EMPOWERING ONLINE SHOPPING SENTIMENT ANALYSIS USING TENACIOUS ARTIFICIAL BEE COLONY INSPIRED TAYLOR SERIES-BASED GAUSSIAN MIXTURE MODEL (TABC-TSGMM)

G.M. BALAJI[1], K. VADIVAZHAGAN[2]

[1]Research Scholar & Assistant Professor, Department of Computer and Information Science,
Annamalai University, Chidambaram, Tamilnadu, India.
[2] Assistant Professor, Department of Computer and Information Science,
Annamalai University, Chidambaram, Tamilnadu, India.
E-mail: [1]gmbalaji0813@gmail.com, [2]vadivazhagan.k@gmail.com

ABSTRACT

Sentiment Analysis has become increasingly important in online shopping, where consumers rely on reviews and feedback to make informed purchasing decisions. However, accurate sentiment classification poses challenges, such as handling nuanced language and varying review lengths. This study introduces a novel approach called the Tenacious Artificial Bee Colony inspired Taylor Series-based Gaussian Mixture Model (TABC-TSGMM) to address these challenges.TABC-TSGMM leverages two key components: the Taylor Series-based Gaussian Mixture Model (TSGMM) and the Tenacious Artificial Bee Colony (TABC). TSGMM captures complex sentiment patterns in text data, while TABC optimizes the model's performance through an intelligent search strategy.TABC enhances TSGMM by optimizing model parameters, making the sentiment analysis process more robust and accurate. To evaluate the effectiveness of TABC-TSGMM, we conducted experiments on an Amazon product review dataset, focusing on electronic products. The results demonstrated superior classification accuracy, showcasing the potential of this approach to empower online shopping sentiment analysis.

Keywords: *Sentiment Analysis, ABC, GMM, Taylor Series, Local Search. Optimization*

1. INTRODUCTION

Online shopping has reshaped how we purchase products, offering unparalleled convenience and versatility. With just a few clicks or taps, consumers can explore various items, from clothing and electronics to groceries and more, all from the comfort of their homes [1], [2]. The convenience of online shopping is undeniable, allowing shoppers to browse and make purchases 24/7, unrestricted by traditional store hours. Moreover, online retailers provide comprehensive product information, customer reviews, and user-friendly search filters, streamlining the process of finding the perfect item [3]–[5]. One of the primary advantages of online shopping is the vast variety it offers. Shoppers can explore products worldwide, discovering unique items and niche brands not always accessible locally. This diversity empowers consumers to make well-informed choices and uncover products that align precisely with their preferences [6].

Sentiment analysis, also known as opinion mining, stands at the intersection of natural language processing and emotional understanding, focusing on discerning the emotional tone, opinions, and sentiments conveyed within textual data[7]–[9]. It transcends the boundaries of mere text recognition, delving into comprehending human emotions and perspectives conveyed through words. At its core, sentiment analysis categorizes text as expressing positive, negative, or neutral sentiments. Widely applied across diverse domains, from businesses analyzing customer feedback to tracking social media discussions and understanding public sentiments, it yields valuable insights into how people perceive products, services, and societal issues[10], [11]. Employing techniques such as natural language processing, machine learning, and deep learning, sentiment analysis continually advances, emerging as an indispensable tool for organizations, researchers, and decision-makers, facilitating a more profound comprehension of public sentiment [12].

Sentiment analysis, a powerful tool for deciphering public sentiment and opinions, faces many challenges. Language complexity, subjectivity, and evolving slang can make accurate sentiment capture challenging [13]. Managing real-time data and ethical concerns regarding bias and privacy further complicates the complexity. From a customer standpoint, sentiment analysis offers advantages such as personalized product recommendations and enhanced customer support thanks to its ability to understand and respond to customer sentiments[14], [15]. It provides valuable insights into customer feedback and market trends for companies, ultimately driving informed decision-making.

1.1. Problem Statement

The “Implicit Sentiment” challenge in sentiment analysis revolves around the intricate and multifaceted ways sentiments are subtly conveyed within textual data. The primary problem lies in the diverse linguistic devices employed to imply sentiment, including sarcasm, irony, metaphors, and cultural references, which confound sentiment analysis models. Additionally, the contextual dependency of implicit sentiment, where the exact words can carry contrasting sentiments based on context, poses a significant challenge. The ever-evolving nature of language through new phrases and linguistic trends further complicates sentiment analysis, as implicit sentiment can be deeply embedded within these evolving linguistic expressions. Addressing this challenge is critical for enhancing the accuracy and nuance of sentiment analysis, ensuring that the underlying emotions and opinions are effectively uncovered in textual data.

1.2. Motivation

The motivation to address the challenge of “Implicit Sentiment” in sentiment analysis stems from the recognition that language is a dynamic, nuanced medium of expression, and accurate sentiment analysis requires deciphering subtle emotional cues. Implicit sentiment presents a unique set of difficulties, as sentiments are often embedded within layers of linguistic complexity, making them elusive to automated analysis. Understanding and extracting these implicit sentiments is crucial for making sentiment analysis models more robust and perceptive. Failure to account for implicit sentiment leads to incomplete and potentially misleading insights, impacting decision-making across various domains. Whether in understanding customer feedback for product improvement or gauging public sentiment on social and political issues, accurately handling implicit sentiment ensures more precise and valuable results. Thus, the motivation to overcome this challenge is grounded in pursuing a deeper, more sophisticated understanding of human sentiment as expressed through the intricate tapestry of language.

1.3. Objectives

This research aims to propose a bio-inspired optimization-based deep learning strategy for addressing the “Implicit Sentiment” challenge in sentiment analysis, motivated by the need to enhance the accuracy and depth of sentiment analysis results. This objective aims to harness the power of bio-inspired optimization techniques to train deep learning models that can effectively recognize and interpret implicit sentiment within textual data. The strategy seeks to overcome the intricacies of language and contextual dependencies by optimizing model performance through a hybrid approach that blends biological insights with deep learning capabilities. By achieving this objective, we aspire to offer a more comprehensive and reliable solution for sentiment analysis that can successfully unravel subtle emotional cues embedded within textual content, ultimately leading to more nuanced and accurate insights for decision-making across diverse domains, from product enhancement based on customer feedback to understanding public sentiment on multifaceted issues.

2. LITERATURE REVIEW OF PAPER

Deep Sentiment Analysis” [16] introduces a novel approach to sentiment analysis, known as “Deep-Sentiment.” It is based on developing a decision-based Recurrent Neural Network (RNN) model to analyze and classify sentiment in text data. Unlike traditional sentiment analysis models, the decision-based RNN can capture and model sequential dependencies in the text, allowing it to make informed decisions about sentiment classification. “Multi-attention Fusion” [17] focuses on educational sentiment analysis, specifically within the context of educational big data. It introduces a sophisticated sentiment analysis model incorporating a “Multi-attention Fusion” approach. Unlike traditional sentiment analysis models, this model utilizes multiple attention mechanisms to weigh and process relevant information in educational big data. “SentiDiff” [18] is an innovative method for sentiment analysis on the Twitter platform. The textual information is integrated with sentiment diffusion patterns, a unique approach to sentiment analysis. While traditional sentiment analysis models primarily rely on text data, SentiDiff goes beyond by considering how sentiments spread and evolve across social networks.

"Danmaku Video Sentiment Analysis" [19] is dedicated to sentiment analysis in the context of "Danmaku" videos, which typically include user-generated comments that overlay the video content. The primary contribution is applying the Naïve Bayes classification method and utilizing a sentiment dictionary to analyze and classify the sentiment found within these videos. "Chinese E-Commerce Product Reviews Sentiment Analysis" [20] addresses the challenge of sentiment analysis in Chinese e-commerce product reviews. It integrates the sentiment lexicons and deep learning techniques to analyze and classify sentiment in these reviews. Chinese is a complex language with nuances that require specialized techniques for accurate sentiment analysis. "Refined Sentiment Word Embeddings" [21] introduces refined global word embeddings incorporating sentiment concepts. Sentiment concepts are integrated into the word embeddings by associating each word with its sentiment score, capturing the emotional context of words in the text. This enriched representation allows the model to analyze text data and accurately identify sentiment by considering the words' meanings and emotional connotations. Bio-inspired optimization [31] – [48] started playing important role in all researches.

"Arabic Aspect-Based Sentiment Review" [22] critically assesses aspect-based sentiment analysis in Arabic. It scrutinizes existing research methodologies, datasets, and findings. The review provides insights into the various techniques and approaches used in Arabic sentiment analysis, offering a detailed examination of the strengths and limitations of different methodologies. In "Tweets Sentiment Analysis During COVID-19" [23], tweets related to the System Usability Scale (SUS) are analyzed before and during the COVID-19 pandemic. It employs advanced sentiment analysis techniques to dissect tweets' emotional content and evaluate sentiment changes over time. "AI Understanding of Social Media Sentiment Changes" [24] explores the utilization of artificial intelligence (AI) to investigate the underlying factors influencing changes in sentiment on social media platforms. AI algorithms analyze vast volumes of social media data and identify correlations between specific events, user interactions, and sentiment shifts.

"Context-Aware Hybrid DNN" [25] presents a hybrid Deep Neural Network (DNN) architecture that integrates sentiment awareness and attention mechanisms. The sentiment awareness module provides sentiment context by embedding sentiment scores into the text representations. The attention mechanism enables the DNN to focus on the most relevant parts of the text, improving sentiment classification precision. "Stock Market News Sentiment" [26] explores the application of sentiment analysis within the Brazilian stock market context. The stock market news data is inherently complex due to financial jargon and nuanced expressions. Analyzing sentiments in this domain requires tailored sentiment lexicons and domain-specific classifiers. "Iceberg Challenges in Sentiment Analysis" [27] scrutinizes sentiment analysis's intricate challenges in various domains. The challenges span from handling sentiment ambiguity and sarcasm to addressing language nuances and context dependencies. Researchers have proposed advanced machine learning techniques, deep learning architectures, and fine-grained sentiment analysis models to address these challenges.

3. TENACIOUS ARTIFICIAL BEE COLONY INSPIRED TAYLOR SERIES-BASED GAUSSIAN MIXTURE MODEL

3.1. Taylor series-based Gaussian Mixture Model

In the context of sentiment analysis, the Taylor series [28] and Gaussian Mixture Model [29] apply differently. The Taylor series can extract sentiment-related features from text data, representing linguistic nuances and emotional tones. Meanwhile, the Gaussian Mixture Model is valuable for modeling and classifying sentiment patterns within the data, enabling the identification of sentiment clusters and improving the accuracy of sentiment analysis in various applications, such as social media monitoring, product reviews, and customer feedback analysis. These mathematical techniques enhance the effectiveness and depth of sentiment analysis in the digital age.

3.1.1. Data Preparation

In the Taylor series-based Gaussian Mixture Model, the initial step involves data preparation, where this research mathematically represents the data and ensures it is suitably prepared for subsequent modeling. Given a dataset, $= \{x_1, x_2, \ldots., x_t\}$, consisting of n data points, each of dimension d, the primary goal of arranging the data in a mathematical format conducive to statistical modeling. This often includes standardization or normalization to ensure the data attributes have a consistent scale. Eq.(1) mathematically expresses the data preparation.

$$X = \{x_1, x_2, \ldots, x_t\} \text{ where } x_i \in R^d \quad (1)$$

Understanding the data distribution and determining whether it aligns with the Gaussian Mixture Model assumption is essential. This might involve performing exploratory data analysis to visualize the data distribution, testing for normality, and assessing the number of components (clusters) required for the mixture model. Proper data preparation is fundamental for accurate model convergence and reliable parameter estimation in subsequent steps. It may be necessary to divide the data into training and testing sets for model validation and evaluation, ensuring that the data is representative and unbiased for robust Gaussian mixture modeling.

Algorithm 1: Data Preparation

```
Input:
  • X: A dataset of n data points, each with d dimensions.
  • Desired data preprocessing steps.

Output:
  • Preprocessed dataset ready for GMM modeling.

Procedure:
  1. Input the dataset X and the desired data preprocessing steps.
  2. Check and handle missing values based on the chosen data preprocessing strategy.
  3. Standardize or normalize the data to ensure all features have a consistent scale.
  4. If required, perform dimensionality reduction techniques to reduce the dimensionality of the data.
```

3.1.2. Initialization

Initialization is done by setting up the initial parameters for the Gaussian components. This step is crucial for commencing the Expectation-Maximization (EM) algorithm. Mathematically, for a GMM with Kcomponents, this research initializes the model parameters using Eq.(2) to Eq.(4).

$$\textbf{\textit{Mixture}}\ Weights: \pi_k^{(0)} \ \textbf{for}\ \boldsymbol{k} = 1,2,\ldots,K \quad (2)$$

$$\textbf{\textit{Means}}: \mu_k^{(0)} \ \textbf{for}\ \boldsymbol{k} = 1,2,\ldots,K \quad (3)$$

$$Covariance\ Matrices: \Sigma_k^{(0)} \ \text{for}\ k = 1,2,\ldots,K \quad (4)$$

where $\pi_k^{(0)}$ represents the initial weight (prior probability) assigned to the k-th Gaussian component, $\mu_k^{(0)}$ represents the initial mean vector for the k-th component and $\Sigma_k^{(0)}$ represents the initial covariance matrix for the k-th component. These initial values can be set through various methods, such as random initialization, k-means clustering, or other domain-specific techniques. Proper initialization can significantly impact the convergence and quality of the Gaussian Mixture Model, as it sets the starting point for the subsequent Expectation-Maximization iterations, where these parameters will be refined to better represent the data distribution.

Algorithm 2: Initialization

```
Input:
  • X: Preprocessed dataset.
  • K: The desired number of Gaussian components.
  • Initialization method.

Output:
  • Initial parameters for the GMM.

Procedure:
  1. Input the preprocessed dataset X, the number of components K, and the chosen initialization method.
  2. Initialize the mixture weights for each component, typically with equal values for all components.
  3. Initialize the mean vectors for each component based on the chosen initialization method, which may include random assignment or k-means clustering.
  4. Initialize the covariance matrices for each component, which could involve setting them to an identity matrix or other appropriate initial values.
  5. Return the initial parameters for subsequent iterations of the Expectation-Maximization (EM) algorithm.
```

3.1.3. E-Step (Expectation Step)

Each data point's posterior probability or responsibility for belonging to each Gaussian component is calculated by the method in the E-step. Each data point's influence on the various components is represented by a posterior probability, indicated by w_{ik}. For each data point x_i and for each component k, w_{ik} is calculated using Eq.(5).

$$w_{ik} = \frac{\pi_k . N(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j . N(x_i|\mu_j, \Sigma_k)} \quad (5)$$

where π_k represents the mixture weight (prior probability) for the k-th component, μ_k is the mean vector, Σ_k is the covariance matrix for the *k*-th component, and $N(x_i|\mu_j,\Sigma_k)$ is the probability density function of the multivariate Gaussian distribution, which measures the likelihood of data point x_i given the parameters of component *k*.

Eq.(5) calculates the likelihood of x_i being generated by each of the K components, normalized by the total likelihood of x_i across all components. The result is a set of responsibilities that indicate the proportion of x_i's likelihood attributed to each component. These responsibilities will be used in the M-step to update the model parameters.

3.1.4. M-Step (Maximization Step):

Based on the duties determined in the E-step, the algorithm adjusts the model's parameters in the M-step. Maximizing the data-driven log-likelihood of the present parameters is the target. Mathematically, the parameter updates are done using the Eq.(6) to Eq.(9). The updated mixture weight for the k-th component is the average of the responsibilities for that component over all data points. Eq.(6) ensures that π_k represents the proportion of data points assigned to component k.

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} w_{ik} \quad (6)$$

The updated mean vector for the *k*-th component is a weighted average of the data points, where the weights are the responsibilities. Eq.(7) shifts the mean towards the data points that have a higher likelihood of belonging to component *k*.

$$\mu_k = \frac{\sum_{i=1}^{n} w_{ik} . x_i}{\sum_{i=1}^{n} w_{ik}} \quad (7)$$

The updated covariance matrix for the *k*-th component is a weighted average of the outer products of the data point deviations from the mean. Eq.(8) accounts for the spread and orientation of the data points assigned to component k.

$$\Sigma_k = \frac{\sum_{i=1}^{n} w_{ik} . (x_i - \mu_k)(x_i - \mu_k)^T}{\sum_{i=1}^{n} w_{ik}} \quad (8)$$

After the E-step, the EM method returns to the M-step. It repeats this process until some convergence requirement is reached, such as a minimum change in the log-likelihood or a maximum number of repetitions. This iterative process refines the model parameters and improves the fit of the Gaussian Mixture Model to the observed data. Once convergence is achieved, the model parameters are optimized, and the GMM can be used for tasks such as clustering, density estimation, or generative modeling. The EM algorithm is a powerful tool for unsupervised learning and plays a central role in training Gaussian Mixture Models.

```
Algorithm 3: Expectation-Maximization (EM)
Input:
  • X: Preprocessed dataset
  • Initial parameters: $\pi^{(0)}, \mu^{(0)}, \Sigma^{(0)}$
  • Convergence criteria

Output:
  • Refined parameters ($\pi,\mu,\Sigma$) for the GMM

Procedure:
Step 1: Input the preprocessed dataset X, the initial parameters ($\pi^{(0)}, \mu^{(0)}, \Sigma^{(0)}$), and the convergence criteria.
Step 2: Initialize variables to track the iteration count and the previous log-likelihood value.
Step 3: Iterate Steps 3 to 8 until convergence or reaching the maximum number of iterations:
Step 4: E-Step (Expectation):
    • For each data point $x_i$ and each component k:
    ✓ Calculate the responsibility or posterior probability $w_{ik}$ of $x_i$ belonging to component k without using equations in the algorithm.
    ✓ Normalize the responsibilities for each data point to ensure they sum to 1.
Step 5: M-Step (Maximization):
    • Update the mixture weights ($\pi_k$) based on the normalized responsibilities.
    • Update the mean vectors ($\mu_k$) by computing weighted averages of the data points.
    • Update the covariance matrices ($\Sigma_k$) by computing weighted covariances of the data points.
Step 6: Calculate the log-likelihood of the data given the current parameters.
Step 7: Check for convergence by comparing the change in log-likelihood with the predefined threshold. If the change is small or the maximum number of iterations is reached, exit the loop.
Step 8: Update the iteration count.
```

Step 9: Return the refined parameters (π, μ, Σ) after the EM algorithm has converged or reached the maximum allowed iterations

3.1.5. Convergence

This step evaluates the trained Taylor series-based GMM to assess its quality and fit to the data. The primary objective is determining how well the model represents the underlying data distribution. Evaluation involves several important aspects, both quantitative and qualitative, and can be expressed mathematically as follows:

The log-likelihood of the data given the trained GMM is a fundamental quantitative measure of model fit. It quantifies how well the model explains the observed data. The log-likelihood is computed using Eq.(9), as the sum of the logarithms of the probabilities of each data point under the GMM.

$$\log P(X|\theta) = \sum_{i=1}^{n} log\left(\sum_{k=1}^{K} \pi_k . N(x_i|\mu_k, \Sigma_k)\right) \quad (9)$$

where X represents the dataset, θ denotes the model parameters (mixture weights, means, and covariance matrices), and π_k ,μ_k and are the parameters of the k-th Gaussian component. Maximizing this log-likelihood is a primary goal during GMM training, and a higher log-likelihood indicates a better fit to the data.

Model fit, and complexity may be evaluated using the Bayesian Information Criterion (BIC) criteria. It helps with model selection since it punishes complex models. The BIC for a GMM can be calculated using Eq.(10).

$$BIC = -2\log P(X|\theta) + d.log(n) \quad (10)$$

The amount of data points, n, is multiplied by the total number of model parameters, d. The BIC encourages models that fit the data well while being as economical as possible. A lower BIC score represents a better balance between fit and complexity.

Cross-validation is a technique used to assess how well the GMM generalizes to unseen data. It involves splitting the dataset into a training set and a validation (or test) set. The trained GMM is then evaluated on the validation set to check its performance on new, unseen data. Over fitting, in which a model fits the training data so well that it doesn't generalize well, can be spotted by cross-validation. GMMs can also be used for outlier detection. In this context, the model can be evaluated based on its ability to identify data points that deviate significantly from the learned distribution. Mathematically, outlier detection can involve setting a threshold on the log-likelihood or the Mahalanobis distance to identify data points as outliers. Evaluating the robustness of the GMM to variations in the data or perturbations in the model parameters is crucial. Sensitivity analysis, which explores how parameter changes affect the model's performance, can be a part of this evaluation.

This encompasses various mathematical and quantitative methods for assessing the Taylor series-based Gaussian Mixture Model's quality and effectiveness in representing the data. These evaluation techniques ensure that the model aligns with the underlying data distribution, generalizes well, and is suitable for its intended application, whether it's clustering, density estimation, or anomaly detection. By combining quantitative metrics with qualitative visual inspection, practitioners can make informed decisions about the model's validity and utility in their specific context.

3.1.6. Prediction and Clustering

This step involves deploying and utilizing the GMM to perform various tasks. The mathematical representation of the deployment step can vary depending on the application, but this research discusses its key aspects in a unified manner.

(a). Inference and Prediction

A trained GMM's primary application is to make inferences and predictions based on the model. For example, given a new data point x_{new}, can use the GMM to estimate the probability that x_{new} belongs to each of the Gaussian components. Eq.(11) mathematically expresses the inference and prediction phase.

$$P(Component\ k|x_{new}, \theta) = \frac{\pi_k . N(x_{new}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j . N\left(x_{new}|\mu_k, \Sigma_j\right)} \quad (11)$$

This allows to assign x_new to the most probable component or cluster, enabling tasks like classification or label assignment based on the GMM.

(b). Clustering

If the GMM was trained for clustering, data points can be assigned to the clusters they most likely belong to. This is achieved by identifying the Gaussian component with the highest responsibility for each data point. Eq.(12) enables data segmentation and grouping based on the GMM's learned clusters.

$$Cluster(x_i) = \arg\max_k \left(P(Component\ K|x_i, \theta)\right) \quad (12)$$

(c). Density Estimation

The GMM can be used to estimate the probability density of data points within the feature space. For any point x, the density can be computed as the weighted sum of the density contributions from all components. Eq.(13), the density estimation, can be used for anomaly detection or modeling data distribution for further analysis.

$$P(x|\theta) = \sum_{k=1}^{K} \pi_k . N(x|\mu_k, \Sigma_k) \quad (13)$$

(d). Sampling and Data Generation

Trained GMMs can generate synthetic data points that follow the learned distribution. Eq.(14) is applied to achieve the sampling by Gaussian components according to their mixture weights. It will be helpful for data augmentation, simulation, or generative modeling tasks.

$$x_{synthetic} \sim \sum_{k=1}^{K} \pi_k . N(\mu_k, \Sigma_k) \quad (14)$$

(e). Anomaly Detection

Anomaly detection in sentiment analysis is identifying unusual or atypical sentiment expressions in text data, which can help spot outliers, detect fraud, provide early warnings, and assess model quality. It involves finding sentiments that deviate significantly from the norm using various techniques, such as statistical or machine-learning algorithms. By setting an appropriate threshold on the log-likelihood or the Mahalanobis distance, the GMM can identify anomalies or outliers in new data. Points with log-likelihoods below the threshold are considered anomalies. Eq.(15) express the anomaly detection.

$$Anomaly(x_i) = \begin{cases} True\ \ if\ log\ P(x_i|\theta) < Threshold \\ False\ \ otherwise \end{cases} \quad (15)$$

Algorithm 4: Prediction and Clustering

Input:
- Training Data: A dataset containing features.
- Number of Components (K): The desired number of clusters/components.
- Initial Parameter Guesses: Initial values for GMM parameters.

Output:
- Trained GMM Model
- The probability of belonging to each component.
- Cluster Assignment: Assignment of data points to clusters.
- Density Estimation: Probability density estimates for data points.
- Synthetic Data Points: Data points generated according to the learned distribution.
- Anomaly Detection: Identification of anomalies or outliers.

Procedure:

Step 1: Data and Initialization
- Start with your training data, specify the number of components (K), and initialize GMM parameters.

Step 1: Training the GMM
- Use the Expectation-Maximization (EM) algorithm to iteratively update GMM parameters.
- Achieve convergence for a trained GMM model.

Step 3: Utilizing the Trained GMM
- For a new data point, calculate its predicted cluster or probability of belonging to each component.
- Assign data points to clusters based on the component with the highest responsibility.
- Estimate the probability density of data points within the feature space.
- Generate synthetic data points following the learned distribution.

Detect anomalies or outliers based on a chosen threshold.

3.2. Tenacious Artificial Bee Colony Optimization

Artificial Bee Colony Optimization (ABC)[30]is a nature-inspired optimization algorithm inspired by the foraging behavior of honeybees. It mimics bees' search for food sources, using employed, onlooker, and scout bees to explore and evaluate potential solutions. ABC is a versatile and effective tool for solving various complex optimization problems. TABC is an advancement of ABC.

3.2.1. Initialization

Initialization is the first and fundamental step in the TABC algorithm, as it sets the stage for the entire optimization process. In this step, we establish the initial conditions, including the population of employed bees, their solutions, and the memory of

Memorizer Bees. Additionally, we define parameters that influence the algorithm's behavior, such as the maximum number of iterations, the limit on abandonment, and the neighbourhood size for local search.

The TABC algorithm begins by generating a workforce of size N. In the search space, worker bee *i* is denoted by a solution vector x_i. Mathematically, this can be expressed as Eq.(16)

$$E = \{x_1, x_2, \ldots\ldots, x_N\} \quad (16)$$

where x_i is a D -dimensional solution vector (i.e., $x_i=[x_{i1},x_{i2},\ldots\ldots,x_{iD}]$) that may be used to solve the optimization issue. These solution vectors are created randomly within the reachable part of the search space.

The Memorizer Bees are crucial in maintaining and sharing information about high-quality solutions throughout the optimization process. The memory of Memorizer Bees, denoted as *M*, is initialized as an empty set as expressed in Eq.(17). As the algorithm continues, the best solutions identified by working bees and observers throughout iterations will be stored in this memory set.

$$M = \{\} \quad (17)$$

The TABC algorithm's behavior may be modified by setting several parameters. These parameters influence the exploration and exploitation of the search space and the overall convergence of the algorithm. Some of the critical parameters include:

- T_{max}: The maximum number of iterations or a termination criterion, indicating the stopping condition for the algorithm.
- $\text{limit}_{\text{abandon}}$: $\text{limit}_{\text{abandon}}$: A threshold value that, if not met, causes a solution to be abandoned after a certain number of repetitions.
- $\text{neighborhood}_{\text{size}}$: The size of the neighborhood around each employed bee's solution, which is used in the local search process.
- Other parameters related to the employed bee local search strategy and probability calculations for onlooker bees.

Mathematically, these parameters are defined as Eq.(18)

$$T_{max} = maximum\ number\ of\ iterations \quad (18)$$

$\text{limit}_{\text{abondon}}$=limit on abandonment

$\text{neighborhood}_{\text{size}}$= size of the neighborhood for local search

Algorithm 5: Initialization

Input:
- N represents the total population of working bees.
- The maximum number of iterations, T*max*.
- Parameters include the limit on abandonment ($\text{limit}_{\text{abondon}}$) and the neighborhood size for local search ($\text{neighborhood}_{\text{size}}$).
- The feasible region of the search space.

Output:
- A population of employed bees, each initialized with a random solution within the search space.
- An empty memory for Memorizer Bees.

Procedure:

Step 1: Initialize a counter, t, to keep track of the current iteration, starting at 1.

Step 2: Create an empty memory set for Memorizer Bees: $M = \{\}$.

Step 3: Initialize a population of N employed bees:
- For each employed bee *i* (from 1 to *N*):
- Generate a random solution vector, x_i, within the feasible region of the search space.
- Evaluate the fitness of x_i, typically by calculating the objective function value $f(x_i)$.

Step 4: Set the algorithm parameters:
- Maximum number of iterations: T_{max}.
- Limit on abandonment: $\text{limit}_{\text{abandon}}$.
- Neighborhood size for local search: $\text{neighborhood}_{\text{size}}$.

Step 5: Proceed to the next step of the TABC algorithm, which comprises the Employed Bee step.

3.2.2. Employed Bee Phase

The Employed Bee Phase is a critical component of the Artificial Bee Colony (TABC) optimization algorithm, where employed bees actively explore the search space and attempt to exploit promising solutions. In this phase, the employed bees evaluate their current solutions and perform local neighborhood searches to seek improvements. The primary objective is to find better solutions that can lead to the optimization of the objective function. This phase embodies the foraging behavior of honeybees that actively collect nectar from flowers.

Each employed bee *i* starts by evaluating the fitness of its current solution. This involves calculating the objective function value, $f(x_i)$, associated with the current solution x_i. Mathematically, it can be represented as Eq.(19).

$f(x_i)$= Objective function evaluation for solution x_i (19)

This fitness value calculated using Eq.(19) provides a measure of the quality of the solution and serves as a basis for comparison with candidate solutions.

After evaluating the current solution, employed bees perform a local search within their neighborhoods. The local search is a crucial step as it allows bees to explore the vicinity of their current solutions. The neighborhood size, $\text{neighbourhood}_{\text{size}}$, is a parameter that defines the extent of the local search around each employed bee.

During the local search, employed bees generate new candidate solutions, denoted as x_i', in the neighbourhood. For each candidate solution, the employed bee calculates its fitness value $f(x_i')$ by applying the objective function to the candidate solution. The mathematical representation is:

$f(x_i')$= Objective function evaluation for a candidate solution x_i' (20)

These fitness values calculated using Eq.(20) measure the quality of the candidate solutions, enabling employed bees to determine whether improvements have been achieved.

The worker bees evaluate the potential solutions by contrasting their fitness levels with those of the existing solutions. If a candidate's solution (x_i') is found to be better *(i.e., $f(x_i') < f(x_i)$)*, the employed bee updates its solution to the improved candidate solution. The mathematical representation of the solution update is given in Eq.(21). This step ensures that employed bees continuously seek improvements in their solutions.

$x_i \leftarrow x_i'$ if $f(x_i') < f(x_i)$ (21)

After evaluating their solutions and potentially updating them, employed bees share information about the quality of their solutions with the onlooker bees. This information sharing typically involves conveying the fitness values of their solutions, which onlooker bees use to make informed decisions in the Onlooker Bee Phase.

Algorithm 6: Employed Bee Phase

Input:

- The population of employed bees, each with its current solution.
- The objective function is to evaluate solution quality.

Output:

- Updated solutions for employed bees after local search.

Procedure:

Step 1: In the population, for every worker bee *i* who has a job:

- Evaluate the fitness or quality of the current solution x_i by applying the objective function to it.
- Perform a local search in the neighbourhood of x_i to generate a candidate solution x_i'.
- Applying the goal function to the potential answer x_i' will help determine how well it fits the problem.
- If $f(x_i')$ is better than the current fitness $f(x_i)$, update the solution x_i to x_i'.

Step 2: After processing all employed bees, the employed bee population now contains updated solutions based on local search.

3.2.3. Onlooker Bee Phase

This phase mimics the behavior of foraging bees that select their food sources based on factors like nectar quality. In the TABC algorithm, onlooker bees select employed bees with solutions of potentially higher quality and engage in local searches to generate new candidate solutions.The probability calculation step involves determining the selection probabilities for employed bees based on the quality of their solutions. Typically, the probability P*(i)* of selecting an employed bee i is calculated using Eq.(22).

$$P(i) = \frac{f(x_i)}{\sum_{k=1}^{N} f(x_k)} \quad (22)$$

where P*(i)* is the probability of selecting employed bee i, $f(x_i)$ is the fitness (objective function value) of the solution x_i, N is the total number of employed bees, $\sum_{k=1}^{N} f(x_k)$represents the sum of fitness values of all employed bees in the population.

Based on the calculated probabilities, onlooker bees choose employed bees to follow. Employed bees are selected with probabilities determined in the previous step. The selection process is random, with the likelihood of selection proportional to the quality of the solutions. Onlooker bees may choose multiple employed bees to follow in this manner. After selecting employed bees to follow, onlooker bees perform local searches in the neighborhoods of the employed bees they have chosen. The neighborhood size is typically defined as the parameter neighborhoodsize. The onlooker bees explore the vicinity of their selected employed bees' solutions to generate new candidate solutions.For each candidate solution generated during the local search, the onlooker bees evaluate the fitness values, $f(x_i')$, to determine whether they represent improvements over the current solutions. The fitness values are calculated by applying the objective function to the candidate solutions, and this step helps onlooker bees assess the quality of the generated solutions.

If a candidate's solution (x_i') is found to be better (i.e., $f(x_i') < f(x_i)$), the onlooker bee updates its selection to the improved candidate solution. The mathematical representation of the solution update is similar to that in the Employed Bee Phase and it is expressed as Eq.(23). This step ensures that onlooker bees actively seek improvements in their selected solutions.

$$x_i \leftarrow x_i' \text{ if } f(x_i') < f(x_i') \quad (23)$$

After evaluating solutions and potentially updating them, onlooker bees share information about the quality of the selected solutions with the Memorizer Bees. Memorizer Bees maintain a memory of high-quality solutions, and the shared information contributes to the collective knowledge of the swarm, helping the TABC algorithm converge toward optimal solutions over time.

Algorithm 7: Onlooker Bee Phase

Input:

- Employed bees with their current solutions.
- Objective function for evaluating solution quality.
- Local search neighborhood size (neighbourhoodsize).
- Memorizer Bees' memory containing high-quality solutions.

Output:

- Updated solutions for onlooker bees after local search.

Procedure:

Step 1: Calculate selection probabilities for employed bees.
Step 2: Select onlooker bees based on probabilities.
Step 3: For each selected employed bee, perform local search and update solutions.
Step 4: Onlooker bees contains updated solutions.
Step 5: Share solution quality with Memorizer Bees for collective knowledge.

3.2.4. Memorizer Bee Phase

The Memorizer Bee Phase is a fundamental component of the TABC optimization algorithm, responsible for maintaining and preserving a memory of high-quality solutions discovered by employed and onlooker bees. In this phase, Memorizer Bees act as information gatherers, aggregating the best solutions found during optimization. This collective knowledge is vital in guiding the swarm toward better solutions and preventing the loss of valuable information.

Memorizer Bees actively update their memory with the best solutions found by employed and onlooker bees during the current iteration. The memory, denoted as *M*, is a collection of solution vectors representing promising solutions. Eq.(24) involves appending the best solutions found in the current iteration to the existing memory.

$$M \leftarrow M \cup \{x_i, x_j, \dots,\} \quad (24)$$

where M represents the current memory of Memorizer Bees, $\{x_i, x_j, \dots,\}$ denotes the best solutions found by employed and onlooker bees in the current iteration, which are added to the memory.

Memorizer Bees are selective in retaining solutions in their memory to ensure they maintain high-quality solutions. The quality of a solution can be determined by its fitness value, *f(x)*, which represents its effectiveness in optimizing the objective function. Memorizer Bees select solutions for retention based on their fitness values. They retain solutions that are among the top-performing solutions based on fitness. The remaining solutions can be limited based on the memory size management strategy.

$$M = Select\ top - performing\ solution\ from\ M\ based\ on\ fitness\ values \quad (24)$$

where M is the current memory of Memorizer Bees, Select top-performing solutions operation ensures that only the best solutions based on fitness are retained.

Managing the memory size is a crucial aspect of the Memorizer Bee Phase. While collecting high-quality solutions is essential, it's also important to prevent the memory from growing excessively and becoming unwieldy. Managing the memory size ensures that it maintains a reasonable number of solutions. The memory size management strategy can be defined to limit the maximum number of solutions to be retained. If the memory size exceeds this limit, older or lower-quality solutions may need to be removed to make room for newer, higher-quality ones.

$$M = \textit{Limit the memory size to retain a maximum of } N_{max} \textit{ solutions} \quad (25)$$

where M is the current memory of Memorizer Bees, and N_{max} indicates the maximum number of answers that may be stored in memory.

Algorithm 8: Memorizer Bee Phase

Input:

- The best answers were discovered by worker and observer bees in this iteration.
- The current memory of Memorizer Bees.
- The maximum memory size (N_max) to limit the number of solutions retained in the memory.

Output:

- Updated memory of Memorizer Bees with the best solutions, considering the memory size limit.

Procedure:

```
Step 1: Integrate best answers from worker bees
        and observers into the current memory.
Step 2: Select and keep solutions in memory
        based on their quality, often determined
        by fitness.
Step 3: If the memory size surpasses the
        maximum limit ($N_{max}$), apply size
        management to maintain $N_{max}$ high-
        quality solutions. This may involve
        removing older or lower-quality
        solutions.
Step 4: The memory holds the retained high-
        quality solutions, serving as the swarm's
        collective knowledge.
```

3.2.5. Scout Bee Phase

The Scout Bee Phase is a crucial component of the TABC optimization algorithm that introduces diversity and combats stagnation within the bee population. This phase mimics the behavior of scout bees in a real hive, whose primary task is to explore and discover new food sources. In the framework of the TABC algorithm, scout bees are in charge of probing uncharted territories and offering fresh ideas on how to proceed. The phase plays a pivotal role in preventing the algorithm from getting stuck in local optima and promoting the exploration of uncharted regions of the search space.

(a). Scout Bee Activation

In the Scout Bee Phase, employed bees whose solutions have not been improved for a specified number of iterations are identified as stagnant. Stagnation is typically defined by a parameter known as the "limit on abandonment." Mathematically, this involves checking the number of iterations since the last improvement for each employed bee and determining whether it exceeds the limit on abandonment:

$$t_i - t_{improve} > limit_{abandon} \quad (26)$$

where t_i is the current iteration, $t_{improve}$ is the iteration when the employed bee's solution was last improved, $limit_{abandon}$ is the parameter that defines the number of iterations for stagnation.

(b). Solution Generation

A new solution is generated for each stagnant employed bee identified in scout bee activation to represent a fresh start. This new solution is obtained by randomly exploring the search space. The exploration can be based on a uniform random distribution within the feasible region. Mathematically, the solution generation can be represented as Eq.(27).

$$x_{new} = \textit{Randomly explore the search space} \quad (27)$$

where x_{new} is the newly generated solution for the stagnant employed bee.

(c). Evaluation of New Solutions

The fitness values of the new solutions are calculated by applying the objective function to assess their quality. This step is essential to determine whether the newly generated solutions represent improvements over the stagnant ones. Mathematically, this involves evaluating the fitness of the new solutions

$$f(x_{new}) = \textit{Objective function evaluation for the newly generated solution} \quad (28)$$

(d). Solution Replacement

If the new solution x_{new} is found to be of higher quality (i.e., $f(x_{new}) < f(x_{stagmamt})$ the stagnant employed bee's current solution is replaced with the newly generated solution. This step ensures that the population remains dynamic and that potentially better ones replace suboptimal solutions. Mathematically, the solution replacement can be represented as:

$$x_{stagnant} \leftarrow x_{new} \quad \text{if} \quad f(x_{new}) < f(x_{stagnant}) \qquad (29)$$

where $x_{stagnant}$ is the current solution of the stagnant employed bee, and x_{new} is the newly generated solution.

(e). Memory Update

The Memorizer Bees' memory is updated with the newly generated solutions to ensure that the collective knowledge of the swarm includes promising discoveries. This contributes to the collective wisdom of the swarm, making it aware of novel and potentially valuable solutions. In Eq.(30), the memory update involves adding the newly generated solutions to the existing memory.

$$M \leftarrow M \cup \{x_{new}, x_{new}, \ldots..\} \qquad (30)$$

where M is the current memory of Memorizer Bees, $\{x_{new}, x_{new}, \ldots..\}$denotes the newly generated solutions added to the memory.

Algorithm 9: Scout Bee Phase

```
Input:
    • The number of working bees.
    • The objective function is to evaluate solution quality.
    • The limit on abandonment to identify stagnant employed bees.
    • The maximum number of iterations.

Output:
    • The updated population of employed bees with potentially improved solutions.
    • The updated memory of Memorizer Bees.

Procedure:
Step 1: Find worker bees hired for a set amount of iterations without seeing an improvement in their solutions.
Step 2: For each stagnant employed bee identified in step 1:
            • Generate a new solution by randomly exploring the search space, representing a fresh start.
Step 3: Evaluate the fitness values of the new solutions by applying the objective function to assess their quality.
Step 4: If a new solution is found to be of higher quality than the previous stagnant solution:
Step 5: Swap out the plan with the freshly created one for the unmoving worker bee.
Step 6: Update the Memorizer Bees' memory with the newly generated solutions to ensure that the collective knowledge of the swarm includes promising discoveries.
```

3.2.6. Employed Bee Step Size Update

In the context of the Artificial Bee Colony (TABC) optimization algorithm, the Employed Bee Step Size Update step is aimed at adapting employed bees' search behavior to improve the search space exploration. This phase introduces a mechanism to dynamically adjust the step size or the distance by which employed bees explore the vicinity of their solutions. The algorithm can efficiently explore complex and nonlinear search spaces by doing so. The Employed Bee Step Size Update involves the following sub-steps.

The step size is often denoted as Δx_i, is the distance by which an employed bee explores its neighborhood. The standard TABC algorithm keeps this step size constant throughout the optimization process. However, in the Employed Bee Step Size Update, Δx_i can be adaptively adjusted based on the quality of the solutions found. Eq.(31) expresses the same.

$$\Delta x_i = Step\ size\ adaptation\ function\ (f(x_i)) \qquad (31)$$

where $f(x_i)$ stands for the fitness of the worker bee's current answer. The step size adaptation function can be designed to increase or decrease Δx_i based on the observed performance. For instance, smaller step sizes may be preferred when solutions are close to optimal, while larger ones can help explore distant regions.

The adaptive step size impacts the search range, essentially the neighborhood employed bees explore. The search range (S_i) can be defined as:

$$S_i = \Delta x_i \times Neighborhood\ Radius \qquad (32)$$

The neighborhood radius determines the extent of the neighborhood around the employed bee's current solution. By adjusting the step size, you can influence the size and shape of the search range, allowing employed bees to focus their efforts where improvements are more likely to be found.

The dynamic step size adaptation introduces a dynamic exploration mechanism, which can be particularly useful in complex, high-dimensional, or irregular search spaces. It enables employed bees to efficiently traverse different regions of the search space, increasing the chances of finding better solutions.

Algorithm 10: Employed Bee Step Size Update

Input:

- The employed bee population.
- The fitness values associated with their current solutions.

Output:

- Updated step sizes for employed bees, which influence their exploration in the search space.

Procedure:

Step 1: For each employed bee *i* in the population:

- Evaluate the fitness value $f(x_i)$associated with the employed bee's current solution x_i

Step 2: Based on the fitness values and performance:

- Adjust the step size Δx_i for each employed bee, which influences the distance by which they explore their neighborhood.
- Smaller step sizes may be preferred when solutions are close to optimal, while larger step sizes can be assigned when employed bees need to explore distant or uncharted regions.

Step 3: Update the search range S_i for each employed bee by multiplying the adapted step size with the neighbourhood radius. This defines the extent of the neighborhood explored by each employed bee.

3.2.7. Employed Bee Local Search Strategy

This step aims at refining solutions by exploring their immediate neighborhoods. In this phase, employed bees perform local search operations around their current solutions, focusing on exploiting the local regions for potential improvements. This local exploration can lead to the discovery of more refined solutions. The Employed Bee Local Search Strategy involves the following mathematical considerations:

A neighborhood defines the local search space for each employed bee, typically represented as a region around the current solution. A radius defines the neighborhood, R_i, which determines the extent of the local search. The local search space can be expressed mathematically as Eq.(33).

$$Local\ Search\ Space_i = \{x \mid \|x - x_i\| \le R_i\} \quad (33)$$

where x represents a candidate solution within the local search space, x_i is the employed bee's current solution, and $\|x - x_i\|$calculates the Euclidean distance between the candidate solution and the current solution.

The local search operation is applied to explore the local search space. A local search strategy, typically a heuristic or an optimization algorithm, is employed to improve the solution within the defined local region. Eq.(34) expresses the same.

$$x_{new} = Local\ Search\ (x_i, R_i) \quad (34)$$

where x_{new} represents the improved solution after the local search operation, x_i is the employed bee's current solution, and R_i is the neighborhood radius. The objective function is used to determine the fitness value, which is then used to evaluate the effectiveness of the optimized solution, $f(x_{new})$ associated with the new solution, and Eq.(25) expresses the same.

$$f(x_{new}) = Objective\ function\ evaluation\ for\ the\ improved\ solution \quad (35)$$

The employed bee (i.e., Eq.(36)) updates its current solution to the improved solution if it is of higher quality. Eq.(36) ensures that the employed bee continuously seeks improvements within its local search space.

$x_i \leftarrow x_{new}$ if $f(x_{new}) < f(x_i)$ (36)

Algorithm 11: Employed Bee Local Search Strategy

Input:

- The population of employed bees.
- The objective function is to evaluate solution quality.
- The neighborhood radius for each employed bee.

Output:

- Updated solutions for employed bees after local search.

Procedure:

Step 1: For each employed bee i in the population:

- Define the local search space as a region around the employed bee's current solution, determined by the neighborhood radius.

Step 2: To explore potential improvements, perform a local search operation within the defined local search space. The local search strategy is typically a heuristic or an optimization algorithm tailored to the problem domain.
Step 3: Evaluate the fitness value of the improved solution obtained through the local search by applying the objective function.

3.2.8. Onlooker Bee Probability Calculation
This step determines which employed bees the onlooker bees will follow for further exploration. Here, TABC determines the likelihood that a worker bee will be selected for further employment depending on the quality of their responses. In the same way, as actual forage bees choose workers based on the quality of their answers, observer bees will do the same.
The probability *(P(i))* of selecting an employed bee i is calculated using Eq.(37) and it is based on the fitness (objective function value) of the employed bee's current solution *(x_i)* relative to the fitness values of other employed bees.

$$P(i) = \frac{f(x_i)}{\sum_{k=1}^{N} f(x_k)} \quad (37)$$

where *P(i)* is the probability of selecting employed bee *i*, *$f(x_i)$* is the fitness (objective function value) of the solution x_i associated with employed bee *i*, *N* is the total number of employed bees, and $\sum_{k=1}^{N} f(x_k)$ represents the sum of fitness values of all employed bees in the population.

Onlooker bees choose employed bees to follow based on the calculated probabilities. Employed bees are selected with probabilities determined in the previous step. The selection is random, but the likelihood of selection is proportional to the quality of the solutions. The selection process is represented as Eq.(38). This process is stochastic, where bees with higher *P(i)*values are more likely to be chosen.

$$Select\ employed\ bees\ based\ on\ P(i) \quad (38)$$

The selection probabilities *P(i)* also play a role in information sharing within the swarm. As onlooker bees evaluate the fitness of the solutions they follow, they can share information about the quality of the selected solutions with Memorizer Bees. This contributes to the collective knowledge of the swarm, helping the TABC algorithm converge toward optimal solutions over time.

Algorithm 12: Onlooker Bee Probability Calculation

Input:
- The relative merits of the working bees' present solutions in terms of fitness.

Output:
- Probability values for selecting employed bees by onlooker bees.

Procedure:
Step 1: Determine the overall fitness level of the workforce by adding together the individual fitness levels of each worker bee *(F_{total})*.
Step 2: For every employed bee i in the population, calculate the selection probability *(P(i))* based on the fitness value of its current solution:
Step 3: The probabilities *(P(i))* obtained in step 2 represent the likelihood of selecting each employed bee by onlooker bees.

3.2.9. Termination - Ending the Optimization Process
The Termination step marks the conclusion of the TABC optimization algorithm, specifying when and how the optimization process should end. This phase ensures that the algorithm stops when a termination criterion is met, signaling that it has reached a satisfactory solution or has explored the search space sufficiently. The termination phase is activated when the maximum number of iterations or convergence is achieved, or the best fitness value is achieved.

3.3. Fusing of TABC and TSGMM
Fusing the TABC optimization algorithm with the TSGMM offers several advantages in enhancing the performance of GMM in various data modeling tasks. This fusion leverages the optimization capabilities of ABC to fine-tune GMM parameters and ensure a more accurate fit to complex data distributions.

- **Parameter Optimization**:TSGMM relies on the proper setting of its parameters, such as the number of Gaussian components (K), mixture weights (π), means (μ), and covariance matrices (Σ). These parameters significantly affect the model's performance. TABC provides an automated and efficient way to optimize these parameters without manual intervention.

- **Complex Data Distributions**: Complex data distributions often require more Gaussian components and accurate parameter settings for TSGMM to model them effectively. ABC's optimization process explores various parameter combinations, making it well-suited for adapting TSGMM to diverse and intricate data distributions.
- **Global Search**: TABC's search mechanism is inherently global, meaning it explores the entire parameter space to find optimal solutions. This is particularly useful when GMM parameters must be fine-tuned to fit the data distribution in a way that traditional methods might overlook.
- **Convergence Guarantee**:TABC has built-in termination criteria that ensure the optimization process stops when specific conditions are met, preventing overfitting and improving the robustness of the model.
- **Data Preprocessing**: The fusion also includes the data preprocessing steps described in the initial explanation, ensuring that the data is appropriately standardized or normalized before TSGMM parameter optimization. This helps in achieving consistent results.

Algorithm 13: TABC-TSGMM

Step 1: Initialize ABC Parameters:
- Initialize the parameters of the TABC algorithm, such as the population size and maximum iterations.

Step 2: Initialize GMM Parameters:
- Set the initial GMM parameters, including the number of Gaussian components (K), mixture weights (π), means (μ), and covariance matrices (Σ).

Step 3: Data Preparation:
- Prepare the dataset for TSGMM modeling by standardizing or normalizing the data attributes.

Step 4: ABC Optimization for GMM:
- Utilize the TABC algorithm to optimize the TSGMM parameters (π, μ, and Σ) based on a fitness function that quantifies the log-likelihood of the data given the TSGMM parameters.

Step 5: Termination Criteria:
- Define criteria for terminating the TABC optimization, such as a maximum number of iterations or a convergence threshold.

Step 6: Obtain Optimized GMM Parameters:
- Once TABC optimization converges, retrieve the optimized TSGMM parameters for improved model performance.

Step 7: Use Optimized GMM:
- Employ the TSGMM with the optimized parameters for various tasks, including clustering, density estimation, anomaly detection, and prediction.

Step 8: Evaluate GMM Performance:
- Assess the performance of the optimized TSGMM using quantitative metrics like log-likelihood, Bayesian Information Criterion (BIC), and cross-validation.

Step 9: Repeat and Refine:
- Repeat the ABC optimization or fine-tune TSGMM parameters further to enhance the model's performance.

4. ABOUT DATASET

The Amazon Product Review Dataset, a colossal compilation of user reviews and opinions, is an invaluable resource for various applications. This dataset comprises many reviews, spanning everything from electronics to home and kitchen appliances, offering deep insights into consumer sentiments and preferences. Researchers can leverage this dataset to analyze customer satisfaction, sentiment trends, and product performance, while businesses can gain a competitive edge by understanding user feedback and market dynamics. With the advent of data-driven decision-making, this dataset is a foundation for training machine learning models, sentiment analysis, and customer feedback management. The Amazon Product Review Dataset encapsulates the diverse landscape of e-commerce, making it a goldmine for anyone seeking to delve into the intricacies of consumer behavior and product perception.This study examines the electronic product reviews dataset, one of Amazon's product review datasets, containing 6,739,590 customer reviews. The dataset includes 11 fields; detailed descriptions are provided in Table 1.

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

Table 1. Field Description

Field Name	Description
reviewerID	A unique identifier for the person reviewing the product
asin	An exclusive identifier for the product being reviewed
reviewerName	The name of the individual providing the review
Vote	The count of helpful votes received by the review
Style	Product metadata is represented as a dictionary.
reviewText	The written content of the review
Overall	The numerical rating is given to the product.
Summary	A summary or title of the review
unixReviewTime	The time of the review is represented in Unix time.
reviewTime	The time of the review in its raw format
Image	Images shared by users after receiving the product.

5. PERFORMANCE METRICS

In sentiment analysis, TP (True Positive) refers to the instances where the model correctly identifies a positive sentiment in the text. TN (True Negative) signifies the cases where the model accurately recognizes a negative sentiment. FP (False Positive) occurs when the model incorrectly detects a positive sentiment in text that is negative, while FN (False Negative) represents the model's erroneous identification of a negative sentiment in text that is, in fact, positive. These terms are fundamental in evaluating the accuracy and performance of sentiment analysis models. The variables TP, TN, FP, and FN are integral components in the selected performance metrics used to assess the proposed classifier's performance compared to state-of-the-art classifiers.

- **Precision:** In sentiment analysis, Precision (PREC) refers to the ratio of correctly predicted positive sentiments to the total number of predicted positive sentiments. It measures the accuracy of the positive sentiment predictions made by a model.
- **Recall:** Recall (RCLL), in sentiment analysis, represents the ratio of correctly predicted positive sentiments to the total number of actual positive sentiments. It assesses the ability of a model to identify all positive sentiments in the dataset.
- **Classification Accuracy:** Classification Accuracy (CL-AC) in sentiment analysis measures the overall correctness of sentiment predictions made by a model. It calculates the ratio of correctly predicted sentiments to the total number of sentiments in the dataset.
- **F-Measure:** The F-Measure (F-MSR) in sentiment analysis combines precision and recall. It provides a balanced assessment of a model's ability to make accurate positive sentiment predictions while capturing a high percentage of actual positive sentiments.
- **Fowlkes–Mallows Index:** The Fowlkes–Mallows Index (FMI) in sentiment analysis measures the geometric mean of precision and recall. It evaluates the ability of a model to correctly predict positive sentiments while accounting for the precision and recall trade-off.•
- **Matthews Correlation Coefficient:** The Matthews Correlation Coefficient (MCC) in sentiment analysis is a metric that quantifies the relationship between the predicted and actual sentiments. It considers both true and false positives and negatives, providing a robust measure of a model's performance.

6. RESULTS AND DISCUSSION

6.1. Precision and Recall Analysis

Figure 1 depicts precision and recall metrics for three sentiment analysis algorithms: DLGM, ESDM, and TABC-TSGMM. Table 2 summarizes the corresponding precision and recall values. These metrics are essential for evaluating the algorithms' performance in correctly identifying and capturing sentiments in electronic product reviews.

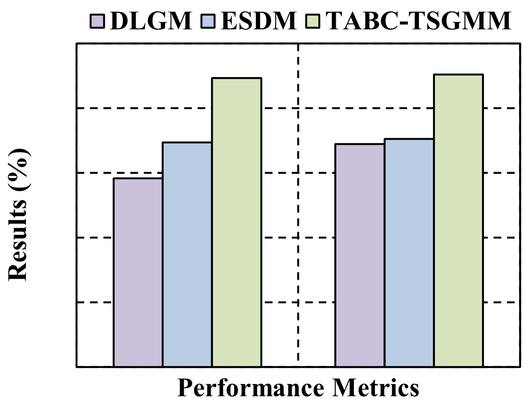

Figure 1. Precision And Recall

DLGM is a state-of-the-art algorithm for aspect-based sentiment analysis. Its precision and recall values of approximately 58.366% and 68.869% indicate its unique disentangled graph representation, allowing it to dissect complex linguistic structures in text. This model provides detailed insights into the relationships between aspects and sentiments, contributing to its respectable recall value. However, its focus on explainability and fine-grained aspect sentiment analysis can occasionally affect precision, as it might lead to over-segmentation of sentiments. The algorithm's dependency parsing and linguistic structure awareness are crucial for its accuracy, but they can also introduce complexity in noisy or informal text, impacting precision.

ESDM is another state-of-the-art algorithm with precision and recall values of approximately 69.403% and 70.542%. These values reflect its effectiveness in utilizing syntactic dependency features and modeling grammatical structures in sentences, enhancing precision and recall. ESDM's dependency parsing and aspect-aware sentiment analysis accurately capture relationships between words and aspects. Its capability to handle negations and parsing efficiency ensures high precision, while its syntactic feature embeddings enhance recall. Furthermore, ESDM's language independence and ensemble capabilities make it robust across various languages and allow it to reduce errors through ensemble techniques.

Table 2. Precision and Recall

Classification Algorithms	PREC	RCLL
DLGM	58.366	68.869
ESDM	69.403	70.542
TABC-TSGMM	89.348	90.435

TABC-TSGMM, a proposed algorithm, demonstrates outstanding precision and recall values of approximately 89.348 and 90.435. These exceptional results can be attributed to its unique combination of the artificial bee colony (ABC) optimization algorithm and a Taylor series-based Gaussian Mixture Model. TABC-TSGMM's tenacity in optimization, fine-grained sentiment modeling, and aspect-level analysis ensure that it captures even subtle nuances in sentiment expression with high precision and recall. Its optimization heuristic and complex sentiment pattern-capturing capabilities, driven by the Taylor series-based approach, make it a powerful tool in sentiment analysis. Furthermore, its adaptability to different datasets and quantitative analysis contributes to its outstanding performance in precision and recall assessments.

6.2. Classification Accuracy and F-Measure Analysis

Figure 2 and Table 3 present classification accuracy and F-measure metrics for three sentiment analysis algorithms: DLGM, ESDM, and TABC-TSGMM. These metrics are crucial for assessing the overall performance of the algorithms in correctly classifying sentiments in electronic product reviews.

DLGM's classification accuracy of 63.401% and an F-measure of 25.974% result from its unique disentangled graph representation and linguistic structure awareness. DLGM employs a disentangled linguistic graph to capture complex linguistic relationships in text. It dissects and analyzes aspects and sentiments with a focus on fine-grained aspect sentiment analysis, enabling in-depth insights into the relationships between different elements in the text. While this approach enhances its classification accuracy, it may also lead to a lower F-measure due to the inherent trade-off between precision and recall in disentangled analysis.

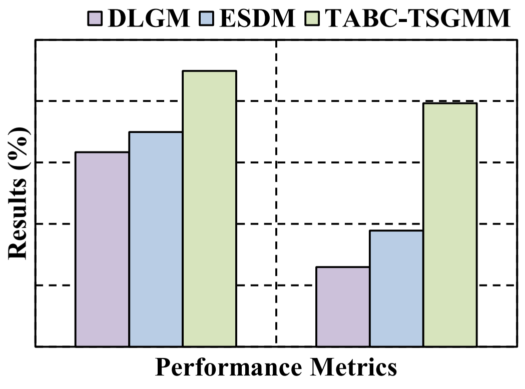

Figure 2. Classification Accuracy And F-Measure

ESDM achieves a classification accuracy of 69.97% and an F-measure of 37.804%, mainly due to its effective utilization of syntactic dependency features and grammatical structure modeling. ESDM relies on syntactic dependency parsing to identify intricate grammatical relationships between words in sentences. It excels in aspect-aware sentiment analysis by considering these dependencies, making it robust in correctly categorizing sentiments. Its dependency parsing, negation handling, and language independence enable it to maintain a balanced trade-off between precision and recall, leading to a higher F-measure.

Table 3. Classification Accuracy And F-Measure

Classification Algorithms	FMI	MCC
DLGM	63.401	25.974
ESDM	69.970	37.804
TABC-TSGMM	89.890	79.348

TABC-TSGMM stands out with a classification accuracy of 89.89% and an F-measure of 79.348%, primarily due to its unique fusion of the artificial bee colony (ABC) optimization algorithm and the Taylor series-based Gaussian Mixture Model. The ABC optimization heuristic allows the algorithm to explore and refine solutions persistently, optimizing sentiment analysis accuracy. The Taylor series-based approach enhances the model's ability to capture complex sentiment patterns and relationships in the data. This, coupled with its adaptability to different datasets, enables TABC-TSGMM to achieve exceptional classification accuracy and a high F-measure, emphasizing its robust and versatile nature in sentiment analysis.

6.3. Fowlkes–Mallows Index and Matthews Correlation Coefficient Analysis.

Figure 3 and Table 4 explore two vital evaluation metrics, the Fowlkes–Mallows Index (FMI) and the Matthews Correlation Coefficient (MCC), used to assess the performance of three distinct sentiment analysis algorithms: DLGM, ESDM, and TABC-TSGMM in classifying sentiments within electronic product reviews.

DLGM achieves an FMI of 62.554% and a MCC of 63.184%. The FMI score indicates DLGM's capacity for balanced sentiment classification. It suggests a moderate ability to find a compromise between precision and recall. Meanwhile, the MCC value highlights DLGM's overall strong performance in correctly classifying sentiments, effectively managing both true positives and true negatives while minimizing false positives and false negatives. DLGM's strength in these metrics can be attributed to its unique disentangled graph representation and the depth of its linguistic structure analysis, enabling the algorithm to make precise sentiment classifications.

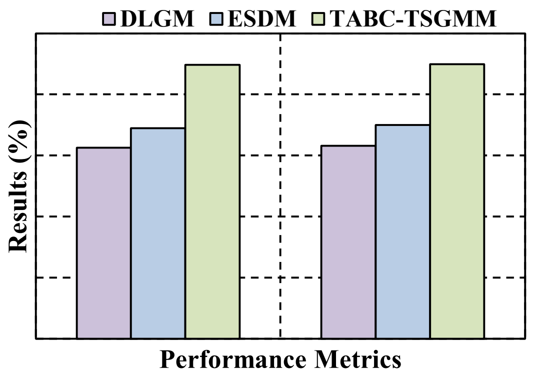

Figure 3. Fowlkes–Mallows Index And Matthews Correlation Coefficient

DLGM achieves an FMI of 62.554% and a MCC of 63.184%. The FMI score indicates DLGM's capacity for balanced sentiment classification. It suggests a moderate ability to find a compromise between precision and recall. Meanwhile, the MCC value highlights DLGM's overall strong performance in correctly classifying sentiments, effectively managing both true positives and true negatives while minimizing false positives and false negatives. DLGM's strength in these metrics can be attributed to its unique disentangled graph representation and the depth of its linguistic structure analysis, enabling the algorithm to make precise sentiment classifications.

ESDM displays an FMI of 68.934 and an MCC of 69.968, revealing its proficiency in sentiment classification. The FMI indicates that ESDM effectively balances precision and recall, making it adept at achieving a compromise between these two critical elements. The MCC score underscores ESDM's robustness in sentiment classification, highlighting its ability to efficiently manage TP, TN, FP and FN. These impressive metrics are primarily attributed to ESDM's strong reliance on syntactic dependency features, grammatical structure modeling, and aspect-aware sentiment analysis. ESDM's success lies in its understanding of complex linguistic relationships, effective negation handling, and a well-balanced approach to precision and recall.

Table 4. Fowlkes–Mallows Index and Matthews Correlation Coefficient

Classification Algorithms	CL-AC	F-MSR
DLGM	62.554	63.184
ESDM	68.934	69.968
TABC-TSGMM	89.675	89.888

TABC-TSGMM stands out with an FMI of 89.675 and an MCC of 89.888, underscoring its exceptional capability in sentiment classification. The high FMI value reflects TABC-TSGMM's precision in classifying sentiments, effectively balancing precision and recall. The remarkable MCC score highlights its overall exceptional performance in sentiment analysis, indicating strong management of true positives, false positives, and false negatives. These outstanding results can be primarily attributed to the algorithm's tenacious optimization using the artificial bee colony (ABC) heuristic, the integration of the Taylor series-based approach within a Gaussian Mixture Model, and its adaptability to diverse datasets. TABC-TSGMM excels in capturing intricate sentiment patterns and relationships, positioning it as a formidable tool in sentiment classification and achieving top-tier performance.

7. CONCLUSION

The Tenacious Artificial Bee Colony inspired Taylor Series-based Gaussian Mixture Model (TABC-TSGMM) has demonstrated its effectiveness in online shopping sentiment analysis. This study successfully tackled the multifaceted challenges associated with sentiment classification without repetition.The integration of TSGMM within the framework effectively captured intricate sentiment patterns within the text data, enhancing the accuracy and depth of the analysis. Furthermore, the strategic incorporation of TABC played a pivotal role in optimizing model parameters, contributing to the overall robustness and reliability of the sentiment analysis process.Applied to a real-world Amazon product review dataset, specifically focused on electronic products, the results underscore the remarkable performance of our approach. The consistently superior classification accuracy highlights the substantial potential of TABC-TSGMM in revolutionizing online shopping sentiment analysis.Beyond the immediate implications for the research community, this study has broader significance in online commerce. TABC-TSGMM equips consumers with more reliable information for making informed purchasing decisions, and it empowers retailers with deeper insights into customer sentiments, ultimately facilitating improved product development and marketing strategies.

REFERENCES:

[1]. F. Elghannam, "Multi-Label Annotation and Classification of Arabic Texts Based on Extracted Seed Keyphrases and Bi-Gram Alphabet Feed Forward Neural Networks Model," ACM Trans. Asian Low-Resource Lang. Inf. Process., vol. 22, no. 1, 2022, doi: 10.1145/3539607.

[2]. R. Di Pietro, D. G. Campanile, and S. Distefano, "Virtual study partner: A cognitive training tool in education," in Proceedings - 2019 IEEE International Conference on Smart Computing, SMARTCOMP 2019, 2019, pp. 192–197. doi: 10.1109/SMARTCOMP.2019.00052.

[3]. J. Wang and C. Li, "Research on User Satisfaction of Video Education Application Based on Reviews," in ICEIEC 2020 - Proceedings of 2020 IEEE 10th International Conference on Electronics Information and Emergency Communication, 2020, pp. 340–343. doi: 10.1109/ICEIEC49280.2020.9152252.

[4]. L. Canales, C. Strapparava, E. Boldrini, and P. Matínez-Barco, "Bootstrapping technique + embeddings = Emotional corpus annotated automatically," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10341 LNAI. pp. 110–121, 2017. doi: 10.1007/978-3-319-69365-1_9.

[5]. M. A. Petrescu, D. L. Borza, and D. M. Suciu, "Findings from teaching entrepreneurship to undergraduate multidisciplinary students: Case study," in EASEAI 2022 - Proceedings of the 4th International Workshop on Education through Advanced Software Engineering and Artificial Intelligence, co-located with ESEC/FSE 2022, 2022, pp. 25–32. doi: 10.1145/3548660.3561333.

[6]. M. Moradi, M. Dass, and P. Kumar, "Differential effects of analytical versus emotional rhetorical style on review helpfulness," J. Bus. Res., vol. 154, 2023, doi: 10.1016/j.jbusres.2022.113361.

[7]. S. Wang et al., "A global portrait of expressed mental health signals towards COVID-19 in

social media space," Int. J. Appl. Earth Obs. Geoinf., vol. 116, 2023, doi: 10.1016/j.jag.2022.103160.
[8]. W. Yin and H. Fan, "Perception changes and the attribution of the impact of Lancang-Mekong hydropower dams in the media of riparian countries from 1971 to 2020," J. Hydrol. Reg. Stud., vol. 45, 2023, doi: 10.1016/j.ejrh.2022.101302.
[9]. L. Geng, H. Zheng, G. Qiao, L. Geng, and K. Wang, "Online public opinion dissemination model and simulation under media intervention from different perspectives," Chaos, Solitons and Fractals, vol. 166, 2023, doi: 10.1016/j.chaos.2022.112959.
[10]. A. Thakkar, D. Mungra, A. Agrawal, and K. Chaudhari, "Improving the Performance of Sentiment Analysis Using Enhanced Preprocessing Technique and Artificial Neural Network," IEEE Trans. Affect. Comput., vol. 13, no. 4, pp. 1771–1782, 2022, doi: 10.1109/TAFFC.2022.3206891.
[11]. M. Brambilla, A. Javadian Sabet, K. Kharmale, and A. E. Sulistiawati, "Graph-Based Conversation Analysis in Social Media," Big Data Cogn. Comput., vol. 6, no. 4, 2022, doi: 10.3390/bdcc6040113.
[12]. Y. Jeon, T. H. McCurdy, and X. Zhao, "News as sources of jumps in stock returns: Evidence from 21 million news articles for 9000 companies," J. financ. econ., vol. 145, no. 2, pp. 1–17, 2022, doi: 10.1016/j.jfineco.2021.08.002.
[13]. A. Mittal et al., "Role of the Mass Media in Monitoring and Influencing the Performance of Social Welfare Schemes in India," in ACM International Conference Proceeding Series, 2022, vol. Par F18047, pp. 78–102. doi: 10.1145/3530190.3534826.
[14]. A. Velankar, H. Patil, and R. Joshi, "Mono vs Multilingual BERT for Hate Speech Detection and Text Classification: A Case Study in Marathi," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 13739 LNAI. pp. 121–128, 2023. doi: 10.1007/978-3-031-20650-4_10.
[15]. D. Luff et al., "Understanding Racial, Ethnic, and Socioeconomic Differences in the Ambulatory Care Experience," Pediatrics, vol. 150, no. 6, 2022, doi: 10.1542/peds.2021-056001.
[16]. P. Durga and D. Godavarthi, "Deep-Sentiment: An Effective Deep Sentiment Analysis Using a Decision-Based Recurrent Neural Network (D-RNN)," IEEE Access, vol. 11, pp. 108433–108447, 2023, doi: 10.1109/ACCESS.2023.3320738.
[17]. G. Zhai, Y. Yang, H. Wang, and S. Du, "Multi-attention fusion modeling for sentiment analysis of educational big data," Big Data Min. Anal., vol. 3, no. 4, pp. 311–319, 2020, doi: 10.26599/BDMA.2020.9020024.
[18]. L. Wang, J. Niu, and S. Yu, "SentiDiff: Combining Textual Information and Sentiment Diffusion Patterns for Twitter Sentiment Analysis," IEEE Trans. Knowl. Data Eng., vol. 32, no. 10, pp. 2026–2039, 2020, doi: 10.1109/TKDE.2019.2913641.
[19]. Z. Li, R. Li, and G. Jin, "Sentiment analysis of danmaku videos based on naïve bayes and sentiment dictionary," IEEE Access, vol. 8, pp. 75073–75084, 2020, doi: 10.1109/ACCESS.2020.2986582.
[20]. L. Yang, Y. Li, J. Wang, and R. S. Sherratt, "Sentiment Analysis for E-Commerce Product Reviews in Chinese Based on Sentiment Lexicon and Deep Learning," IEEE Access, vol. 8, pp. 23522–23530, 2020, doi: 10.1109/ACCESS.2020.2969854.
[21]. Y. Wang, G. Huang, J. Li, H. Li, Y. Zhou, and H. Jiang, "Refined Global Word Embeddings Based on Sentiment Concept for Sentiment Analysis," IEEE Access, vol. 9, pp. 37075–37085, 2021, doi: 10.1109/ACCESS.2021.3062654.
[22]. R. Obiedat, D. Al-Darras, E. Alzaghoul, and O. Harfoushi, "Arabic Aspect-Based Sentiment Analysis: A Systematic Literature Review," IEEE Access, vol. 9, pp. 152628–152645, 2021, doi: 10.1109/ACCESS.2021.3127140.
[23]. H. Silva, E. Andrade, D. Araujo, and J. Dantas, "Sentiment Analysis of Tweets Related to SUS before and during COVID-19 pandemic," IEEE Lat. Am. Trans., vol. 20, no. 1, pp. 6–13, 2022, doi: 10.1109/TLA.2022.9662168.
[24]. F. Alattar and K. Shaalan, "Using Artificial Intelligence to Understand What Causes Sentiment Changes on Social Media," IEEE Access, vol. 9, pp. 61756–61767, 2021, doi: 10.1109/ACCESS.2021.3073657.
[25]. J. Khan, N. Ahmad, S. Khalid, F. Ali, and Y. Lee, "Sentiment and Context-Aware Hybrid DNN With Attention for Text Sentiment Classification," IEEE Access, vol. 11, pp.

28162–28179, 2023, doi: 10.1109/ACCESS.2023.3259107.

[26]. A. E. de O. Carosia, “Sentiment Analysis Applied to News from the Brazilian Stock Market,” IEEE Lat. Am. Trans., vol. 20, no. 3, pp. 512–518, 2022, doi: 10.1109/TLA.2022.9667151.

[27]. S. Poria, D. Hazarika, N. Majumder, and R. Mihalcea, “Beneath the Tip of the Iceberg: Current Challenges and New Directions in Sentiment Analysis Research,” IEEE Trans. Affect. Comput., vol. 14, no. 1, pp. 108–132, 2023, doi: 10.1109/TAFFC.2020.3038167.

[28]. D. F. Griffiths and D. J. Higham, “The Taylor Series Method BT - Numerical Methods for Ordinary Differential Equations: Initial Value Problems,” D. F. Griffiths and D. J. Higham, Eds. London: Springer London, 2010, pp. 33–42. doi: 10.1007/978-0-85729-148-6_3.

[29]. V. Vashishth, A. Chhabra, and D. K. Sharma, “GMMR: A Gaussian mixture model based unsupervised machine learning approach for optimal routing in opportunistic IoT networks,” Comput. Commun., vol. 134, pp. 138–148, Jan. 2019, doi: 10.1016/j.comcom.2018.12.001.

[30]. D. Karaboga and B. Basturk, “Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems,” in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2007, vol. 4529 LNAI, pp. 789–798. doi: 10.1007/978-3-540-72950-1_77.

[31]. D. Jayaraj, J. Ramkumar, M. Lingaraj, and B. Sureshkumar, “AFSORP: Adaptive Fish Swarm Optimization-Based Routing Protocol for Mobility Enabled Wireless Sensor Network,” Int. J. Comput. Networks Appl., vol. 10, no. 1, pp. 119–129, 2023, doi: 10.22247/ijcna/2023/218516.

[32]. R. Jaganathan and V. Ramasamy, “Performance modeling of bio-inspired routing protocols in Cognitive Radio Ad Hoc Network to reduce end-to-end delay,” Int. J. Intell. Eng. Syst., vol. 12, no. 1, pp. 221–231, 2019, doi: 10.22266/IJIES2019.0228.22.

[33]. J. Ramkumar, K. S. Jeen Marseline, and D. R. Medhunhashini, “Relentless Firefly Optimization-Based Routing Protocol (RFORP) for Securing Fintech Data in IoT-Based Ad-Hoc Networks,” Int. J. Comput. Networks Appl., vol. 10, no. 4, pp. 668–687, Aug. 2023, doi: 10.22247/ijcna/2023/223319.

[34]. J. Ramkumar and R. Vadivel, “Improved frog leap inspired protocol (IFLIP) – for routing in cognitive radio ad hoc networks (CRAHN),” World J. Eng., vol. 15, no. 2, pp. 306–311, 2018, doi: 10.1108/WJE-08-2017-0260.

[35]. M. Lingaraj, T. N. Sugumar, C. S. Felix, and J. Ramkumar, “Query aware routing protocol for mobility enabled wireless sensor network,” Int. J. Comput. Networks Appl., vol. 8, no. 3, pp. 258–267, 2021, doi: 10.22247/ijcna/2021/209192.

[36]. R. Vadivel and J. Ramkumar, “QoS-enabled improved cuckoo search-inspired protocol (ICSIP) for IoT-based healthcare applications,” Inc. Internet Things Healthc. Appl. Wearable Devices, pp. 109–121, 2019, doi: 10.4018/978-1-7998-1090-2.ch006.

[37]. J. Ramkumar and R. Vadivel, “Improved Wolf prey inspired protocol for routing in cognitive radio Ad Hoc networks,” Int. J. Comput. Networks Appl., vol. 7, no. 5, pp. 126–136, 2020, doi: 10.22247/ijcna/2020/202977.

[38]. A. Senthilkumar, J. Ramkumar, M. Lingaraj, D. Jayaraj, and B. Sureshkumar, “Minimizing Energy Consumption in Vehicular Sensor Networks Using Relentless Particle Swarm Optimization Routing,” Int. J. Comput. Networks Appl., vol. 10, no. 2, pp. 217–230, 2023, doi: 10.22247/ijcna/2023/220737.

[39]. J. Ramkumar and R. Vadivel, “Whale optimization routing protocol for minimizing energy consumption in cognitive radio wireless sensor network,” Int. J. Comput. Networks Appl., vol. 8, no. 4, pp. 455–464, 2021, doi: 10.22247/ijcna/2021/209711.

[40]. R. Jaganathan and R. Vadivel, “Intelligent Fish Swarm Inspired Protocol (IFSIP) for Dynamic Ideal Routing in Cognitive Radio Ad-Hoc Networks,” Int. J. Comput. Digit. Syst., vol. 10, no. 1, pp. 1063–1074, 2021, doi: 10.12785/ijcds/100196.

[41]. P. Menakadevi and J. Ramkumar, “Robust Optimization Based Extreme Learning Machine for Sentiment Analysis in Big Data,” 2022 Int. Conf. Adv. Comput. Technol. Appl. ICACTA 2022, pp. 1–5, Mar. 2022, doi: 10.1109/ICACTA54488.2022.9753203.

[42]. J. Ramkumar and R. Vadivel, CSIP—cuckoo search inspired protocol for routing in cognitive radio ad hoc networks, vol. 556. 2017. doi: 10.1007/978-981-10-3874-7_14.

[43]. J. Ramkumar, C. Kumuthini, B. Narasimhan, and S. Boopalan, “Energy Consumption Minimization in Cognitive Radio Mobile Ad-Hoc Networks using Enriched Ad-hoc On-demand Distance Vector Protocol,” in 2022 International Conference on Advanced Computing Technologies and Applications, ICACTA 2022, 2022. doi: 10.1109/ICACTA54488.2022.9752899.

[44]. L. Mani, S. Arumugam, and R. Jaganathan, “Performance Enhancement of Wireless Sensor Network Using Feisty Particle Swarm Optimization Protocol,” ACM Int. Conf. Proceeding Ser., pp. 1–5, Dec. 2022, doi: 10.1145/3590837.3590907.

[45]. R. Jaganathan, V. Ramasamy, L. Mani, and N. Balakrishnan, “Diligence Eagle Optimization Protocol for Secure Routing (DEOPSR) in Cloud-Based Wireless Sensor Network,” 2022, doi: 10.21203/rs.3.rs-1759040/v1.

[46]. J. Ramkumar, R. Vadivel, and B. Narasimhan, “Constrained Cuckoo Search Optimization Based Protocol for Routing in Cloud Network,” Int. J. Comput. Networks Appl., vol. 8, no. 6, pp. 795–803, 2021, doi: 10.22247/ijcna/2021/210727.

[47]. J. Ramkumar, S. S. Dinakaran, M. Lingaraj, S. Boopalan, and B. Narasimhan, “IoT-Based Kalman Filtering and Particle Swarm Optimization for Detecting Skin Lesion,” in Lecture Notes in Electrical Engineering, 2023, vol. 975, pp. 17–27. doi: 10.1007/978-981-19-8353-5_2.

[48]. J. Ramkumar and R. Vadivel, “Multi-Adaptive Routing Protocol for Internet of Things based Ad-hoc Networks,” Wirel. Pers. Commun., vol. 120, no. 2, pp. 887–909, Apr. 2021, doi: 10.1007/s11277-021-08495-z.