(β): Governs the significance of the heuristic distance ratio.
- Evaporation Coefficient(ρ): Sets the evaporation rate of pheromones.

Ant Colony Algorithm Selection Formula

The equation (1) indicates the probability pattern for path selection, with τ_{ij}, representing the pheromone amount on the i-j path and η_{ij} the heuristic value associated with that path, typically the inverse of the distance (d_{ij}).

$$P_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{u\in J_k}\tau_{iu}^{\alpha}\eta_{iu}^{\beta}} if\ j \in J_k \\ 0 \qquad otherwise \end{cases} \tag{1}$$

- τ_{ij}: The amount of pheromone between i and j.
- η_{ij}: Intuitive distance between i and j. ($1/d_{ij}$, d_{ij}: distance between i and j)
- J_k: The set of neighborhoods to go to.

The pheromone levels are updated using the following relations:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2}$$

where each $\Delta\tau_{ij}^{k}$, as defined by:

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{1}{L_k}\ if\, ant\ k\ \ used\ the\ i-j\ route\ on\ this\ route \\ 0 \qquad\qquad\qquad otherwise \end{cases} \tag{3}$$

- L_k: The route length used by the ant k.

According to these dynamics, the ants choose paths based on the selection criteria, with pheromone intensities being updated after the completion of each tour.

Ant Colony Algorithm Pseudo-Code

Input: Parameters related to ant colony optimization.
Output: The best path identified by maximum pheromone deposition.

```
• Initialize placement of ants at random positions.
• Repeat the following until termination conditions are met:
    ○ Compute paths for each ant.
    ○ Intensify pheromone trail on the most promising route.
    ○ Apply evaporation to pheromone trails on all paths.
• End the iterative process upon reaching stop conditions.
```

Classification Techniques in the Study

K Nearest Neighbor Algorithm (KNN)

The K nearest neighbor (KNN) algorithm, initially introduced by Fix and Hodges (1989) and further developed by Cover and Hart (1967), is a type of supervised machine learning approach. As a supervised learning method, it operates with well-defined class labels and applies the learned patterns to new data. The KNN algorithm is versatile, suitable for both classification and regression tasks, leveraging the patterns discovered in the training data. A majority vote among the k closest neighbors determines the classification of test data, a process that circumvents issues associated with unbalanced data distributions as indicated by Malini Devi, Seetha, and Sunitha (2016).

Working Mechanics of the KNN Algorithm

During classification, the process involves selecting a predefined number of neighbors, "K." The algorithm then calculates the distance between the non-labeled sample and all other samples in the data set.

- It repeats the procedure for each data point in the dataset.
- Once distances are computed, they are organized in ascending order to identify the nearest "K" samples.
- A sample is tagged as a true positive if it belongs to a specific class, and as a true negative if it does not.
- Accuracy is calculated based on the count of true positives and true negatives.
- KNN is often dubbed a "lazy learner" because it doesn't learn from the training set immediately. It stores the data and only processes it during the classification stage.

Key Parameters of the KNN Algorithm

- Number of Neighbors (K): This critical parameter defines the number of closest neighbors to include in the voting process.
- Distance Metric: This specifies the method to compute the proximity between points. The KNN algorithm employs various distance metrics such as:
  - Minkowski Distance: Applicable for real-numbered vectors, it demands non-negative elements and calculates distance using the formula given as equation (4), where "x" and "y" denote Cartesian coordinates.

$$d = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

  - Manhattan Distance: A specific instance of Minkowski distance with $p = 1$, it sums the absolute differences of Cartesian coordinates, defined by equation (5).

$$d = \sum_{i=1}^{n} |x_i - y_i|$$

- Euclidean Distance: Another special case of Minkowski distance with $p = 2$, it measures the straight-line distance between points and is represented as equation (6).

$$d = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Support Vector Machines (SVM)

Support Vector Machines (SVM) (Boser, Guyon, and Vapnik 1992) are esteemed within the realm of classification algorithms for their proficiency in pinpointing the optimal decision boundary or hyperplane, effectively distinguishing between different categories of data. Their hallmark lies in optimizing the margin's breadth between distinct classes to enhance separation efficiency.

Linear SVMs typically utilize a hyperplane, which is linear in nature, to separate data points in a binary classification setting. However, when dealing with multiclass datasets, multiple such hyperplanes are strategically employed either through "One-vs-One" or "One-vs-the-Rest" approaches, facilitating the division of the dataset into its respective multiple categories. Figure 1 illustrates the decision boundary created by the linearly separable SVM. This boundary line separates the different classes and aims to keep the distance between them at the maximum distance (Yang, Javed Awan, and Vall-Llosera 2019).

Pertinent to the linear SVM formulation, several entities are introduced:

- **w** stands for the hyperplane's weight vector.
- **x**: represents the vector of input features.
- The term b denotes the bias, a scalar acting as the hyperplane equation's offset component.
- The term //**w**// signifies the Euclidean length of the vector **w**.
- Equation (7) formally defines the decision boundary for the SVM's decision-making plane.

$$\boldsymbol{wx} + b = 0 \tag{7}$$

Within this linear framework, $\boldsymbol{w}$ is oriented orthogonally to the partitioning hyperplane, whereas b facilitates the hyperplane's placement within the feature domain.

- The SVM's ultimate objective is crystallized in Equation (8), elucidating the margin's maximization:

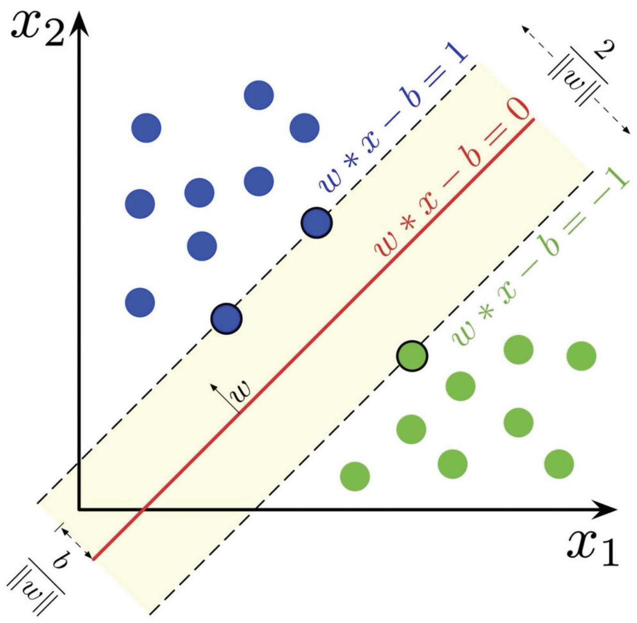

Figure 1. Hyperplane and margins for an SVM for samples with two classes.

$$margin = \frac{2}{\| \mathbf{w} \|} \tag{8}$$

- The function that adjudicates the class labels, known as the decision function and articulated in Equation (9), ascribes labels contingent on the positioning of the data point relative to the hyperplane:

$$f(x_i) = \{ \begin{matrix} 1, if\ \boldsymbol{wx_i} + b \geq 1 \\ -1, if\ \boldsymbol{wx_i} + b \leq -1 \end{matrix} \tag{9}$$

In essence, linear SVM is geared toward the calibration of the optimal hyperplane equation ($\boldsymbol{wx} + b = 0$) to segregate the classes with the utmost margin. It's an exercise in fine-tuning $\boldsymbol{w}$ and b to satisfy the class labeling constraints outlined in Equation (9).

Classification Metrics

In this study, accuracy rate, and f-score metrics are considered for performance measurement. These metrics can be calculated as follows (Naser and Amir 2023)

$$Accuracy\,Rate = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

In (10), the meanings of abbreviations are as follows for binary (e.g. 0/1) class labels;

- True Positive (TP): Examples where the true value is 1 and the predicted value is 1.
- True Negatives (TN): Examples, where the true value is 0 and the predicted value, is 0.
- False Positives (FP): Examples where the true value is 0 but the predicted value is 1.
- False Negatives (FN): Examples where the true value is 1 but the predicted value is 0.
- The f-score is the harmonic mean of the ratio of Positive Predictive Value (PPV) and True Positive Rate (TPR) of the classification result. It is a measure of how well the classifier is performing and is often used to compare classifiers. The calculation method is shown in (11).

$$f - score = \frac{\text{2x PPV xTPR}}{\text{PPV} + \text{TPR}} \tag{11}$$

In (11), the meanings of abbreviations are as follows; PPV means the proportions of positive observations that are true positives and can be calculated as in (12). TPR, which might be calculated as in (13), measures the proportion of actual positives that are correctly identified as positives.

$$PPV = \frac{TP}{TP + FP} \tag{12}$$

$$TPR = \frac{TP}{TP + FN} \tag{13}$$

Datasets

This section provides a description of the datasets utilized in the research, with Table 1 displaying details about each dataset. The data used are open source, acquired from the UCI Machine Learning Repository (Dua and Graff 2017).

The Wine dataset consists of results from a chemical analysis of various wines, categorized into three distinct types. The aim is to predict the class of a wine based on its analysis. The hepatitis dataset, on the other hand, divides instances into two categories: those that result in death and those where the patient survives. The WDBC dataset is derived from images obtained through fine needle aspirate procedures on breast masses, describing the attributes of the cell nuclei present in these images. This dataset categorizes findings into malignant or benign classes. Data from the Ionosphere dataset was collected by a radar system in Goose Bay,

Table 1. Dataset information.

Dataset	# of Feature	# of Class	# of Instance
Wine	13	3	178
Hepatitis	19	2	155
WDBC	30	2	569
Ionosphere	34	2	351
Dermatology	34	6	366
Spambase	57	2	4601
Arrhythmia	279	16	452
Madellon	500	2	4400

Labrador, and include two classifications: "good" signals, which bounce off the ionosphere, and "bad" signals, which pass through it. The Dermatology dataset aims to diagnose types of Erythemato-Squamous Diseases and includes six different classes, such as psoriasis and seborrheic dermatitis, among others. The Spambase dataset is a binary class data collection that distinguishes between spam and non-spam e-mails. The Arrhythmia dataset is designed to detect the presence or absence of various types of cardiac arrhythmias and sorts them into one of 16 categories. Lastly, the Madelon dataset is a synthetic, two-class dataset specifically generated for experimenting with binary classification tasks.

The selection of datasets has been guided by various criteria, such as the number of features, dimensions, and instances present. Datasets of diverse sizes, ranging from small to medium to large, have been intentionally selected to evaluate the performance of the proposed method across various scales in terms of feature quantity, data volume, and class variety.

Proposed FsvACO Method

The objective of the suggested approach is to enhance the indicators of classification performance through feature selection. In the ant colony optimization algorithm, pheromones are deposited on the paths that constitute the optimal route. To function effectively, the algorithm requires estimates of heuristic distances between features. These distances are determined using the cosine similarity method, which is calculated as the dot product of two vectors normalized by the product of their magnitudes. The formula for cosine similarity is presented in equation (14), where A and B symbolize the respective vectors.

$$Similarity(A,B) = \frac{AxB}{\parallel A \parallel x \parallel B \parallel} = \frac{\sum_{=1}^{n} A_i x B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} x \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{14}$$

Previously mentioned in the context of the ant colony algorithm, the distance between two points, denoted as d_{ij}, is determined using the cosine similarity, as outlined in the formula presented in equation (14) for the adaptation to feature selection.

The Main Steps of the Proposed Algorithm Are As Follows

```
(1) Initialize algorithm parameters: Alpha(α): 0.4, Beta(β): 0.6, Evaporation
    Coefficient(ρ): 0.9, Initial Pheromone Quantity: 10, Number of Ants:10,
    Max Iterations: 5, Ant Run Quantity:10, Frequency of selection:0.8.
(2) Define functions:
    CF: to initialize pheromone levels:
        Create a pheromone matrix with all features set to "Initial
        Pheromone Quantity" for each ant.
        Set diagonal elements to 0 (no pheromone for self-loop)
    PROB: to calculate transition probabilities:
        Calculate the probability of transition from one feature
        to another based on pheromone levels and similarity of
        features.
    CALC: to calculate the total distance of a given route:
        Sum the distances associated with transitioning between all
        pairs of subsequent features in the route.
    ROUTE: to construct a route for an ant:
        Start from a given feature.
            Select subsequent features based on the calculated
            probabilities (PROB) until all features are visited.
        Return the complete route.
    ANT_TOUR: to simulate one tour of all ants:
        For each starting feature in starting point
            Find a route with ROUTE
            Calculate the distance of the route with CALC
            If the route is better than the best known, update best
              distance and route of each ant.
    UPDATE: to update pheromone levels:
        Evaporate some of the pheromone on all paths.
        Add new pheromone to the paths included in the best route.
(3) Run Algorithm
    RUN: to run the algorithm for "Max Iterations" times:
      Run the algorithm for "Ant Run Quantity" times, in each:
          Reset pheromone levels with CF
          Simulate a number of ant tours with ANT_TOUR
          Update pheromone levels with UPDATE
      Create a list of frequency considering the Frequency of selection
        parameter (0.8=%80) of the best route per iteration
      Sort features by their selection frequency in descending order.
```

The **RUN** function operates for 5 iterations, and during each iteration, each defined ant (Number of Ants: 10) runs 10 times (Ant Run Quantity: 10). In every step, all the ants will execute the **ANT_TOUR** function, meaning that the total number of ant tours throughout the entire **RUN** function will be 5 × 10 * 10, which equals 500. Therefore, there will be 500 instances of ant tour simulations during the complete run of the algorithm.

The frequency list can be explained using an example with 5 iterations, as shown in Table 2. In this example, let's consider 10 features. For each iteration, initially, each of the 10 ants randomly settles on a feature, then moves on to the next feature based on the probabilities obtained by considering the pheromone levels on the route and the similarities. All 10 ants complete the tour over all attributes 10 times (So totally 100 runs per iteration). The best result obtained by the best ant is written for Iteration 1 column as seen in Table 2(a). Similar processes are repeated for 5 iterations. Afterward, 8 of the 10 attributes obtained in each iteration are considered, which is 80% of each iteration. These are indicated in bold in the Table 2(a). The number of times all features are encountered is counted (by counting the features which are highlighted in bold in Table 2(a)). This could be called the frequency list that is shown in Table 2(b). When the frequency list is sorted in non-increasing order, the most important features will appear at the top. As for the classification algorithms, each feature is added sequentially according to their importance (which might be seen in frequency list) and run. For example, first the 1st feature is added, then both the 1st and 3rd attributes are added, and so on. In data sets, after a certain number of attributes are added, there usually begins a decline in performance metrics. For the proposed algorithm, this represents the best number of attributes obtained with the relevant classifier for that data set.

The algorithm consists of two stages. In the first stage, frequency sequences are created via the proposed FSvACO. In the second stage, classification algorithms are used.

This article considers the following assumptions: The parameters used in the ant colony algorithm and the proposed algorithm, such as Alpha(α), Beta (β), Evaporation Coefficient(ρ), Initial Pheromone Quantity, Number of Ants, Max Iterations, Ant Run Quantity, Frequency of Selection, can modify the selected features. The type of classifier used after feature selection also affects

Table 2. An example to explain frequency list.

Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5		Features	Frequency
5	**2**	**3**	**4**	**4**		1	5
1	**4**	**2**	**3**	**5**		3	5
4	**1**	**1**	**7**	**6**		8	5
3	**3**	**4**	**8**	**1**		4	5
6	**5**	**5**	**9**	**2**	⇨	2	4
8	**6**	**7**	**10**	**7**		5	4
9	**8**	**8**	**1**	**8**		7	4
10	**7**	**9**	**2**	**3**		6	3
2	9	10	5	9		9	3
7	10	6	6	10		10	2

(a) The best ant tour for each iteration (b) Frequency list the performance metrics obtained. SVM and KNN classifiers are used in this study. In comparisons made with literature results, the accuracy value is used as the performance indicator. Additionally, in the comparison of the original data sets with the selected features obtained using the proposed algorithm, the f-score value is chosen. This is because f-score can measure the accuracy of predictions in the negative and positive classes in a balanced way.

The approach outlined here diverges from existing literature in two key aspects related to the implementation of the ant colony algorithm. Firstly, while heuristic distance is commonly derived in the literature through the "inverse property of multiplication" in the algorithm's probability function, this step has not been employed in the current method. Second, traditional interpretations of information within the ant colony algorithm suggest that the paths generate subsets of attributes, or that the most pheromone-rich paths are utilized. In contrast, the method proposed here involves multiple runs of the ant colony algorithm, from which a frequency ranking is established based on the paths obtained.

Computational Results

This section provides a summary of the study's findings. The coding is conducted using the Python programming language and is divided into two subsections: experimental design followed by comparison and discussion. The initial subsection details the methodology behind the selection of parameters for the proposed algorithm, while the second subsection evaluates the algorithm's validation.

Experimental Design

For the Ant Colony Optimization (ACO) to run efficiently, it's crucial to select the right parameter values. This study conducts parameter tuning using the WDBC dataset, which is split into two portions: a training set comprising two-thirds of the data and a test set consisting of the remaining one-third. The KNN algorithm, set with K equal to 5, is the chosen classification method, and performance is gauged based on the accuracy rate. The classification algorithm is run five times, and the average accuracy is taken as the final value. Referring to Figure 2, the parameters yielding the highest accuracy rates have been identified as 0.4 for alpha, 0.6 for beta, and 0.9 for the evaporation rate; hence, these values have been adopted for the algorithm. It's noted that as the ant run quantity of the ant colony algorithm increases, so does the computational cost, which grows exponentially. To evaluate performance with different ant run quantities, the study conducts tests with both 100 and 1000 ant runs on the WDBC dataset. With 100 ant runs and 14 features, the algorithm attains an accuracy of 0.95,

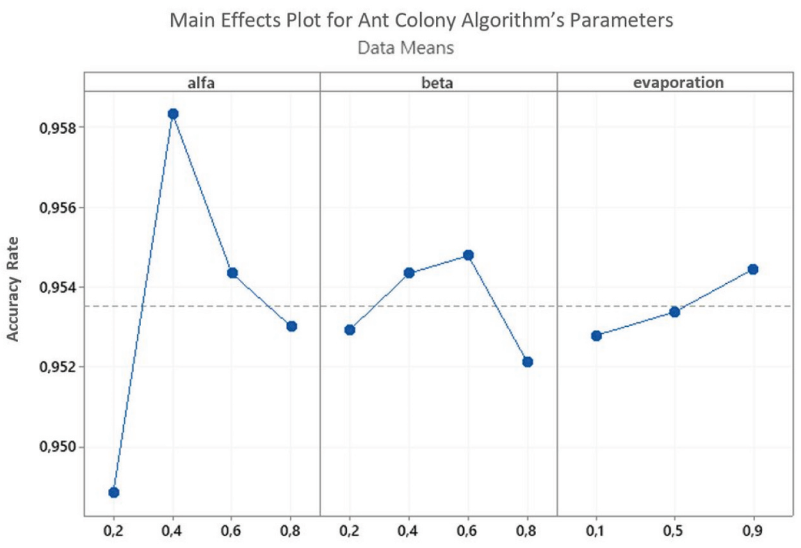

Figure 2. Ant colony algorithm parameters' main effect plot.

Table 3. Selected parameter values of the ant colony algorithm.

Parameter	Value
Alpha (α)	0.4
Beta (β)	0.6
Evaporation Coefficient (ρ)	0.9
Number of Ants	10
Initial Pheromone Quantity	10
Ant Run Quantity	10

representing a 2% enhancement compared to just 10 ant runs. Likewise, executing 1000 ant runs with 5 features also yields a 0.95 accuracy rate, denoting a 2% uptick from 10 ant runs while requiring fewer features than at 100 ant runs. Nevertheless, given the prolonged execution times experienced with larger datasets, the study prefers 10 ant runs, guaranteeing CPU times of less than one second. The selected parametric settings for the ant colony algorithm are explicated in Table 3.

Utilizing the ant colony algorithm with the parameter settings provided in Table 3 results in the creation of feature frequency sequences for every dataset involved.

Comparison and Discussion

This section will be examined in three subsections. The first two subsections compare the results obtained with the proposed method with those obtained from the feature selection studies in the literature. The reason they are shown in different subsections is due to the different accuracy values obtained with different data partitioning techniques ("k-fold cross-validation," "2/3 training, 1/3 test split"). The third subsection demonstrates the algorithm's feature selection performance with different classification techniques and different performance indicators, specifically the f-score. And finally, the convergence of the proposed algorithm has been detailed in the analysis on some datasets.

Comparing Methods: Using K-Fold Cross Validation

The data in the Table 4, excluding, proposed method FSvACO, were obtained from review study conducted by Solorio-Fernández, Ariel Carrasco-Ochoa, and Fco Martínez-Trinidad (2020). Proposed FSvACO's results compared with 15 algorithm's results. The SVD-Entropy method, proposed in Varshavsky et al. (2006), Laplacian Score (LS) proposed by He, Cai, and Niyogi (2005), Spectral feature selection (SPEC) by Zhao and Liu (2007), Unsupervised Spectral Feature Selection Method (USFSM) developed by Solorio-Fernández, Fco Martínez-Trinidad, and Ariel Carrasco-Ochoa (2017), Feature Selection using Feature Similarity (FSFS) by Mitra, Murthy, and Sankar (2002), RRFS (Relevance Redundancy Feature Selection) proposed by Ferreira and Figueiredo (2012), UDFS (Unsupervised Discriminative Feature Selection algorithm) by Yang et al. (2011), NDFS (Nonnegative Discriminative Feature Selection), proposed by Li et al. (2012), UFSACO (Unsupervised Feature Selection method based on Ant Colony Optimization) proposed by Tabakhi, Moradi, and Akhlaghian (2014), MGSACO (Multi-objective Gene Selection Algorithm based on Ant Colony Optimization) by Tabakhi et al. (2015), DSRMR (Dual Self-Representation and Manifold Regularization) proposed by Tang et al. (2018), LLC-fs (Locality Constrained Linear Coding with Feature Selection) proposed by Zeng and Cheung (2010), DGUFS (Dependence Guided Unsupervised Feature Selection) proposed by Guo and Zhu (2018), the hybrid methodology by Li, Lu, and Wu (2006), which is called using citation name: Li, Lu, and Wu (2006) in

Table 4. Comparison of average accuracy rates using 5-fold cross validation of different methods using SVM classifier for four datasets.

METHOD	Hepatitis	Ionosphere	Wdbc	Wine
SVD-entropy	0.852	0.866	0.944	0.955
LS	0.852	0.832	0.952	0.955
SPEC	0.858	0.789	0.965	0.955
USFSM	0.832	0.889	0.952	0.949
FSFS	0.839	0.852	0.958	0.933
RRFS	0.826	0.866	0.963	0.921
UDFS	0.858	0.869	0.951	0.926
NDFS	**0.871**	0.880	0.967	0.966
UFSACO	0.839	0.889	0.965	0.938
MGSACO	0.839	0.866	0.961	0.944
DSRMR	0.845	0.886	0.956	0.938
LLC-fs	0.845	0.866	**0.975**	0.972
DGUFS	0.819	**0.889**	0.972	0.938
Li et al.	0.852	0.874	0.949	0.910
LS-WNCH-BE	0.845	0.852	0.935	0.961
FSvACO	0.857	**0.889**	0.960	**0.988**

the Table 4, and LS-WNCH-BE (Laplacian Score Weighted Normalized Calinski – Harabasz Backward Elimination) proposed by Solorio-Fernández, Carrasco-Ochoa, and Martínez-Trinidad (2016) are the compared algorithms. A comparison has been made for four commonly used datasets in the studies. Table 4 shows a comparison of average accuracy rates using 5-fold cross validation, using the SVM classifier. The best results are shown in boldface for each dataset in the table. The performance of the proposed algorithm, FSvACO, achieves the best result for two out of the four datasets.

Comparing Methods: Using 2/3 Training, 1/3 Test Set Division

In order to see its performance on more datasets, the performance of the proposed algorithm (FSvACO) has also been compared with the results of three feature selection algorithms in the literature. The comparisons consider the unsupervised feature selection ant colony algorithm (UFSACO) by Tabakhi, Moradi, and Akhlaghian (2014), the random subspace method (RSM) by Lai, Reinders, and Wessels (2006), and the mutual correlation method (MC) by Haindl et al. (2006), which allow evaluations over the 8 datasets commonly dealt with in this study. The average classification accuracy rates of 5 independent runs utilizing the SVM algorithm have been applied to assess the performance of FSvACO. The number of selected features is the same for the datasets compared, and in addition, a "2/3 training," "1/3 test" set division is adopted for the mentioned algorithms. Table 5 shows a comparison of average accuracy rates over 5 runs of FSvACO, UFSACO, RSM, and MC using the SVM classifier. The best result is shown in boldface for each dataset in the table. According to Table 5, the proposed FSvACO outperformed compared methods in four datasets (WDBC, Dermatology, Arrhythmia, and Spambase).

Feature Selection Performance of the Proposed FsvACO Utilizing KNN and SVM Classifiers (Using 2/3 Training, 1/3 Test Set Division, and F-Score As Performance Metric)

After conducting a comparative analysis with other algorithms in the literature, the performance of the proposed FSvACO algorithm in feature selection is demonstrated. As mentioned earlier, the selection process focuses on the first 80% of the

Table 5. Comparison of average accuracy rates over five runs of FSvACO, UFSACO, RCM, and MC using SVM classifier for different datasets.

Datasets	# of Features	FSvACO	UFSACO	RSM	MC
WDBC	5	**0.941**	0.907	0.838	0.890
Dermatology	25	**0.968**	0.953	0.949	0.946
Ionosphere	30	0.882	**0.886**	0.878	0.853
Arrhythmia	20	**0.597**	0.592	0.561	0.455
Wine	5	0.857	**0.951**	0.820	0.896
Hepatitis	5	0.788	**0.831**	0.809	0.827
Spambase	40	**0.998**	0.878	0.855	0.863
Madellon	70	0.486	**0.611**	0.535	0.515

best ant's route in each iteration. After running each iteration, the most frequently encountered feature in the final frequency list is identified as the most important, followed by the second most frequent feature, and so on. The classification algorithm initially operates with the top-ranking feature. Subsequently, it incorporates the top two features from the list. After adding a certain number of important features, the presence of less important features does not create significant changes in performance metrics. This threshold can be considered as the optimal feature count for the dataset and classifier. Studies have been conducted to provide insights into the algorithm's performance in this aspect. Studies are conducted by two well-known classifiers KNN (with K = 5), and SVM before and after selecting the most important features. The proposed algorithm is executed 5 times with different random seeds each time (i.e., replications), and "2/3 training," and "1/3 test" set splitting schema is adopted. Calculated average accuracy rates and f-score results are shown in Table 6 and 7 for SVM and KNN classifiers respectively.

Table 6 demonstrates the results of the SVM algorithm. As it can be shown, after selecting the most important features, better f-score values are obtained nearly for all the datasets. The Wilcoxon signed rank test is performed before and after feature selection, and the p-value is found as 0.015 for the f-score. Therefore, "feature selection by FSvACO" with SVM, outperforms "working with whole features" statistically in f-score with a 90% confidence interval in the analyses.

Table 6. Comparison of f-score values of SVM for different datasets before and after feature selection via FSvACO (the best result is shown in boldface for each dataset).

	Before Feature Selection		After Feature Selection	
Dataset	# of Features	f-score	# of Features	f-score
WDBC	30	0.955	25	**0.972**
Dermatology	34	0.960	19	**0.975**
Ionosphere	34	0.907	31	**0.927**
Arrhythmia	279	0.673	241	**0.690**
Wine	13	0.976	12	**0.980**
Hepatitis	19	0.874	12	**0.923**
Spambase	57	**0.999**	34	**0.999**
Madellon	500	0.543	461	**0.549**

Table 7. Comparison of f-score values of KNN (K = 5) for different datasets before and after feature selection via FSvACO (the best result is shown in boldface for each dataset).

	Before Feature Selection		After Feature Selection	
Dataset	# of Features	f-score	# of Features	f-score
WDBC	30	0.941	25	**0.947**
Dermatology	34	0.836	23	**0.953**
Ionosphere	34	0.891	4	**0.897**
Arrhythmia	279	0.630	233	**0.659**
Wine	13	0.681	8	**0.942**
Hepatitis	19	0.857	3	**0.887**
Spambase	57	**0.998**	34	0.996
Madellon	500	0.706	441	**0.721**

Table 7 presents the outcomes from the KNN algorithm's application. According to Table 7, after selecting the most important features, better f-score values are obtained for most of the datasets. The Wilcoxon signed rank test is performed before and after feature selection, and the p-value is found as 0.021 for the f-score. Therefore, "feature selection by FSvACO" with KNN, outperforms "working with whole features" statistically in f-score with a 90% confidence interval in the analyses.

The Convergence of the Proposed Algorithm

To analyze the convergence of the proposed algorithm, some additional analyses have been conducted. Table 8 displays the accuracy rates of the KNN with and without feature selection, with all results appearing to show improvement. This table exhibits the average outcomes obtained via 10-fold cross-validation after reaching the optimal number of features decided upon.

To pinpoint the optimal number of features, various algorithms are tested with differing numbers of features to track their convergence trajectories. To illustrate the algorithm's rate of convergence and its efficiency, four datasets are selected, with the corresponding graphical representations depicted in Figure 3. The graphs are generated by sequentially adding the features from the frequency list determined by the FSvACO algorithm. Here, a direct inference cannot be made between Table 8 and Figure 3. This is because while Table 8 shows the result obtained with the number of selected features using 10-fold cross-validation, Figure 3 depicts the result obtained with each feature selected from the frequency list added sequentially. The outcome can sometimes be better and sometimes worse than that presented in Table 8. The purpose of constructing Figure 3 is to observe how the algorithm performs with different numbers of features. As per the findings showcased in Figure 3, in the WDBC dataset, the feature increase from 3 to 4 correlates with a 30% enhancement in accuracy, from 0.695 to 0.905. Within the range of 4 to 30 features, the accuracy rate oscillates between 0.892 and 0.945, marking a variance rate of 5.94%.

Table 8. Comparison of average accuracy rates using 10-fold cross validation of KNN for different datasets before and after feature selection via FSvACO (the best result is shown in boldface for each dataset).

	Before Feature Selection		After Feature Selection	
Dataset	# of Features	Accuracy Rate	# of Features	Accuracy Rate
WDBC	30	0.929	27	**0.942**
Dermatology	34	0.879	23	**0.967**
Ionosphere	34	0.843	3	**0.869**
Arrhythmia	279	0.628	129	**0.638**
Wine	13	0.691	8	**0.943**
Hepatitis	19	0.766	4	**0.818**
Spambase	57	**0.998**	34	**0.998**
Madellon	500	0.731	497	**0.733**

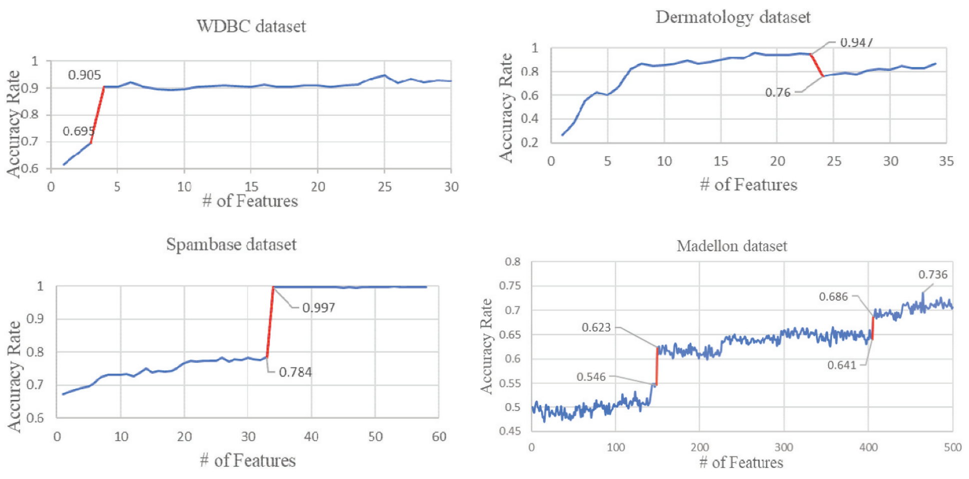

Figure 3. Obtained accuracy rates of KNN (K = 5) after sequentially adding the features selected by the FSvACO method for the chosen four datasets.

For the Dermatology dataset, an increment from 23 to 24 features is observed to decrease accuracy by 24.6%, from 0.947 down to 0.760, indicating some features may lead to misclassification. In the Spambase dataset, increasing features from 33 to 34 boosts accuracy by 27%, from 0.784 to 0.997. Additionally, for Spambase, applying the KNN algorithm takes 0.77 seconds with 57 attributes, compared to 0.71 seconds with 34 attributes, yielding an 8.4% improvement in time efficiency of feature selection. Lastly, in the Madellon dataset, a feature increase from 148 to 149 enhances accuracy by 14%, from 0.546 to 0.623, and an increase from 405 to 406 features sees a 7% rise in accuracy, from 0.641 to 0.686, with the accuracy rate varying between 0.686 and 0.736 when comparing between 406 and 500 features.

Conclusion

Feature selection plays a crucial role in industries as it helps to improve the efficiency and accuracy of data analysis and modeling. By selecting the most relevant features, it reduces the dimensionality of data and improves the performance of machine learning algorithms. This has a significant impact on industries such as intrusion detection, text categorization, DNA microarray analysis, music information retrieval, image retrieval, customer relationship management, and remote sensing (Kumar and Minz 2014). These industries heavily rely on data analysis and modeling, and the use of feature selection can greatly enhance their processes and outcomes. Additionally, feature selection helps to reduce the computational time and resources needed for data analysis, making it a valuable tool for industries dealing with large and complex datasets. The proposed approach has industrial significance because it aims to reduce the computational costs and improve the results of feature selection. The proposed approach has been compared to other feature selection algorithms in the literature, demonstrating its effectiveness and potential for practical use in various industries.

The FSvACO method proposed here directs ant movements by considering feature similarity and pheromone levels. The selection process is fine-tuned by focusing on the top 80% of the path chosen by the best-performing ant in each run. Upon completion of all runs, the feature that appears most frequently in the frequency list is deemed the most crucial. Subsequent features are ranked accordingly. The classification algorithm then takes these features into account, starting with the most significant feature and progressively incorporating others based on their rank of importance. Generally, adding a certain number of key features to the dataset does not significantly alter the performance. This point, where performance gain levels off, is identified as the optimal feature count for that particular dataset and classifier combination.

For benchmarking the proposed algorithm against existing methods, studies have been found in the literature that were conducted under consistent conditions (like dataset, division of data, type of classifier, metric of performance, and so on). Initially, in a comparison with 15 other algorithms on four datasets, the proposed method outperformed the others on two. Further comparative analysis with three algorithms across eight datasets revealed superior results in half of these datasets with the proposed method. Additionally, other evaluations using the f-score as a performance metric have been carried out. It has been demonstrated that classifiers show a statistically significant improvement in performance when they are fed selected features rather than the complete set. Lastly, by observing the improvements gained from sequentially feeding the features selected by the proposed method into the classification algorithm, studies have been conducted regarding the method's convergence performance.

The novel contribution of this paper is the development of the FSvACO algorithm, which uniquely applies ant colony optimization techniques to feature selection by integrating feature similarity and pheromone levels. This approach strategically directs the movement of ants in processing high-dimensional datasets, effectively reducing data dimensionality while retaining essential information. This innovative method is a significant addition to the current literature.

It would be useful to provide information about the limitations of this article. It should be noted that the datasets used during the study were consistent with the ones recommended in the literature on the subject. The analysis employed the same data splitting method, the same number of runs, and the same classifier and performance metric, allowing for fair comparisons. The study utilized two algorithms, KNN and SVM, both of which have been proven effective in numerous studies. It is worth noting that more recent classification techniques could potentially yield even better results. Based on the results, the proposed method can be used in feature selection problems. When there is no time constraint, better routes can be obtained by increasing the number of ants and iterations in the ACO. Better prediction values can be obtained if ensemble learning is performed by obtaining more than one feature subset by random selection of features with the same frequency value. Since the success of the proposed method varies with the classification methods, the results can be evaluated by more advanced classification techniques that have not been evaluated within the scope of the study. Since the heuristic distance calculation method used in the ACO greatly affects the success of the algorithm, studies can also be carried out using methods that can better capture inter-feature patterns. In addition, if mutual information is exchanged between the ACO and the classification algorithms by using the embedded method instead of the hybrid approach, better routes might be obtained by evaluating the route selection of the ants with the classification results.

Disclosure Statement

No potential conflict of interest was reported by the author(s).

ORCID

Duygu Yilmaz Eroglu http://orcid.org/0000-0002-7730-2707

References

Aghdam, M. H., N. Ghasem-Aghaee, and M. Ehsan Basiri. 2009. Text feature selection using ant colony optimization. *Expert Systems with Applications* 36 (3):6843–53.

Agrawal, P., H. F. Abutarboush, T. Ganesh, and A. W. Mohamed. 2021. Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019). *Institute of Electrical and Electronics Engineers Access* 9:26766–91.

Akinola, O. O., A. E. Ezugwu, J. O. Agushaka, R. A. Zitar, and I. Abualigah. 2022. Multiclass feature selection with metaheuristic optimization algorithms: A review. *Neural Computing & Applications* 34 (22):19751–90.

Al-Ani, A. 2007. Ant colony optimization for feature subset selection. *International Journal of Computer and Information Engineering* 1 (4):999–1002.

Ali, S. I., and W. Shahzad. 2012. A feature subset selection method based on symmetric uncertainty and ant colony optimization. In *2012 International Conference on Emerging Technologies*, Islamabad, Pakistan, IEEE.

Bindu, M. G., and M. K. Sabu. 2020. A hybrid feature selection approach using artificial bee colony and genetic algorithm. In *2020 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)*, Cochin, India, IEEE.

Boser, B. E., I. M. Guyon, and V. N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, Pittsburgh, Pennsylvania, USA.

Cover, T., and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13 (1):21–27.

Deneubourg, J.-L., S. Aron, S. Goss, and J. M. Pasteels. 1990. The self-organizing exploratory pattern of the argentine ant. *Journal of Insect Behavior* 3:159–68.

Deriche, M. 2009. Feature selection using ant colony optimization. In *2009 6th International Multi-Conference on Systems, Signals and Devices*, Djerba, Tunisia, IEEE.

Dorigo, M. Optimization, learning and natural algorithms. 1992. *Ph. D. Thesis*, Politecnico di Milano.

Dorigo, M., M. Birattari, and T. Stutzle. 2006. Ant colony optimization. *IEEE Computational Intelligence Magazine* 1 (4):28–39.

Dua, D., and C. Graff. 2017. UCI machine learning repository. https://archive.ics.uci.edu/.

Eroglu, D. Y., and K. Kilic. 2017. A novel hybrid genetic local search algorithm for feature selection and weighting with an application in strategic decision making in innovation management. *Information Sciences* 405:18–32.

Fahrudin, T. M., I. Syarif, and A. Ridho Barakbah. 2016. Ant colony algorithm for feature selection on microarray datasets. In *2016 International Electronics Symposium (IES)*, Denpasar, Indonesia, IEEE.

Ferreira, A. J., and M. A. Figueiredo. 2012. An unsupervised approach to feature discretization and selection. *Pattern recognition* 45 (9):3048–60.

Fix, E., and J. L. Hodges Jr. 1989. Discriminatory analysis. Nonparametric discrimination: Consistency properties. *International Statistical Review*. 57 (3):238–47.

Garnier, S., J. Gautrais, and G. Theraulaz. 2007. The biological principles of swarm intelligence. *Swarm Intelligence* 1:3–31.

Ghosh, M., R. Guha, R. Sarkar, and A. Abraham. 2020. A wrapper-filter feature selection technique based on ant colony optimization. *Neural Computing & Applications* 32:7839–57.

Guo, J., and W. Zhu. 2018. Dependence guided unsupervised feature selection. *Proceedings of the AAAI Conference on Artificial Intelligence* 32 (1):2232–2239.

Haindl, M., P. Somol, D. Ververidis, and C. Kotropoulos. 2006. Feature selection based on mutual correlation. In *Progress in Pattern Recognition, Image Analysis and Applications: 11th Iberoamerican Congress in Pattern Recognition, CIARP 2006 Cancun, Mexico*, Berlin Heidelberg, Springer. November 14-17, 2006. *Proceedings 11*.

Hamed, T., R. Dara, and S. C. Kremer. 2014. An accurate, fast embedded feature selection for SVMs. In *2014 13th International conference on machine learning and applications*, Detroit, MI, USA, IEEE.

He, X., D. Cai, and P. Niyogi. 2005. Laplacian score for feature selection. *Advances in Neural Information Processing Systems* 18.

Huang, X., P. Yun, S. Wu, and Z. Hu. 2023. Abnormal driving behavior detection based on an improved ant colony algorithm. *Applied Artificial Intelligence* 37 (1):2216060.

Kanan, H. R., and K. Faez. 2008. An improved feature selection method based on ant colony optimization (ACO) evaluated on face recognition system. *Applied Mathematics and Computation* 205 (2):716–25.

Kashef, S., and H. Nezamabadi-Pour. 2015. An advanced ACO algorithm for feature subset selection. *Neurocomputing* 147:271–79.

Kumar, V., and S. Minz. 2014. Feature selection: A literature review. *SmartCR* 4 (3):211–29.

Lai, C., M. J. Reinders, and L. Wessels. 2006. Random subspace method for multivariate feature selection. *Pattern Recognition Letters* 27 (10):1067–76.

Li, Y., B.-L. Lu, and Z.-F. Wu. 2006. A hybrid method of unsupervised feature selection based on ranking. In *18th International Conference on Pattern Recognition (ICPR'06)*, Hong Kong, Vol. 2. IEEE.

Li, Z., Y. Yang, J. Liu, X. Zhou, and H. Lu. 2012. Unsupervised feature selection using nonnegative spectral analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence* 26 (1):1026–1032.

Malini Devi, G., M. Seetha, and K. V. N. Sunitha. 2016. A novel K-Nearest neighbor technique for data clustering using swarm optimization. *International Journal Geoinformatics* 12 (1).

Min, H., and W. Fangfang. 2010. Filter-wrapper hybrid method on feature selection. In *2010 Second WRI Global Congress on Intelligent Systems*, Wuhan, China, Vol. 3. IEEE.

Mitra, P., C. A. Murthy, and K. P. Sankar. 2002. Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 24 (3):301–12.

Mohammed, S. K., F. Deeba, F. M. Bui, and K. A. Wahid. 2016. Feature selection using modified ant colony optimization for wireless capsule endoscopy. In *2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, New York, NY, USA, IEEE.

Naser, M. Z., and H. A. Amir. 2023. Error metrics and performance fitness indicators for artificial intelligence and machine learning in engineering and sciences. *Architecture, Structures and Construction* 3 (4):499–517.

Palma-Mendoza, R.-J., L. de-Marcos, D. Rodriguez, and A. Alonso-Betanzos. 2019. Distributed correlation-based feature selection in spark. *Information Sciences* 496:287–99.

Paniri, M., M. Bagher Dowlatshahi, and H. Nezamabadi-Pour. 2020. MLACO: A multi-label feature selection algorithm based on ant colony optimization. *Knowledge-Based Systems* 192:105285.

Qiu, G., and W. Huang. 2023. Mechanical design, manufacture and automation: Research progress and fusion ant colony algorithm-based optimization. *Applied Artificial Intelligence* 37 (1):2219565.

Rostami, M., K. Berahmand, E. Nasiri, and S. Forouzandeh. 2021. Review of swarm intelligence-based feature selection methods. *Engineering Applications of Artificial Intelligence* 100:104210.

Saraç, E., and S. Ayşe Özel. 2014. An ant colony optimization based feature selection for web page classification. *Scientific World Journal* 2014:1–17.

Shunmugapriya, P., and S. Kanmani. 2017. A hybrid algorithm using ant and bee colony optimization for feature selection and classification (AC-ABC hybrid. *Swarm and Evolutionary Computation* 36:27–36.

Sivagaminathan, R. K., and S. Ramakrishnan. 2007. A hybrid approach for feature subset selection using neural networks and ant colony optimization. *Expert Systems with Applications* 33 (1):49–60.

Solorio-Fernández, S., J. Ariel Carrasco-Ochoa, and J. Fco Martínez-Trinidad. 2016. A new hybrid filter–wrapper feature selection method for clustering based on ranking. *Neurocomputing* 214:866–80.

Solorio-Fernández, S., J. Ariel Carrasco-Ochoa, and J. Fco Martínez-Trinidad. 2020. A review of unsupervised feature selection methods. *Artificial Intelligence Review* 53 (2):907–48.

Solorio-Fernández, S., J. Fco Martínez-Trinidad, and J. Ariel Carrasco-Ochoa. 2017. A new unsupervised spectral feature selection method for mixed data: A filter approach. *Pattern Recognition* 72:314–26.

Tabakhi, S., P. Moradi, and F. Akhlaghian. 2014. An unsupervised feature selection algorithm based on ant colony optimization. *Engineering Applications of Artificial Intelligence* 32:112–23.

Tabakhi, S., A. Najafi, R. Ranjbar, and P. Moradi. 2015. Gene selection for microarray data classification using a novel ant colony optimization. *Neurocomputing* 168:1024–36.

Tang, C., X. Liu, M. Li, P. Wang, J. Chen, L. Wang, and W. Li. 2018. Robust unsupervised feature selection via dual self-representation and manifold regularization. *Knowledge-Based Systems* 145:109–20.

Unler, A., A. Murat, and R. Babu Chinnam. 2011. mr2PSO: A maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification. *Information Sciences* 181 (20):4625–41.

Varshavsky, R., A. Gottlieb, M. Linial, and D. Horn. 2006. Novel unsupervised feature filtering of biological data. *Bioinformatics* 22 (14):e507–13.

Wald, R., T. M. Khoshgoftaar, and A. Napolitano. 2013. Should the same learners be used both within wrapper feature selection and for building classification models? In *2013 IEEE 25th International Conference on Tools with Artificial Intelligence*, Herndon, VA, USA, IEEE.

Wang, Z., S. Gao, M. Zhou, S. Sato, J. Cheng, and J. Wang. 2022. Information-theory-based nondominated sorting ant colony optimization for multiobjective feature selection in classification. *IEEE Transactions on Cybernetics* 53 (8):5276–5289.

Yang, J., A. Javed Awan, and G. Vall-Llosera. 2019. Support vector machines on noisy intermediate scale quantum computers. *arXiv preprint arXiv* 1909:11988.

Yang, Y., H. T. Shen, Z. Ma, Z. Huang, and X. Zhou. 2011. ℓ 2, 1-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI international joint conference on artificial intelligence*, Barcelona, Catalonia, Spain.

Yan, H., and J. Yang. 2015. Sparse discriminative feature selection. *Pattern recognition* 48 (5):1827–35.

Zeng, H., and Y.-M. Cheung. 2010. Feature selection and kernel learning for local learning-based clustering. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 33 (8):1532–47.

Zhang, S., C. Zhang, and Q. Yang. 2003. Data preparation for data mining. *Applied Artificial Intelligence* 17 (5–6):375–81.

Zhao, Z., and H. Liu. 2007. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th international conference on Machine learning*, Corvalis, Oregon, USA.