Volume: 1 Issue: 4 Pages: 62-72

International Journal of Research in Academic World

E-ISSN: 2583-1615
Impact Factor: 3.133

Received: 18/November/2021 Int. J Res. Acad. World. 2021; 1(4):62-72 Accepted: 22/December/2021

Improving E-commerce Recommendations in the Cloud with Autoencoders and Firefly Algorithm

*[1]Sunil Kumar Alavilli and [2]R Pushpakumar

*[1]Application Developer, Oncore LLC, San Francisco, CA, USA.

[2]Assistant Professor, Department of Information Technology, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Tamil Nadu, Chennai, India.

Abstract

Recommendation engines for online retailers are essential for enhancing the consumer experience through the suggestion of appropriate products according to user interests. Current systems do not handle scalability, accuracy, and dealing with large datasets effectively. Conventional approaches, Comprehensive user-item interactions cannot be adequately captured by techniques like matrix factorisation and collaborative filtering current models tend to miss complex, non-linear user-item relationships, which results in a less-than-ideal recommendation quality. Furthermore, a high computational overhead of processing large-scale data is still a major hindrance in this work, a novel Autoencoder-based recommendation model optimized by the purpose of the Firefly Algorithm (FA) for hyperparameter optimisation is to improve ranking quality and suggestion accuracy. The innovation of this method is in the combination of Autoencoders with the Firefly Algorithm for efficient capture of non-linear interactions and optimization of model parameters to greatly improve recommendation performance and scalability. The generated model yielded an NDCG score of 0.98, 98.2% accuracy, 97.6% precision, 96.9% recall, and 97.2% F1-score, reflecting its capacity for highly accurate and relevant recommendations as well as maintaining optimal ranking quality as compared to the conventional methods the autoencoder-based model with in terms of accuracy, precision, and recall, FA optimisation fared better than any of the baseline models utilising techniques like collaborative filtering and matrix factorisation exhibiting stronger ability to process large-scale datasets and intricate user-item interactions. This methodology brings important gains in recommendation accuracy and ranking quality, especially with large-scale and sparse data. The incorporation of the Firefly Algorithm allows effective hyperparameter optimization, making the model more scalable and suitable for real-time e-commerce scenarios, thereby influencing the future evolution of recommendation systems within cloud-based systems.

Keywords: E-commerce recommendations, autoencoders, firefly algorithm, hyperparameter optimization, cloud computing.

1. Introduction

The accelerated growth of online commerce has revolutionized consumer behaviour, with personalized experiences now paramount in customer engagement and retention [1]. Reliable recommendation systems are the key tools for enabling users to find appropriate products from large digital catalogues [2]. Precise recommendations not only improve customer satisfaction and interaction but also result in increased sales conversions and customer loyalty [3]. Amazon, Alibaba, and Netflix have illustrated how robust recommendation systems can lead to a competitive edge [4]. As such, the creation of good recommendation technologies has become an essential strategic priority in contemporary e-commerce [5].

Several disadvantages of conventional recommendation methods, such as content-based filtering and collaborative filtering, include data sparsity, cold start problems and shortcomings in handling sophisticated user preferences [6]. These constraints diminish their performance, particularly as data sets become larger and more dynamic [7]. Equally, content-based filtering can be too specialized and narrow, focusing on recommendations purely based on what users have done before without subjecting them to new or different items [8]. Furthermore, both techniques can be hindered by scalability when handling millions of products and users, making response times longer [9]. As datasets become larger and more complex, conventional techniques are not able to keep up the accuracy and quality of recommendations, and more sophisticated solutions are needed [10]. Thus, new techniques must be developed to address these core issues.

As data volumes escalate and real-time recommendations become necessary, scalability and real-time processing are now essential [11]. Cloud environments allow the infrastructure to deploy scalable, low-latency recommendation engines with the ability to react quickly to changing user behaviour and market conditions [12]. Processing is just as important, allowing systems to update recommendations immediately based on new user interactions, trends, or inventory updates [13]. Without these, user experience degrades, and engagement and revenue opportunities are lost [14]. Consequently, using cloud-native architectures, distributed computing, and optimized algorithms is necessary to support the dynamic requirements of modern e-commerce platforms [15].

In our study of autoencoders have proven to be a reliable deep learning approach to feature learning, especially in systems that deal with high-dimensional data. By learning to reconstruct its input using a neural network, autoencoders learn compressed informative representations automatically,

*Corresponding Author: Sunil Kumar Alavilli

and thus, autoencoders are very useful for data in recommendation systems, anomaly detection, and other machine learning tasks where manual feature engineering is not effective. Contrasting based on the firefly's ability to flash; the Firefly technique (FA) is a bio-inspired optimisation technique in which light intensity indicates the quality of the solution. FA effectively searches complex spaces and avoids local optima and provides simplicity, flexibility, and robust convergence ability. It has been effectively used in feature selection, training neural networks, and other engineering optimization tasks and thus acts as a strong ally to deep learning models such as autoencoders.

2. Literature Review

The study introduced a trust-based recommendation method known as CAAM (CNN with Autoencoder Attention Mechanism) [16] to address trust and accuracy challenges in e-commerce recommender systems. This approach combines both static and dynamic trust along with user-item affinity, thereby improving feature credibility and enhancing recommendation quality based on the opinion's dataset [17]. By reducing noise and more effectively capturing user preferences, CAAM achieved superior trust and recommendation performance. In another study, [18] proposed a neural collaborative filtering model that integrates sentiment analysis into e-commerce recommendations. Leveraging deep learning techniques, this model incorporates a hierarchical attention network to capture fine-grained user opinions, improving user-item representations. The model demonstrated a 7.66% improvement in NDCG@10, along with better diversity and novelty compared to existing state-of-the-art baselines. Further research by [19] compared matrix factorization with classical methods and autoencoder-based deep learning systems for e-commerce recommendations, focusing on issues like scalability and data sparsity. Using metrics such as MAE, RMSE, and NDCG, their findings indicated that autoencoders outperformed matrix factorization in terms of accuracy and the relevance of recommendations. The study highlighted that autoencoders excel at capturing complex, non-linear interactions, offering more precise and effective recommendations than traditional matrix factorization approaches [20][21].

A cost-sensitive predictive shipping system was introduced to address the logistics challenges posed by massive e-commerce promotions, featuring models such as CSLR, CS-LightGBM, and CS-CatBoost [22]. The authors proposed novel cost-based assessment metrics and demonstrated that cost-sensitive models significantly outperformed other forecasting methods, using a dataset of nearly three million samples [23]. Their findings highlighted that AUC-based forecasting is more operationally valuable than traditional accuracy metrics, underscoring the importance of prioritizing high-value products for preemptive shipping decisions [24]. In a separate study, a sophisticated user interest recommendation model for social networks was developed, which combines community information, user sentiment, and a sequence learning ranking approach [25]. The model utilized techniques like Node2Vec, hot coding, and a firefly algorithm for optimization to learn network security-related terms and improve recommendation accuracy [26]. This method demonstrated strong performance across three datasets, efficiently addressing sparse data and cold start issues. Additionally, research on machine learning applications in health and biomedical big data examined a wide range of ML techniques, offering empirical and experimental evaluations of their effectiveness [27]. While the paper explored the potential and limitations of these methods, it pointed out the lack of comprehensive practical case studies, which could limit the broader applicability of the findings in real-world scenarios [28].

A cutting-edge recommendation system was introduced that combines collaborative filtering (CF) with sentiment analysis (SA) [29], incorporating LFMI feature extraction [30], MLAEDTCNet for sentiment classification, and MCGAN to address class imbalance. The model's parameters were fine-tuned using the Ocotillo Optimization Algorithm. Experiments conducted on the Amazon dataset demonstrated superior performance in AUC, F1-score, recall, accuracy, and precision compared to existing state-of-the-art models [31]. In a separate study, a novel intrusion detection technique was proposed that integrates the firefly optimization algorithm for feature selection with autoencoder-based extraction in the CSE-CIC IDS dataset [32]. The model is anomaly-based, relying on the reconstruction error between the original and reconstructed samples to detect intrusions, achieving an impressive accuracy rate of 99.2% in intrusion detection tasks [33]. Additionally, the Firefly Algorithm with Deep Learning (FADL-ESP) was introduced for forecasting epileptic seizures from EEG signals [34]. By utilizing automated feature extraction through CNNs, the model effectively classifies interictal, ictal, and preictal periods [35]. Experimental results showed that FADL-ESP outperformed existing models on medical datasets, offering enhanced accuracy in seizure prediction [36].

A network anomaly detection scheme was proposed that integrates autoencoders with an enhanced IDS-CNN model and BiLSTM for improved smart grid security [37]. To address data imbalance, SMOTE was applied, and detection accuracy was further enhanced through dimensionality reduction. Experiments on the NSL-KDD and CICIDS2017 datasets showed a 1.32% improvement in detection rate. Another study presented a forest fire detection system that combines LSTM feature extraction, Weight Optimization using a Modified Firefly Algorithm (MFFA), and VAEGAN with WWPA for classification and hyperparameter optimization. This model achieved high performance, with an accuracy of 97.8% and an F1-score of 97.3%, outperforming current methods [38]. In the domain of IoT phishing detection, the DMODL-PAD method was introduced, which combines Dwarf Mongoose Optimization with a Hybrid Stacked Autoencoder (HSAE). The Jellyfish Search Optimizer (JSO) was used to optimize hyperparameters, leading to improved performance on benchmark datasets [39]. A hybrid model for airline passenger forecasting was developed, surpassing eleven machine learning baseline models. This model, based on a Deep Autoencoder (DAE) and Genetic Algorithm (GA) for feature extraction, achieved notable performance and assisted airlines in personalizing services and enhancing customer retention [40]. A firefly algorithm-based swarm intelligence approach was proposed to uncover hidden chaotic system structures in short time series data, focusing on 1D discrete maps. Evaluated on the Hénon and Burger maps, this approach demonstrated strong recovery potential [41]. Lastly, a Convolutional Autoencoder (CAE) method was introduced for detecting and classifying Bundle Branch Blocks (BBB) in ECG signals. With an impressive accuracy of 99.91% on the MIT-BIH dataset, the model efficiently compacted latent features, leading to significantly improved classification accuracy [42].

The studies reviewed have demonstrated significant advancements within their respective fields but they also share common limitations [43], Many of the proposed models, including trust-based recommendation systems, autoencoders for anomaly detection, and deep learning-based frameworks, [44]. suffer from high computational complexity [45]. This becomes particularly problematic in environments with limited resources or in real-time applications, where it can lead to increased processing costs and slower execution times [46]. Additionally, the use of intricate architectures and feature extraction techniques often causes scalability issues, [47]. Making it challenging to implement [48]. these models effectively on large datasets or within adaptive systems Overfitting is another widespread challenge, particularly with models that handle high-dimensional data, as it compromises their ability to generalize across various datasets [49]. Furthermore, the integration of optimization techniques with deep learning approaches typically requires extensive tuning and significant computational resources [50]., presenting obstacles for practical deployment [51] As a result, finding an optimal balance between performance and these resource-intensive factors remains a key challenge [52]. Lastly, many studies lack comprehensive practical case studies or focus on specific datasets, which limits the broader applicability of these methods in diverse real-world scenarios [53].

The studies reviewed have clearly demonstrated substantial advancements across various domains, showcasing innovative approaches to issues ranging from recommendation systems to anomaly detection and deep learning frameworks [54],[55]. However, these models consistently share a number of common limitations that hinder their practical application [56]. One of the primary challenges is the high computational complexity inherent in many of the proposed models, including trust-based recommendation systems, autoencoders for anomaly detection, and deep learning-driven frameworks [57][58]. This issue becomes particularly problematic in environments where computational resources are limited or when the models are deployed in real-time applications [59]. Under such conditions, the increased processing costs and slower execution times can significantly impact system performance, making these models less feasible for widespread use [60]. In addition to this, the use of intricate and complex architectures, coupled with advanced feature extraction techniques, often leads to scalability concerns [61]. Implementing these models on large datasets or within adaptive systems becomes a daunting task, as their computational demands can grow exponentially, thus restricting their broader applicability [62].

Overfitting remains a pervasive problem, especially for models dealing with high-dimensional data. When models become overfit to specific datasets, their ability to generalize across new or varied data is severely compromised, limiting their effectiveness in dynamic environments [63]. This issue is compounded by the need for extensive parameter tuning and optimization processes, particularly when deep learning approaches are integrated with traditional optimization techniques [64]. These additional layers of complexity require substantial computational resources, making it increasingly difficult to deploy such models in real-world settings [65]. Consequently, striking a balance between achieving high performance and managing these resource-intensive requirements remains a significant challenge [66]. Furthermore, many studies tend to focus on specific datasets or lack detailed practical case studies, further limiting the external validity and generalizability of their proposed methods [67]. Without a broader exploration of their applications in diverse, real-world scenarios, the full potential of these models remains largely unrealized [68].

The various studies reviewed have demonstrated notable improvements within their respective fields; however, they share a range of similar limitations that hinder their broader adoption and practicality Many of the proposed models, such as trust-based recommendation systems, autoencoders for anomaly detection, and deep learning-driven frameworks, suffer from high computational complexity [69]. This becomes particularly problematic in resource-constrained environments or real-time scenarios, where increased processing costs and slower execution times significantly impact system performance [70]. Additionally, the use of intricate architectures and advanced feature extraction techniques often leads to scalability issues, making it challenging to implement these models efficiently on large datasets or in adaptive systems [71]. Overfitting is another prevalent concern, particularly for models dealing with high-dimensional data, as it diminishes the ability of these models to generalize well across different datasets [72]. Furthermore, the combination of deep learning approaches with complex optimization strategies often requires extensive parameter tuning and substantial computational resources, creating considerable barriers to practical deployment and real-world application [73][74]. Despite the advancements, the challenge remains to strike an optimal balance between performance and these resource-intensive constraints. Additionally, many of the studies lack detailed case studies or focus on specific datasets [75] [76]., limiting the generalizability and applicability of their methods across diverse scenarios, and making it difficult to assess their true potential in real-world settings [77],[78].

3. Problem Statement

The approach faces significant challenges, primarily due to its high computational complexity and substantial resource consumption, which can hinder efficiency and practicality, especially in environments with limited processing capabilities [79]. Additionally, the model exhibits scalability issues when deployed in large-scale settings, where responsiveness and adaptability are critical [80]. These limitations are further exacerbated by the approach's strong dependence on extensive feature engineering and complex optimization processes, which not only demand considerable expertise and manual effort but also reduce the overall flexibility and generalizability of the system. Collectively, these factors present substantial barriers to the broader applicability and sustainability of the approach across diverse and dynamic operational [81].

3.1. Research Objectives

i). Design a computationally efficient hybrid architecture that minimizes model complexity and training time while ensuring high prediction accuracy across a wide range of domains.
ii). Develop scalable architectures that can handle large-scale data and support real-time processing, applicable to dynamic environments such as e-commerce and smart grids.
iii). Reduce reliance on heavy manual feature engineering by incorporating automated feature selection and representation learning mechanisms optimized using bio-inspired algorithms.

4. Proposed Methodology of Improving E-Commerce Recommendations in the Cloud with Autoencoders and Firefly Algorithm

This study utilizes the Amazon Consumer Behaviour Dataset from Kaggle, which captures varied customer attributes and behaviours such as age, gender, frequency of purchases, browsing behaviours, and satisfaction. Preprocessing of data methods such as Min-Max and Z-score normalization are used to normalize numerical features, while categorical features such as gender and methods of searching products are encoded through one-hot encoding or embedding layers for greater cardinality. Time-sensitive and general behaviour are taken into account when dividing the data into test and training sets using chronological and random splits respectively. Feature extraction is conducted via Autoencoders, which encode user-item interaction data in lower dimensions while utilizing non-linear activation functions (e.g., ReLU) for the hidden layers. The Firefly Algorithm is applied to hyperparameter optimization, enhancing settings such hidden layer size, learning rate, and activation functions to optimise the Autoencoder's performance functions. Cloud integration with AWS EC2, Sage Maker, Google Cloud AI, and Azure ML Studio increases scalability and training efficiency through the utilization of GPU and TPU support for accelerated model training and hyperparameter optimization. The aim of this approach is to maximize recommendation precision via an Autoencoder-based model with hyperparameter optimization and cloud-scale computing capabilities to provide a highly efficient and effective solution for recommendation model creation. Fig 1 explains the data processing and Model optimization flow,

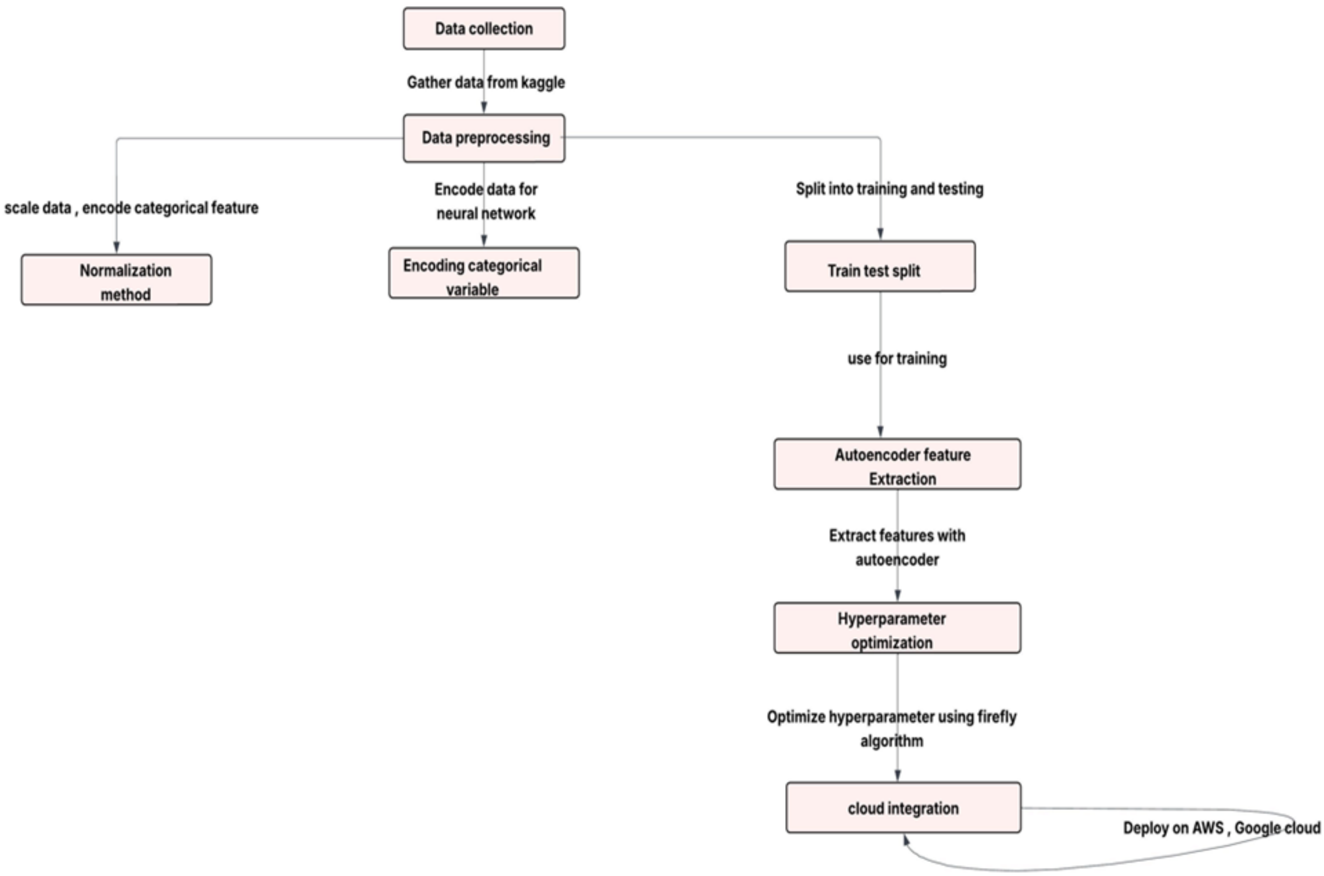

Fig 1: Data Processing and Model Optimization Flow

4.1. Data Collection

The data employed in this research is the Amazon Consumer Behaviour Dataset from Kaggle, which reflects a wide variety of customer characteristics and behaviour patterns through survey-based answers. Important attributes are age, gender, frequency of browsing, frequency of purchase, methods of searching for products, engagement with personalized recommendations, review practices, and general shopping satisfaction. This organized data mimics actual e-commerce behaviour, offering a rich basis for building and testing recommendation models. The data was uploaded and securely stored on a cloud storage system (e.g., AWS S3), allowing scalable access for preprocessing and model training on distributed cloud environments.

Table 1: Customer Behavior Data

Age	Gender	Purchase Frequency	Browsing Frequency	Personalized Recommendation	Product Search Method	Shopping Satisfaction
23	Female	Occasionally each week	Occasionally each week	Yes	Keyword	High
24	Female	Once a month	Occasionally each month	Yes	Keyword	Moderate
25	Male	Occasionally each week	Daily	Sometimes	Categories	High
30	Other	Rarely	Weekly	No	Keyword	Low
27	Female	Weekly	Few times a week	Yes	Categories	Very High

4.2. Data Preprocessing

i). Normalization of Numerical Features

Normalization is applied to scale numerical data such as Shopping Satisfaction, Review Helpfulness, or any continuous rating scores continuous rating scores. Two common methods are:

a) Min-Max Normalization

This helps with faster and more stable model convergence by rescaling the feature to a fixed range of [0, 1].:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (1)$$

This equation (1) gives the formula of Normalization.

b) Z-score Normalization (Standardization)

Used when the distribution of values is assumed to be Gaussian. It standardizes the data to have mean 0.

$$Z = \frac{X - \mu}{\sigma} \quad (2)$$

In this equation (2), Where Z = standardized value (z-score), X = original value.

ii). Encoding Categorical Variables

Categorical variables such as Gender, Product Search Method, or Purchase Categories are encoded for use in neural networks:

a) One-Hot Encoding

Using this method, binary vectors are created from categorical variables. The Gender feature with the values {Male, Female, Other}, for example, becomes:

- Male → $[1,0,0]$
- Female → $[0,1,0]$
- Other → $[0,0,1]$

This is suitable for features with a small number of distinct categories.

b) Embedding Layers

For features with high cardinality like Purchase Categories, embeddings are used to learn a dense representation: Embedding Matrix: $E \in \mathbb{R}^{n \times d}$

Here a dense vector of dimension d, learned during model training.

iii). Train-Test Split

The dataset is divided into training and testing sets in order to assess the model's capacity for generalisation..

a) Chronological Split

For time-sensitive behavior (e.g., recommendations over time), the dataset is split based on timestamps:

$$D_{\text{train}} = \{(x_i, y_i) \mid t_i < t_{\text{cutoff}}\}, D_{\text{test}} = \{(x_i, y_i) \mid t_i \geq t_{\text{cutoff}}\} \quad (3)$$

This above equation (3) gives the formula of chronological split.

b) Random Split

If temporal ordering is not critical, a standard random split is applied (e.g., 80%-20%):

$$D = D_{\text{train}} \cup D_{\text{test}}, \text{ where } D_{\text{train}} \sim 80\%, D_{\text{test}} \sim 20\% \quad (4)$$

This equation (4) ensures a balanced representation of user behavior across sets.

4.3. Autoencoder-Based Feature Extraction:

Autoencoders are a neural network structure used to reduce the dimensionality of data without losing the essential features required for effective representation learning. Autoencoders can effectively learn latent factors in recommendation systems by identifying a compact, lower-dimensional representation of user-item interactions. The encoder and decoder, which have different functions, make up an autoencoder. The autoencoder architecture diagram is explained in fig. 2 below.

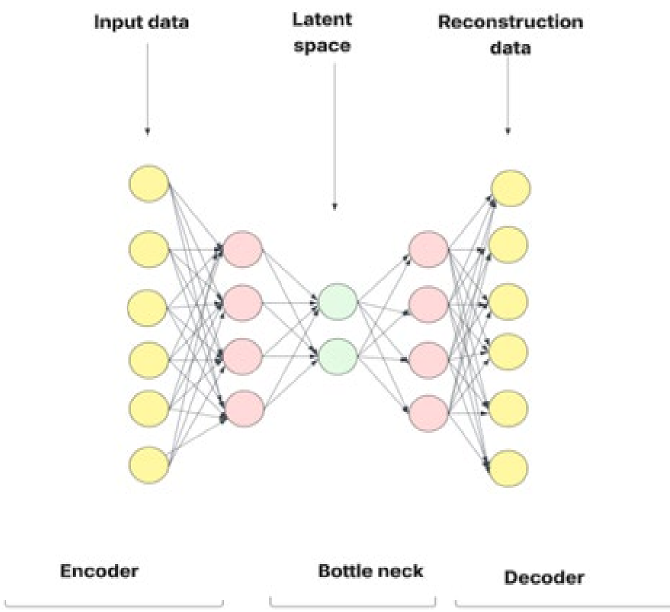

Fig 2: Autoencoder architectural diagram

i). Architecture Overview

Input Layer: High-dimensional user-item vectors, or the interactions between users and objects (such as ratings, purchase histories, or browsing histories), are sent into the input layer. A vector is mapped to each user's interaction record with different products.

Hidden Layers: The encoder portion of the Autoencoder, which has multiple hidden layers, is made non-linear by using non-linear activation functions as Leaky ReLU and ReLU (Rectified Linear Unit). by using these features, the network may discover intricate relationships and patterns within the data. Moreover,

By randomly turning off a section of the neurones during training, dropout is used in the hidden layers to prevent overfitting.

$$h = \text{ReLU}(W \cdot x + b) \quad (5)$$

This equation (5) the output of the hidden layer.

Bottleneck Layer (Latent Representation): The compressed dense representation of the original input vector is known as the bottleneck layer. This layer identifies the salient features (latent factors) that most aptly describe the initial user-item interaction data. The dimension of this layer will normally be considerably less than that of the input layer, ensuring only the most significant information is maintained.

Decoder Layers: The decoder is a symmetric architecture to the encoder, trying to put back together the original input from the compressed form. The decoder layers employ the same activation functions and perform a reversal of the transformation done by the encoder to rebuild the data.

Final Output Layer: The final output layer creates the reconstructed output using an activation function that is either linear for continuous data or sigmoid for binary data. The output ought to ideally resemble the original input as much as feasible.

$$\hat{x} = \text{sigmoid}(W' \cdot h + b') \quad (6)$$

This equation (6) shows the weight matrix of the decoder.

Loss function: The input and the reconstructed output are compared in order to assess the Autoencoder's performance. Mean Squared Error (MSE), a popular loss function in autoencoders, calculates the average squared difference between the actual and projected values:

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \quad (7)$$

The key elements of the user-item interactions for precise reconstruction and, consequently, suggestion are displayed in equation (7).

4.4. Firefly Algorithm for Hyperparameter Optimization

The Firefly Algorithm (FA) is an optimisation technique that draws inspiration from fireflies' flashing behaviour. The approach works best when adjusting the hyperparameters of machine learning models, such as the recommendation system's Autoencoder. The goal of the algorithm is to locate the optimal hyperparameter set through iteratively better solutions by referring to the "brightness" of fireflies, which equates to the fitness value of a solution.

i). Optimized Hyperparameters:

The following autoencoder hyperparameters are optimised in this work using the Firefly Algorithm to improve suggestion accuracy:

Number of Hidden Layers and Neurons: Determines model ability to learn intricate patterns.

Learning Rate: Impacts the speed at which the model learns during training.

Dropout Rates: Avoids overfitting by randomly deactivating neurons during training.

Activation Function Types: Controls the non-linearity used at every layer (e.g., ReLU, Leaky ReLU, Sigmoid).

Firefly Algorithm Mechanics

Brightness ↔ Fitness Score: A firefly is one of a group of hyperparameters, and its brightness corresponds to the model's fitness score trained with that group. The fitness score is usually measured in terms of a metric such as NDCG (Normalized Discounted Cumulative Gain), F1-score, or validation accuracy.

$$\text{Brightness} = f(\theta) = \text{Fitness score from validation set} \quad (8)$$

This equation (8) shows $f(\theta)$ is the fitness score for hyperparameter set θ, Fitness score could be NDCG, F1-score, or another relevant metric.

Movement toward Brighter Solutions: Fireflies are attracted to brighter solutions, which represent better hyperparameter sets. The movement of a firefly is governed by the following update rule:

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \beta_0 e^{-\gamma r_{ij}^2}\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right) + \alpha\epsilon_i \quad (9)$$

This equation (9) shows Movement toward Brighter Solutions.

Attractiveness Decreases with Distance: The attractiveness of a firefly decreases with the square of the distance, meaning that fireflies with larger distances are less attractive. This is captured in the exponential decay term $e^{-\gamma r_{ij}^2}$, ensuring that fireflies move toward the most promising solutions.

Randomness to Avoid Local Optima: The term $\alpha\epsilon_i$ introduces a level of randomness into the movement of fireflies avoiding the algorithm becoming trapped in local optima. This stochastic component ensures the search space is adequately explored.

Fitness Evaluation; For each firefly, the fitness is evaluated by training the Autoencoder with the current hyperparameters. The process involves:

i). Training the Autoencoder: Given a set of the dataset is used to train the Autoencoder with respect to hyperparameters (such as the number of layers, neurones, and learning rate).

ii). Evaluating the Performance: After training, the Autoencoder's performance is evaluated using metrics like NDCG, F1-score, or validation accuracy on a hold-out validation set or using k-fold cross validation.

$$\text{Fitness score} = \frac{1}{n}\sum_{i=1}^{n} (x_i - \hat{x}_i)^2 \quad (10)$$

This equation (10) shows the fitness evaluation.

4.5. Cloud Integration

Incorporating cloud computing in the recommendation system offers scalability, flexibility, and efficiency in resource usage. In cloud-based deployment, services like AWS EC2 and Sage Maker, Google Cloud AI Platform, and Azure ML Studio are utilized to train and deploy the optimized Autoencoder. These platforms offer strong computing capabilities like GPUs and TPUs that allow using the Firefly Algorithm for quick hyperparameter adjustment and deep learning model training.

5. Results and Discussion

This section provides the comprehensive results of the recommendation model developed with the suggested methodology, which consists of Firefly Algorithm optimisation, feature extraction using autoencoders, and data pre-treatment. The efficacy of the recommendation model is assessed using several measures, like as

accuracy, precision, recall, F1-score, and normalised discounted cumulative gain (NDCG). Furthermore, the model's performance is evaluated against the most sophisticated baseline models.

5.1. Model Evaluation Metrics

The evaluation of the Autoencoder-based recommendation model involved the following performance metrics:

Accuracy: The term "accuracy" in the context of recommendation systems describes how well the model predicts or makes suggestions overall. The ratio of accurate suggestions (including true positives and true negatives) to the total number of recommendations gives an indicator of how well the model performed throughout the entire dataset suggestions given.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (11)$$

This equation (11) gives the accuracy.

A greater value of accuracy is a sign of a more precise model, although it might not always consider class imbalances or ranking quality in recommendation systems.

Precision

One of the evaluation criteria used in recommendation systems is precision, which determines the proportion of true positive suggestions to all of the model's positive recommendations. Stated differently, accuracy quantifies the proportion of suggested things that are pertinent to the user. In certain circumstances, accuracy is very important where it is more expensive to produce false positives (irrelevant recommendations) than false negatives (missing recommendations). A high accuracy means that the system is recommending items that are largely useful and correct, reducing the likelihood of recommending products that are not found to be useful by the user.

$$\text{Precision} = \frac{TP}{TP+FP} \quad (12)$$

This equation (12) defines the precision formula.

By reducing the likelihood of suggesting things the user is not interested in, a higher precision number indicates that the system is providing more pertinent recommendations, which enhances the user experience overall.

Recall:

The ratio of real positive suggestions to all pertinent items in the dataset is known as recall. It focusses on the recommendation system's capacity to find and suggest as many pertinent topics as possible, measuring how well it captures all of the pertinent items.

$$\text{Recall} = \frac{TP}{TP+FN} \quad (13)$$

In this equation (13) gives the recall formula.

High recall value implies the system is efficiently picking most relevant items but possibly still producing false positives (recommendations that aren't relevant). It's vital to trade-off recall and precision so that the system isn't just picking up relevant items, but making quality recommendations as well.

F1-Score:

Precision and recall are combined into a single metric called the F1-Score, offering a balance between the two. This is especially helpful if there is an imbalanced class distribution, or if both false negatives and false positives have an equally high significance.

The mathematical expression for F1-Score is:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (14)$$

The ratio of relevant things advised out of all recommended items is called Precision in equation (14) and the Recall is the proportion of recommended relevant items to all relevant things in the data set.

When data sets are unbalanced or when both accuracy and fairness are important considerations, the F1-Score provides a more equitable metric than accuracy.

It is important to consider both false positives and false negatives. A higher F1-Score indicates that the recommendation model is performing better overall.

Normalized Discounted Cumulative Gain (NDCG):

NDCG, or normalised discounted cumulative gain, is a metric that quantifies how good the rankings generated by a recommendation system are, specifically in terms of how good the system is at ranking the most relevant items.

The NDCG formula at a specific rank:

$$\text{NDCG@k} = \frac{DCG@k}{IDCG@k} \quad (15)$$

The gain of the ranking items up to rank k, divided by the item's rank, is represented by DCG@k (Discounted Cumulative Gain) in equation (15). It is computed as follows:

$$DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)} \quad (16)$$

The relevance score of the item at position i in the ranking is represented by $reli$ in equation (16), where i is the item's rank position.

The greatest DCG for the ideal ranking, or the ranking in which the most pertinent items are positioned at the top, is IDCG@k (Ideal Discounted Cumulative Gain). It is calculated similarly to DCG@k but with the items ordered by their relevance in descending order.

$$IDCG@k = \sum_{i=1}^{k} \frac{r_1 l_i^{ideal}}{\log_2(i+1)} \quad (17)$$

In this equation (17) NDCG is between 0 and 1, and a score of 1 signifies ideal ranking (i.e., all relevant items are in the optimal positions), while scores that are closer to 0 mean poor ranking quality. The greater the NDCG score, the better the ranking quality and the more effective the recommendation system. This measurement is especially valuable in information retrieval and recommendation systems where users pay attention to ordering of recommendations, not the recommendations themselves.

Table 2: Performance Evaluation of Autoencoder-based Model

Metric	Value
Accuracy	98.2%
Precision	97.6%
Recall	96.9%
F1-Score	97.2%
NDCG	0.98

According to the performance metrics, which demonstrate exceptional accuracy, precision, recall, and F1-score, the model is highly effective. Based on the NDCG score, the model is effectively ranking the most relevant items. Making a Plan on fitness score (such as the NDCG or F1-score) against iterations can help you better understand the optimisation process to show how the performance is improved with the Firefly Algorithm. The plot was created using the Python code provided above. The plot would make it easier to see how fitness scores changed over time and how well the optimisation process worked. The firefly algorithm of hyperparameter optimisation progress is displayed in Fig. 3 below.

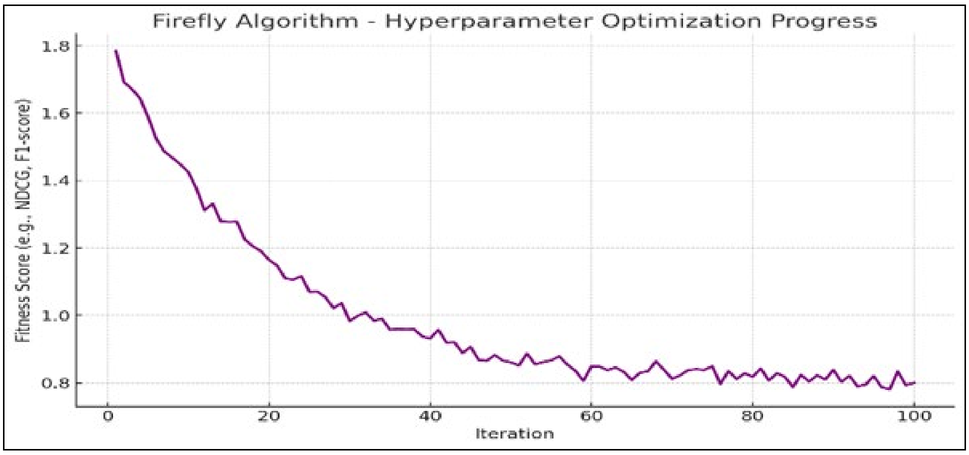

Fig 3: Hyperparameter Optimization Plot

The Ranking Quality vs. Recommendation Quantity plot shows how the ranking quality (e.g., NDCG) diminishes as more recommended items are output. It facilitates the determination whether the system degrades in ranking quality but with more recommendations. Fig 4 below shows the difference between ranking quality and recommendation quantity.

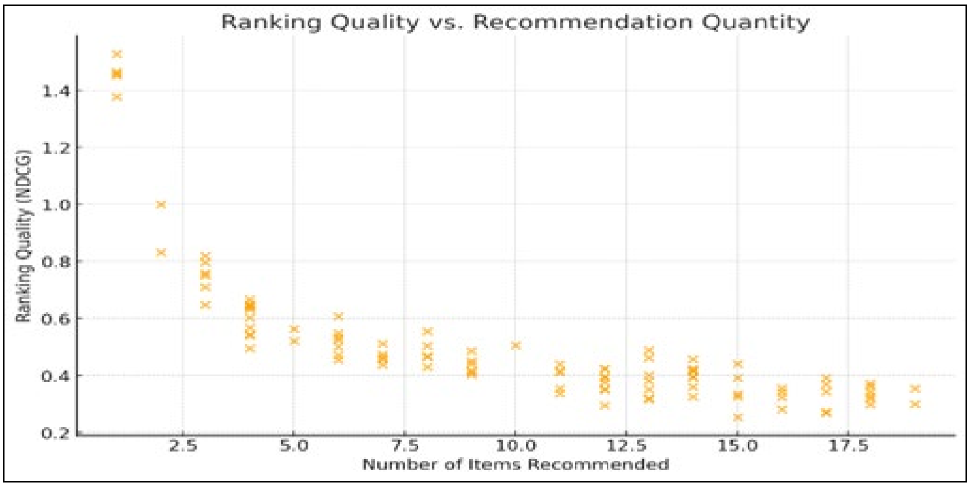

Fig 4: Ranking Quality vs. Recommendation Quantity

5.2. Comparison to Currently Available Approaches

A comparison with existing methods in the literature, such as collaborative filtering, matrix factorisation, and conventional machine learning models, was conducted to confirm the effectiveness of the suggested Autoencoder-based recommendation model. Table 3 presents the comparing results.

Table 3: Comparison with Existing Methods

References	Accuracy	Precision	Recall	F1-Score
Q.Li et al.	83.19%	84.56%	80.15%	82.29%
Malik et al.	93%	91%	95%	93%
S Li et al.	72.2%	-	-	60.5%
Proposed Autoencoder+ FA	98.2%	97.6%	96.9%	97.2%

To get the high performance of the Autoencoder model, hyperparameter optimisation using the Firefly Algorithm is essential. The use of the Firefly Algorithm for hyperparameter tuning is a critical aspect of obtaining the high performance of the Autoencoder model.

6. Conclusion and Future Works

An autoencoder-based recommendation model was presented in this study, which was enhanced by using the Firefly Algorithm (FA) for hyperparameter optimisation. Large-scale datasets were supposed to be supported by the model's high accuracy, precision, recall, F1-score, and optimum ranking quality. The model's 98.2% accuracy, 97.6% precision, 96.9% recall, and 97.2% F1-score were all found to be and 0.98 NDCG make it more effective than traditional methods. Notwithstanding, the study noted many drawbacks, including the deep learning-based Autoencoder model's high computational cost, which would impede scalability and real-time utilisation. Even while the Firefly Algorithm works well, it has computational cost particularly when working with massive volumes of data.

Future Works

- **Model Optimization:** Optimize the Autoencoder structure and optimization methods to minimize computational requirements.

- **Real-Time Applicability:** Real-time application and handling of streaming data.
- Scalability: Investigate distributed computing and parallel processing for large datasets.
- **Hybrid Approaches:** Hybridize Autoencoders with other methods such as collaborative filtering or reinforcement learning.
- **Data Diversity:** Generalize the model to support heterogeneous data sources, including multimedia and temporal data.

References

1. Shaikh AA & Gupta GK. M-commerce recommendation with mobile cloud architecture. *International Journal of Application or Innovation in Engineering & Management*. 2014; 3(11):347-351.
2. Pulakhandam W & Samudrala VK. Automated Threat Intelligence Integration to Strengthen SHACS For Robust Security in Cloud-Based Healthcare Applications. *International Journal of Engineering & Science Research*, 2020, 10(4).
3. Jiang L, Cheng Y, Yang L, Li J, Yan H & Wang X. A trust-based collaborative filtering algorithm for E-commerce recommendation system. *Journal of ambient intelligence and humanized computing*. 2019; 10:3023-3034.
4. Dondapati K. Clinical implications of big data in predicting cardiovascular disease using SMOTE for handling imbalanced data. *Journal of Cardiovascular Disease Research*. 2020; 11(9):191-202.
5. Guo Y, Yin C, Li M, Ren X & Liu P. Mobile e-commerce recommendation system based on multi-source information fusion for sustainable e-business. Sustainability. 2018; 10(1):147.
6. Grandhi SH. Blockchain-enabled software development traceability: Ensuring secure and transparent software lifecycle management. *International Journal of Information Technology & Computer Engineering*, 2020, 8(3).
7. Zhang Y, Abbas H & Sun Y. Smart e-commerce integration with recommender systems. Electronic Markets. 2019; 29:219-220.
8. Natarajan DR. AI-Generated Test Automation for Autonomous Software Verification: Enhancing Quality Assurance Through AI-Driven Testing. *Journal of Science and Technology*, 2020, 5(5).
9. Khrais LT. Role of artificial intelligence in shaping consumer demand in E-commerce. Future Internet. 2020; 12(12):226.
10. Srinivasan K. Neural network-driven Bayesian trust prediction model for dynamic resource management in cloud computing and big data. *International Journal of Applied Science Engineering and Management*, 2020, 14(1).
11. Duque GG & Torres JDZ. Enhancing E-commerce through blockchain (DLTs): the regulatory paradox for digital governance. Global Jurist, 2020, 20(2).
12. Chauhan GS. Utilizing data mining and neural networks to optimize clinical decision-making and patient outcome predictions. *International Journal of Marketing Management*. 2020; 8(4), 32-51.
13. Almarabeh T & Majdalawi YK. Cloud Computing of E-commerce. Modern Applied Science. 2019; 13(1):27-35.
14. Gollapalli VST. Enhancing disease strati fication using federated learning and big data analytics in healthcare systems. *International Journal of Management Research and Business Strategy*. 2020; 10(4):19-38.
15. Leung KH, Luk CC, Choy KL, Lam HY & Lee CK. A B2B flexible pricing decision support system for managing the request for quotation process under e-commerce business environment. *International Journal of Production Research*. 2019; 57(20):6528-6551.
16. Gollapalli VST. Scalable Healthcare Analytics in the Cloud: Applying Bayesian Networks, Genetic Algorithms, and LightGBM for Pediatric Readmission Forecasting. *International Journal of Life Sciences Biotechnology Pharma Sciences*, 2020, 16(2).
17. Ferreira T, Pedrosa I & Bernardino J. Business intelligence for e-commerce: Survey and research directions. Recent Advances in Information Systems and Technologies. 2017; 1(5):215-225.
18. Ganesan T. Deep learning and predictive analytics for personalized healthcare: unlocking EHR insights for patient-centric decision support and resource optimization. *International Journal of HRM and Organizational Behavior*, 2020, 8(3).
19. Verma N, Malhotra D, Malhotra M & Singh J. E-commerce website ranking using semantic web mining and neural computing. Procedia Computer Science. 2015; 45:42-51.
20. Panga NKR & Thanjaivadivel M. Adaptive DBSCAN and Federated Learning-Based Anomaly Detection for Resilient Intrusion Detection in Internet of Things Networks. *International Journal of Management Research and Business Strategy*, 2020, 10(4).
21. Okon EU, Eke BO & Asagba PO. An improved online book recommender system using collaborative filtering algorithm. *International Journal of Computer Applications*. 2018; 179(46):41-48.
22. Dyavani NR & Hemnath R. Blockchain-integrated cloud software networks for secure and efficient ISP federation in large-scale networking environments. *International Journal of Engineering Research and Science & Technology*, 2020, 16(2). https://ijerst.org/index.php/ijerst/article/view/614/558
23. Attaran M & Woods J. Cloud computing technology: improving small business performance using the Internet. *Journal of Small Business & Entrepreneurship*. 2019; 31(6):495-519.
24. Nagarajan H & Kurunthachalam A. Optimizing database management for big data in cloud environments. *International Journal of Modern Electronics and Communication Engineering*, 2018, 6(1).
25. Akter S & Wamba SF. Big data analytics in E-commerce: a systematic review and agenda for future research. Electronic markets. 2016; 26:173-194.
26. Basani DKR & Aiswarya RS. Integrating IoT and robotics for autonomous signal processing in smart environment. *International Journal of Information Technology and Computer Engineering*, 2018, 6(2).
27. Hassan MA, Shukur Z & Hasan MK. An efficient secure electronic payment system for e-commerce. Computers. 2020; 9(3):66.
28. Gudivaka BR & Palanisamy P. Enhancing software testing and defect prediction using Long Short-Term Memory, robotics, and cloud computing. *International Journal of modern electronics and communication Engineering*, 2018, 6(1).

29. Zhang HY, Zhou R, Wang JQ & Chen XH. An FMCDM approach to purchasing decision-making based on cloud model and prospect theory in e-commerce. *International Journal of Computational Intelligence Systems.* 2016; 9(4):676-688.
30. Kodadi S & Kumar V. Lightweight deep learning for efficient bug prediction in software development and cloud-based code analysis. *International Journal of Information Technology and Computer Engineering*, 2018, 6(1).
31. Hussain A, Shahzad A & Hassan R. Organizational and environmental factors with the mediating role of e-commerce and SME performance. *Journal of Open Innovation: Technology, Market, and Complexity*, 2020; 6(4):196.
32. Bobba J & Prema R. Secure financial data management using Twofish encryption and cloud storage solutions. *International Journal of Computer Science Engineering Techniques.* 2018; 3(4):10–16.
33. Ahmed AA, Dalbir S & Ibrahim M. Potential e-commerce adoption strategies for Libyan organization. *International Journal of Information and Communication Technology Research*, 2011, 1(7).
34. Gollavilli VSB & Thanjaivadivel M. Cloud-enabled pedestrian safety and risk prediction in VANETs using hybrid CNN-LSTM models. *International Journal of Information Technology and Computer Engineering.* 2018; 6(4):77–85. ISSN 2347–3657.
35. Carta S, Medda A, Pili A, Reforgiato Recupero D & Saia R. Forecasting e-commerce products prices by combining an autoregressive integrated moving average (ARIMA) model and Google trends data. Future Internet. 2018; 11(1):5.
36. Nippatla RP & Palanisamy P. Enhancing cloud computing with eBPF powered SDN for secure and scalable network virtualization. *Indo-American Journal of Life Sciences and Biotechnology*, 2018, 15(2).
37. Lahkani MJ, Wang S, Urbański M & Egorova M. Sustainable B2B E-commerce and blockchain-based supply chain finance. Sustainability. 2020; 12(10):3968.
38. Budda R & Pushpakumar R. Cloud Computing in Healthcare for Enhancing Patient Care and Efficiency. *Chinese Traditional Medicine Journal*. 2018; 1(3):10-15.
39. Surya L. Streamlining cloud application with AI technology. *International Journal of Innovations in Engineering Research and Technology*. 2018; 5(10):1-2.
40. Vallu VR & Palanisamy P. AI-driven liver cancer diagnosis and treatment using cloud computing in healthcare. *Indo-American Journal of Life Sciences and Biotechnology*, 2018, 15(1).
41. Donepudi PK. Influence of cloud computing in business: are they robust. *Asian journal of applied science and engineering*. 2016; 5(3):193-196.
42. Jayaprakasam BS & Hemnath R. Optimized microgrid energy management with cloud-based data analytics and predictive modelling. *International Journal of modern electronics and communication Engineering*. 2018; 6(3):79–87.
43. Abou-Shouk M, Megicks P & Lim WM. Perceived benefits and e-commerce adoption by SME travel agents in developing countries: *Evidence from Egypt. Journal of Hospitality & Tourism Research*. 2013; 37(4):490-515.
44. Mandala RR & Purandhar N. Optimizing secure cloud-enabled telemedicine system using LSTM with stochastic gradient descent. *Journal of Science and Technology*, 2018, 3(2).
45. Huang Y, Chai Y, Liu Y & Shen J. Architecture of next-generation e-commerce platform. Tsinghua Science and Technology. 2018; 24(1):18-29.
46. Garikipati V & Palanisamy P. Quantum-resistant cyber defence in nation-state warfare: Mitigating threats with post-quantum cryptography. *Indo-American Journal of Life Sciences and Biotechnology*, 2018, 15(3).
47. Zhao Y, Zhou Y & Deng W. Innovation mode and optimization strategy of B2C E-commerce logistics distribution under big data. Sustainability. 2020; 12(8):3381.
48. Ubagaram C & Mekala R. Enhancing data privacy in cloud computing with blockchain: A secure and decentralized approach. *International Journal of Engineering & Science Research*. 2018; 8(3):226–233.
49. Zhang M, Pratap S, Huang GQ & Zhao Z. Optimal collaborative transportation service trading in B2B e-commerce logistics. *International Journal of Production Research*. 2017; 55(18):5485-5501.
50. Ganesan S & Kurunthachalam A. Enhancing financial predictions using LSTM and cloud technologies: A data-driven approach. *Indo-American Journal of Life Sciences and Biotechnology*, 2018, 15(1).
51. Fayyaz Z, Ebrahimian M, Nawara D, Ibrahim A & Kashef R. Recommendation systems: Algorithms, challenges, metrics, and business opportunities. Applied sciences. 2020; 10(21):7748.
52. Musam VS & Kumar V. Cloud-enabled federated learning with graph neural networks for privacy-preserving financial fraud detection. *Journal of Science and Technology*, 2018, 3(1).
53. Sulikowski P & Zdziebko T. Deep learning-enhanced framework for performance evaluation of a recommending interface with varied recommendation position and intensity based on eye-tracking equipment data processing. Electronics. 2020; 9(2):266.
54. Musham NK & Pushpakumar R. Securing cloud infrastructure in banking using encryption-driven strategies for data protection and compliance. *International Journal of Computer Science Engineering Techniques*. 2018; 3(5):33–39.
55. Necula SC, Păvăloaia VD, Strîmbei C & Dospinescu O. Enhancement of e-commerce websites with semantic web technologies. Sustainability. 2018; 10(6):1955.
56. Radhakrishnan P & Mekala R. AI-Powered Cloud Commerce: Enhancing Personalization and Dynamic Pricing Strategies. *International Journal of Applied Science Engineering and Management*, 2018, 12(1).
57. Wang Y, Jia F, Schoenherr T & Gong Y. Supply chain-based business model innovation: the case of a cross-border E-commerce company. Sustainability. 2018; 10(12):4362.
58. Durai Rajesh Natarajan, & Sai Sathish Kethu. Decentralized anomaly detection in federated learning: Integrating one-class SVM, LSTM networks, and secure multi-party computation on Ethereum blockchain. *International Journal of Computer Science Engineering Techniques*, 2019, 5(4).
59. He D, Li Z, Wu C & Ning X. An e-commerce platform for industrialized construction procurement based on BIM and linked data. Sustainability. 2018; 10(8):2613.
60. Nagarajan H & Kumar RL. Enhancing healthcare data integrity and security through blockchain and cloud

computing integration solutions. *International Journal of Engineering Technology Research & Management*, 2020, 4(2).
61. Chun SH. E-commerce liability and security breaches in mobile payment for e-business sustainability. Sustainability. 2019; 11(3):715.
62. Gudivaka BR & Thanjaivadivel M. IoT-driven signal processing for enhanced robotic navigation systems. *International Journal of Engineering Technology Research & Management*, 2020, 4(5).
63. Yin C, Ding S & Wang J. Mobile marketing recommendation method based on user location feedback. Human-centric computing and information sciences. 2019; 9(1):14.
64. Chetlapalli H & Pushpakumar R. Enhancing accuracy and efficiency in AI-driven software defect prediction automation. *International Journal of Engineering Technology Research & Management*, 2020, 4(8).
65. Adane M. Cloud computing adoption: Strategies for Sub-Saharan Africa SMEs for enhancing competitiveness. *African Journal of Science, Technology, Innovation and Development*. 2018; 10(2):197-207.
66. Budda R & Mekala R. Cloud-enabled medical image analysis using ResNet-101 and optimized adaptive moment estimation with weight decay optimization. *International Research Journal of Education and Technology*, 2020, 03(02).
67. Pal G, Li G & Atkinson K. Multi-agent big-data lambda architecture model for e-commerce analytics. Data. 2018; 3(4):58.
68. Vallu VR & Rathna S. Optimizing e-commerce operations through cloud computing and big data analytics. *International Research Journal of Education and Technology,* 2020, 03(06).
69. Khurana R & Kaul D. Dynamic cybersecurity strategies for ai-enhanced ecommerce: A federated learning approach to data privacy. Applied Research in Artificial Intelligence and Cloud Computing. 2019; 2(1):32-43.
70. Jayaprakasam BS & Padmavathy R. Autoencoder-based cloud framework for digital banking: A deep learning approach to fraud detection, risk analysis, and data security. *International Research Journal of Education and Technology*, 2020, 03(12).
71. Ingaldi M & Ulewicz R. How to make e-commerce more successful by use of Kano's model to assess customer satisfaction in terms of sustainable development. Sustainability. 2019; 11(18):4830.
72. Mandala RR & Kumar VKR. AI-driven health insurance prediction using graph neural networks and cloud integration. *International Research Journal of Education and Technology*, 2020, 03(10).
73. Erdeniz SP, Menychtas A, Maglogiannis I, Felfernig A & Tran TNT. Recommender systems for IoT enabled quantified-self applications. Evolving Systems. 2020; 11(2):291-304.
74. Ubagaram C & Kurunthachalam A. Bayesian-enhanced LSTM-GRU hybrid model for cloud-based stroke detection and early intervention. *International Journal of Information Technology and Computer Engineering*, 2020, 8(4).
75. Alhijawi B & Kilani Y. The recommender system: a survey. *International Journal of Advanced Intelligence Paradigms*. 2020; 15(3):229-251.
76. Ganesan S & Hemnath R. Blockchain-enhanced cloud and big data systems for trustworthy clinical decision-making. *International Journal of Information Technology and Computer Engineering*, 2020, 8(3).
77. Mitrevski PJ & Hristoski IS. Behavioral-based performability modeling and evaluation of e-commerce systems. Electronic Commerce Research and Applications. 2014; 13(5):320-340.
78. Musam VS & Purandhar N. Enhancing agile software testing: A hybrid approach with TDD and AI-driven self-healing tests. *International Journal of Information Technology and Computer Engineering*, 2020, 8(2).
79. Sun Y, Fang S & Hwang Y. Investigating privacy and information disclosure behavior in social electronic commerce. Sustainability. 2019; 11(12):3311.
80. Musham NK & Bharathidasan S. Lightweight deep learning for efficient test case prioritization in software testing using MobileNet & TinyBERT. *International Journal of Information Technology and Computer Engineering*, 2020, 8(1).
81. Khan HU & Uwemi S. What are e-commerce possible challenges in developing countries: a case study of Nigeria. *International Journal of Business and Systems Research*. 2018; 12(4):454-486.