مركز الأكاديمية للنشر
ACADEMY Publishing Center

E-ISSN: 2735-5985

ACE

ADVANCES in COMPUTING & ENGINEERING

JOURNAL

VOLUME 5
ISSUE 1
JUNE 2025

Academy Publishing Center
Journal of Advances in Computing and Engineering (ACE) First edition 2021

Arab Academy for Science, Technology, and Maritime Transport, AASTMT Abu Kir Campus, Alexandria, EGYPT
P.O. Box: Miami 1029
Tel: (+203) 5622366/88 – EXT 1069 and (+203) 5611818
Fax: (+203) 5611818
Web Site: http://apc.aast.edu

Volume 5, Issue 1, June 2025
ISSN: 2735-5977 Print Version
ISSN: 2735-5985 Online Version

ACE
Journal of Advances in Computing and Engineering

Journal of Advances in Computing and Engineering (ACE) is a peer-reviewed, open access, interdisciplinary journal with two issues yearly. The focus of the journal is on theories, methods, and applications in computing and engineering. The journal covers all areas of computing, engineering, and technology, including interdisciplinary topics.
All submitted articles should report original, previously unpublished research results, experimental or theoretical. Articles submitted to the journal should meet these criteria and must not be under consideration for publication elsewhere. The manuscript should follow the style specified by the journal.
ACE also publishes survey and review articles in the scope. The scope of ACE includes but not limited to the following topics: Computer science theory; Algorithms; Intelligent computing; Bioinformatics; Health informatics; Deep learning; Networks; Wireless communication systems; Signal processing; Robotics; Optical design engineering; Sensors.

The ACE journal does not collect Articles Processing Charges (APCs) or submission fees, and is free of charge for authors and readers, and operates an online submission with the peer review system allowing authors to submit articles online and track their progress via its web interface. The journal is financially supported by the Arab Academy for Science, Technology, and Maritime Transport AASTMT in order to maintain quality open-access source of research papers on Computing and Engineering.

ACE has an outstanding editorial and advisory board of eminent scientists, researchers and engineers who contribute and enrich the journal with their vast experience in different fields of interest to the journal.

Editorial Committee

Advisory Board

Administrative Committee

TABLE OF CONTENT

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

REAL-TIME MOBILE BROADBAND QUALITY OF SERVICE PREDICTION USING AI-DRIVEN CUSTOMER-CENTRIC APPROACH

Ayokunle A. Akinlabi[1,a*], Folasade M. Dahunsi[2,b], Jide J. Popoola[3,c] and Lawrence B. Okegbemi[4,d]

[1,2]Department of Computer Engineering, The Federal University of Technology Akure, Nigeria

[3]Department of Electrical and Electronics Engineering, The Federal University of Technology Akure, Nigeria

[4]Department of Mechanical Engineering, The Federal University of Technology Akure, Nigeria

Emails: {aaakinlabi@futa.edu.ng, fmdahunsi@futa.edu.ng, jjpopoola@futa.edu.ng, lawrencebolu@gmail.com}

Received on, 05 May 2025 - Accepted on, 26 May 2025 - Published on, 18 June 2025

ABSTRACT

Statistical methods employed in evaluating the quality of service (performance) of mobile broadband (MBB) networks face drawbacks relating to the accurate and reliable processing of the huge amounts of heterogenous real time traffic data generated from MBB networks. Since the traffic patterns experienced in MBB networks are largely complex, highly dynamic, and heterogenous in nature, statistical methods may not adjust adequately to the changing network conditions. This highlighted gap can be addressed by machine learning (ML), as it has been effectively used in the past to support the analysis and knowledge discovery of communication systems' traffic data through the identification of intricate and hidden patterns. This paper presents the application of ML techniques to predict MBB Quality of Service (QoS) in real-time, using a custom-built MBB performance application referred to as MBPerf that collects five (5) network metrics (DNS lookup, download and upload speeds, latency, signal strength), location information, and device characteristics across diverse network conditions in the South West region of Nigeria. The QoS modeling task was carried out using an MBPerf pre-processed dataset. Three (3) classification algorithms, including Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGBoost), were trained using the MBPerf QoS dataset and then evaluated in order to determine the most effective model based on certain evaluation metrics – accuracy, precision, F1-Score and recall. Following hyperparameter tuning to improve the model's performance, the selected model was deployed in a real-world network environment to classify QoS into "Above Average," "Average," and "Below Average" categories. Mobile customers receive real-time notifications with actionable insights based on the predicted QoS class, empowering them to optimize their usage and troubleshoot issues. From the performance results obtained for the 3 ML models trained with the MBPerf dataset, SVM (95%) and XGBoost (97%) significantly outperformed RF (59%) in terms of accuracy. However, the performance difference between the SVM and XGBoost models is not significant. Interestingly, the three models showed great capability to accurately make predictions on the three QoS categories (classes) as depicted by the ROC-AUC and mlogloss curves.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

Lastly, the feature importance plot shows that QoS is the collective effect of service performance and not a function of QoS metrics only that determines the degree of satisfaction of a user of the service. This Artificial Intelligence (AI) powered system promotes a more transparent and efficient MBB experience for all stakeholders in Nigeria's fast evolving digital landscape.

Index words: Cellular Network, Mobile Broadband, Quality of Service, Machine Learning, Crowdsourcing, Extreme Gradient Boosting, Random Forest, Support Vector Machine

1. INTRODUCTION

Mobile broadband stands out as one of the most groundbreaking technologies of the 20th century and has continued to revolutionize global communications significantly in the current 21st century. MBB networks, which support a number of vital services essential to present-day society, are arguably now the cornerstone of the worldwide communications infrastructure [1]. As daily activities across education, health, business, entertainment, social life, and news sectors become increasingly bandwidth-intensive and bandwidth-hungry, mobile customers demand fast and highly responsive access wherever they go – whether at home, work, in cities or rural areas, in vehicles or on the move. As a result, their reliance on MBB networks continues to grow. One cannot overstate MBB's many benefits. In the true sense, the fourth generation (4G) technologies are the bedrock of MBB, capable of delivering more than 100 Mbps when customers are on the move. The fifth generation (5G) technologies, which have not yet been fully rolled out in developing markets, are expected to deliver enhanced MBB speeds reaching 20 Gbps, thereby extending 4G's Internet of Things (IoTs) capability and enabling mission-critical applications that demand ultra-high reliability and very low latency. The sixth generation (6G) system is anticipated to be in commercial use by 2030 [2]. The massive amounts of customer data collected continuously on a daily basis by MNOs from different sources, such as network traffic, social media, phone calls, Short Message Service (SMS), Internet activities, and mobile device interactions, provide useful information about usage patterns, service quality, network operations, and market trends. Machine learning offers algorithmic models that can intelligently analyze these data, uncover patterns, make predictions, and recommend actions without having to do explicit programming. The core of this research involves the development and deployment of an AI-powered system capable of accurately predicting MBB QoS on customers' smartphones, based on the historical and real-time data streams comprising five measured QoS metrics: phone, location, and network information. A MBB performance application (MBPerf), custom-built for the purpose of data collection, provided the dataset needed for the QoS prediction and classification tasks.

Since MBPerf data was unlabeled, a widely used unsupervised learning algorithm (k-means clustering) attempted to segment the data into $k = 3$ distinct groups, effectively uncovering basic patterns within the data [3]. The preprocessed k-means classified MBPerf dataset was used to train and test 3 ML algorithms (classifiers). These classifiers include RF, SVM, and XGBoost. The selection of the three classifiers was based on their collective ability to address the complex, non-linear, and noisy characteristics inherent in real-world network data. Existing studies have also revealed the efficiency of these classifiers in network-related modeling. The combination of the aforementioned strengths allows for robust modeling of the diverse feature interactions and the handling of the high dimensional data, which is likely to result in high prediction accuracy. Based on precision, F1-Score, and recall evaluation metrics, XGBoost was selected as the best performing model. Following hyperparameter tuning to optimize the model's Performance, XGBoost was deployed and integrated into MBPerf to classify the predicted QoS into "Above Average," "Average," and "Below Average" categories. The successful implementation of the QoS prediction system offers significant advantages to key stakeholders within the Nigerian telecommunications ecosystem. Importantly for mobile customers, the integration of the trained ML model into MBPerf provides real-time notifications regarding their predicted QoS class. This empowers users with actionable insights like those experiencing "Above Average" QoS can confidently engage in bandwidth-intensive activities, while users in the "Average" category are informed of typical performance levels. Crucially, users predicted to experience "Below Average" QoS receive notifications suggesting potential causes such as network congestion, weak signal, and actionable steps

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

to mitigate these issues, such as moving to a different location, switching network modes, or contacting their MNOs for support. This transparency enhances user experience, reduces frustration, and grants informed usage of MBB services. This paper is organized as follows: section 2 discusses existing studies relating to network performance improvements and the integration of AI into cellular networks. Section 3 presents the proposed framework employed in mobile broadband quality of service prediction, while section 4 presents and analyzes the results of data validation, clustering techniques, and classifier evaluation. Finally, section 5 concludes the paper.

2. RELATED WORKS

With respect to network performance improvement, a great deal of studies have been carried out on integrating AI into communication networks. Wireline and wireless communication networks are constantly growing due to the growing number of devices that are linked to the Internet. The number of connected people, devices, and data transmission volumes on the Internet is growing at an exponential rate. As such, it is getting harder to control, manage, monitor, and predict traffic patterns in mobile broadband networks through manual techniques due to their vast scale and rapid growth. As a result, ML, which is capable of accurate traffic predictions, can be the key to automating network decisions to boost performance [4].

[5] Suggested a personalized approach for predicting Internet throughput for streaming applications using vectorization and clustering methods. Consequently, decision trees, Naye Bayes, and ARIMA models were trained and evaluated to determine which algorithmic model predicted throughput most accurately. Location coordinates, time of the day, user physical movement speed, and download speed were the features used for the prediction task. The time-series dataset was found to be a better fit for the ARIMA model, but it was not particularly accurate. The decision tree and vectorized inputs showed the best accuracy during the testing phase. The throughput predictor agent was developed using only a single 4G network dataset, which may limit the generalizability of the results to other network technologies or scenarios.

[6] Employed ML and Deep Learning (DL) algorithms, including RF, SVM, and Long Short-Term Memory (LSTM) to predict throughput in cellular networks with Channel Quality Indicator (CQI), cell load, throughput, and mobility mode as test parameters (features). The results of the study show that ML and DL can improve throughput prediction accuracy in cellular networks and that feature engineering and architectural components have a significant impact on prediction accuracy.

[7] Considered a Fuzzy Knowledge-Based (FKB) approach with Triangular Membership Function (TMF) for evaluating commonly used QoS indicators needed to model the MBB network performance provided by three (3) MNOs in the Niger Delta region. Data were collected using the software-based approach. Signal strength, packet loss, and upload and download speeds were the test parameters considered by the authors of this study. Results showed that the selected MNOs vary in QoS with respect to the test parameters compared. The research only considered just three (3) QoS metrics, and the number of days for data collection was too short – 21 days. These factors can limit the research since a high volume of observations and features in the training and test data will enhance the performance of the used model.

[8] Proposed a data-driven framework for detecting 4G cells with underlying network throughput problems with the help of ML techniques. The methodology employed by the authors is a combination of clustering models and deep neural networks (DNNs) that require only a small amount of expert-labeled data. The test parameters measured and fed to the models include CQI, Throughput, Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), etc. The result showed that resolving problematic cells could boost the network's total throughput by 8%, based on data obtained from real 4G cells. A comparison of the proposed framework with SVM yielded unsatisfactory results, indicating that the DNN is a better choice for this problem. The proposed framework was tested on a single 4G network dataset; hence, it may limit the generalizability of the results to other networks.

[9] Proposed a customer churn prediction model designed to detect potentially dissatisfied

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

customers at an early stage and implement proactive retention strategies using classification and clustering techniques. The study used six ML models to predict customer churn: Logistic Regression, K-Nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, and XGBoost. The test parameters considered include international plan, voice mail plan, number of voicemail messages, total day minutes, total day calls, total day charge, total evening minutes, total evening calls, total evening charge, total night minutes, total night calls, total night charge, etc. The results indicate that XGBoost and RF outperformed K-Nearest Neighbors, Support Vector Machines, and Decision Trees, achieving higher prediction accuracy across accuracy, precision, F1-score, and recall metrics.

[10] Presented a methodology for modeling and predicting the Quality of Experience (QoE) of mobile applications in WiFi or cellular networks. This study showed that gathering background information from smartphones and in-situ QoE ratings and applying these data to standard ML methods can create a precise predictive model for 'High' and 'Low' QoE. [10] were able to construct predictive QoE models with qualitative and quantitative data collected in a living laboratory. Their work also provided guidance to mobile application developers on how to design QoE-aware applications. ML has been identified as a proficient AI technology that helps in understanding and predicting customer churn, thereby allowing MNOs to anticipate churn problems by recognizing possible disgruntled consumers early on and taking proactive steps to keep them. Undoubtedly, this will result in enhanced retention tactics and a rise in profitability for the MNOs.

The study carried out by [11] uses a DL-based approach, specifically a hybrid CNN-LSTM model, to detect QoS anomalies. The UNSW-NB15 dataset used was preprocessed by handling missing or inconsistent values and selecting relevant features, including the QoS metrics such as bandwidth, latency, jitter, packet loss, and service availability. The study then trains the CNN, LSTM, and hybrid CNN-LSTM models using the preprocessed dataset and evaluates the models using metrics such as accuracy, precision, recall, F1-Score, and false positives. The hybrid CNN- LSTM model outperforms the individual CNN and LSTM models, achieving high accuracy and efficiency in detecting anomalies. The hybrid model outperforms the individual models with 98.67% accuracy, precision, recall, and F1-Score. This proves its anomaly detection resilience. Measurable results show the model improves network dependability, resource allocation, and user satisfaction. Potential limitations of the study include the need for large amounts of data to train the DL models and the complexity of implementing these models in real-time network environments.

[12] proposed a deep learning-based approach for QoS prediction to address the problems of data sparsity and cold-start inherent in the current QoS prediction approaches, such as collaborative filtering methods. The study also explored the impact of geographical characteristics of services/users and QoS rating on the prediction problem. The methodology involves combining a matrix factorization model based on a deep auto-encoder (DAE) and a clustering technique based on geographical characteristics to improve prediction effectiveness. In achieving the QoS prediction, the QoS data was clustered using a self-organizing map that incorporates the knowledge of geographical neighborhoods. This clustering step effectively handled the cold start problem. In addition, for each cluster, a DAE was trained so as to minimize the squared loss between the ground truth QoS and the predicted one. Next, the missing QoS of a new service is predicted using the trained DAE related to the closest cluster. The effectiveness and robustness of the proposed approach were evaluated using a comprehensive set of experiments based on a real-world web service QoS dataset. The experimental results showed that the method achieves a better prediction performance compared to several state-of-the-art methods.

The objective of the research conducted by [13] was to explore the application of artificial intelligence (AI) techniques, particularly ML, in predicting QoS on mobile networks. The main focus was to test the ability of AI models to predict several QoS parameters, such as throughput, latency, and packet loss. The methods used in this study include data collection from simulations of mobile networks generating a dataset containing 100,000 network traffic samples, data pre-processing, feature selection, and model evaluation using metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). The AI models used include K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Deep Learning (LSTM). The results of this study show that LSTM-based deep learning models have the lowest prediction error rate in estimating packet loss,

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

followed by SVM and then KNN. The study also found that AI approaches, especially Deep Learning such as Long Short-Term Memory (LSTM), are able to predict QoS with high accuracy in a dynamic cellular network environment.

The research conducted by [19] addressed the critical problem of predicting QoS in cellular networks, a key factor in ensuring a seamless user experience in a world of ubiquitous computing and giant data networks. With the help of DL techniques, future complex lags based on time series data in three simulated drift scenarios created in OMNET++, Simu5G, Veins, and Sumo were predicted. The study employed the N-BEATS model – a deep learning architecture specialized in time-series forecasting that can identify unique patterns and trends to predict future values. The implementation was done with PyTorch, including data preparation, creation of N-BEATS model architecture, configuration of loss functions and optimizers, and training of the model for prediction. Evaluation metrics such as MSE, RMSE, and MAE were used for error estimation.

3. PROPOSED SCHEME FOR THE PREDICTION OF MOBILE BROADBAND QUALITY OF SERVICE

3.1. PHASES OF THE RESEARCH

This research is divided into two (2) phases. The flow diagram in Figure 1 presents the step-by-step process followed to achieve the research objectives.

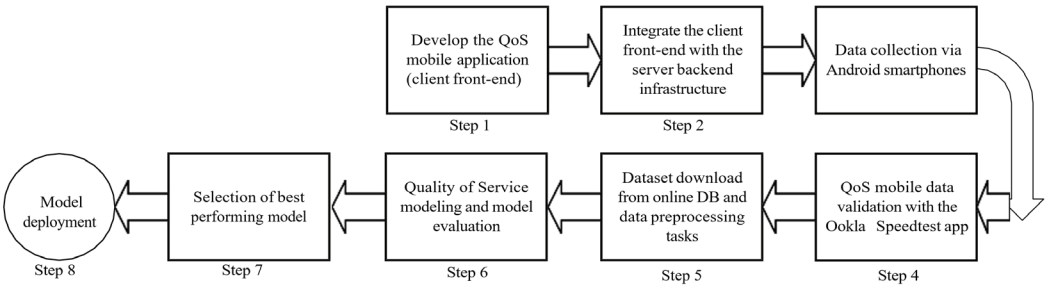


Fig 1: Flow Diagram Showing Step-by-Step Process to Achieve Research Objectives

3.1.1. PHASE ONE – QOS DATA COLLECTION AND VALIDATION

Step 1: Development of MBPerf, which when installed on the mobile devices of customers collected their relevant user-data. Additionally, MBPerf measured five (5) selected QoS metrics. MBPerf was developed for the main purpose of data collection.

Step 2: The client front-end system (MBPerf) was integrated with the server back-end DB (Google Firestore) for seamless data collection and storage.

Step 3: MBPerf was installed on five hundred (500) Android smartphones to collect MBB QoS data for a period of nine (9) months.

Step 4: The dataset collected with MBPerf was validated with the data obtained from the Ookla Speedtest app. The result of the data validation task is presented in section 4.1.

Step 5: The total collected data which amounted to 6500 measurement instances were downloaded from the online DB to the research Laptop (specification: Intel(R) Core (TM) i3-8130U CPU at 2.21 GHz, 8.00 GB RAM and x64-based processor).

The output of this phase is MBPerf QoS dataset needed for phase two. Some of the User Interfaces (UIs) of MBPerf QoS application (Figures 11a, 11b and 11c) and MBPerf's online dashboard (Figures 13 to 16) are shown as supplementary Figures.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

3.1.2. PHASE TWO – QUALITY OF SERVICE MODELLING AND PREDICTION

Steps 8-11: The QoS modeling task, which is described in detail in section 3.6 was carried out using the pre-processed dataset. Three (3) ML models – RF, SVM and XGBoost were trained with the QoS dataset and then evaluated in order to select the best performing model based on certain evaluation metrics. Following hyperparameter tuning to optimize the model's performance, the selected model was deployed in a real-world network environment.

3.2. QUALITY OF SERVICE MODELLING USING MACHINE LEARNING TECHNIQUES

The QoS prediction system was built with strong foundations in robustness, security, and scalability and relies on a simple, modern, and cloud-based architecture perfectly suited for the dynamics of real-time cellular networks. Robustness is inherent in the choice of Google Firebase as the cloud server backend infrastructure and specifically adopting Firestore as the DB. Firebase provides a fully managed, globally distributed infrastructure, minimizing downtime and ensuring high availability for continuous data collection from the growing customer (volunteer) base. Because the MBPerf mobile application and the online dashboard are based on the Flutter framework, the applications include solid error handling and data validation, which makes sure the data is safe and accurate, regardless of any challenging connection issues. With respect to security, the admin's dashboard login is managed by Firebase authentication and user data is protected, anonymized, and only accessed by authorized stakeholders. Data privacy is very important because we already have over 500 individuals sending in their data. Suffice to state that MBPerf developers place top priority on customers' personal information, which is why network performance data are collected anonymously. None of the customer's personally identifiable information is collected by MBPerf. The more reason volunteers are not asked to sign up or register to install or use the QoS application. Lastly, Firestore and its linked services, all within Firebase, were designed to scale seamlessly, hence, a developer does not have to manage complex infrastructures as user count and data keep rising. Both the Flutter front end and Flutter web dashboard are built for high performance, which keeps the experience smooth as the volunteer base grows and more stakeholders are onboarded. The step-by-step process followed to achieve the QoS modeling and prediction is illustrated in Figure 2 and elaborated in sub-sections 3.2.1 to 3.2.10.

3.2.1. MBPERF DATASET

The first requirement for building a machine learning model is the dataset. A dataset is a collection of data in a specific format for a specific problem. The volunteers' data records, including their location, network, mobile device information, and measurement results stored on Google Firestore, were downloaded as a single .csv file and then mounted on Google Drive. The coding environment utilized for the various ML tasks was Google Colab, a cloud-based service that requires no setup and offers free access to open-source tools such as Jupyter Notebook and Python. Furthermore, it provides computing resources like GPUs and TPUs, making it particularly suitable for ML, data science, and educational purposes [14]. The drive module imported unto the google.colab package, allows access to the QoS dataset already mounted on the Google drive. NumPy, Pandas, Matplotlib, Seaborn, Sklearn, and XGBoost were the necessary libraries needed to be imported for data preprocessing, data analysis, visualization, and machine learning tasks. The QoS dataset consists of 6500 measurement instances (rows) and 20 (features) columns. The dataset is not labeled, i.e., no record of QoS classification, as this is the target variable. The dataset consists of both numerical and categorical variables.

3.2.2. DATASET PREPROCESSING

Raw data must undergo preprocessing and feature engineering to ensure its suitability for ML models. Data cleaning, data encoding, frequency encoding, label encoding and one-hot encoding are the various data preprocessing tasks carried out on the raw QoS dataset. The QoS dataset comprises a mix of categorical and numerical data types. Since many ML models are incompatible with categorical data, hence, it was converted into numerical format before

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

being utilized in the selected models.

3.2.3. FEATURE ENGINEERING

Feature engineering is a crucial step in preparing data for ML. It is the process of creating appropriate features from pre-existing features, thereby leading to improved predictive performance [15]. Domain expertise/knowledge of this research area and iterative trial and error and model evaluation were followed in performing the following feature engineering tasks, which include feature selection/extraction, feature expansion, feature scaling, and Principal Component Analysis (PCA). Standardization was chosen as the feature scaling method, ensuring that all features in the training and test sets are kept on the same scale. This process also accelerates model training. The mathematical formula for standardization, according to [9], is:

$$x' = \frac{[x - \bar{x}]}{\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}} \qquad (1)$$

where x is original feature, () is the scaled feature, () is the mean of x and the denominator is the standard deviation of x." below the equation, i.e.,

PCA was useful for several purposes, including noise reduction, feature extraction, and dimensionality reduction of the QoS data before applying the selected ML algorithms, which led to improved efficiency and reduced overfitting.

3.2.4. DATA SPLITTING

The preprocessed QoS dataset was divided equally into two parts (50:50). The first half (split A) was used to train the k-means clustering algorithm, and the resulting model was then applied to predict the target variables (QoS categories) for the second half (split B). Split B was now treated as a completey labeled dataset and further divided in an 80:20 ratio, serving as the ground truth for training and testing the three selected classification models – RF, SVM, and XGBoost. The afore-explained data splitting strategy was adopted to prevent any data leakage, which can result in poor model generalization or creating a biased model, by ensuring that first, the clustering algorithm is trained on a separate dataset from what is used in the classification models. Second, the classification models are evaluated on unseen data, and lastly, no information from the test data influences the training process.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

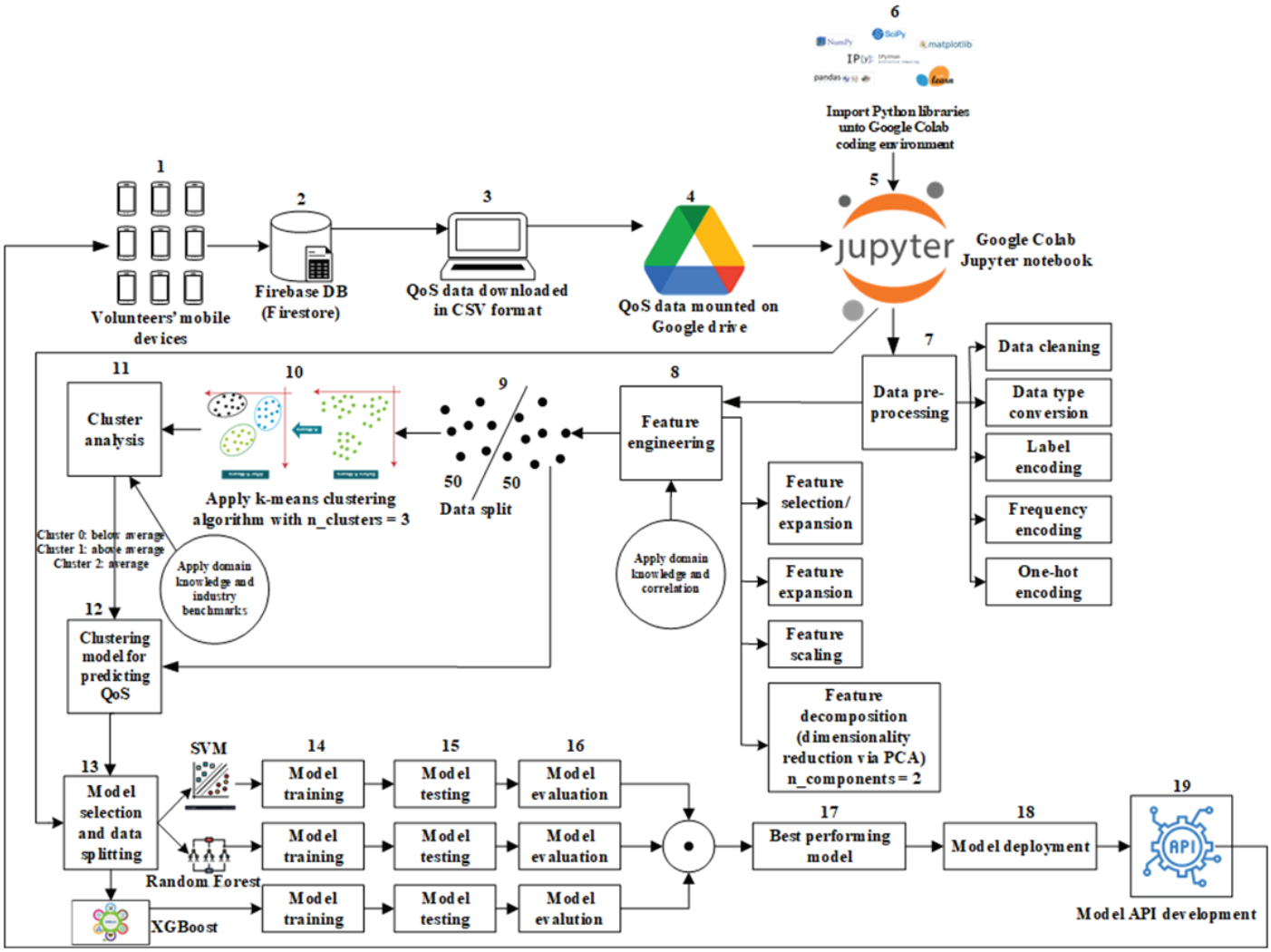


Fig. 2: Step-by-Step Process to Achieve the QoS Modelling

3.2.5. K-MEANS CLUSTERING

Since the QoS dataset was unlabeled, a popular unsupervised learning algorithm known as k- means clustering attempted to divide the data into k = 3 discrete groups and is effective at uncovering basic data patterns [3]. The k-means clustering algorithm worked by splitting the data into 3 clusters representing the three yet unknown QoS classifications/categories. The first step to achieve the 3 cluster classes was to select a centroid for each k cluster. The remaining data points on the scatterplot are then assigned to the closest centroid by measuring the Euclidean distance given by the formula [3]:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots (x_n - y_n)^2} \qquad [2]$$

where x and y are two data points, and n is the number of features. The centroid of the ith cluster is calculated as [3]:

$$centroid_i = \left(\frac{1}{N_i}\right) * \sum x_j \qquad [3]$$

where $centroid_i$ is the centroid of the ith cluster, N_i is the number of data points in the ith cluster, and x_j are the data points assigned to the ith cluster.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

3.2.6. CLUSTER ANALYSIS

The analysis of the outputs of the k-means clustering algorithm are 3 QoS classes/categories designated as 0, 1 and 2. After thorough analysis of the 3 clusters, it was concluded that cluster 0 is the 'average class', cluster 1 is the 'above average class' and cluster 2 is the 'below average class'.

3.2.7. SELECTION, TRAINING, AND TESTING OF ML MODELS

The preprocessed k-means classified QoS dataset was used to train and test 3 ML algorithms (classifiers). As explained below, the classifiers include RF, SVM, and XGBoost. These three classifiers were selected because of their collective ability to address the complex, non-linear, and noisy characteristics inherent in real-world network data. Existing studies have also revealed the efficiency of these classifiers in network-related modeling. The combination of the highlighted strengths allows for robust modeling of the diverse feature interactions and the handling of the high dimensional data, which is likely to result in high prediction accuracy.

i. Random Forests: RFs are an ensemble classifier which fits a large number of decision trees to a dataset, and then combines the predictions from all the trees [3]. Each tree in a RF is a weak learner. However, the RF is a strong learner combining multiple predictions into one aggregate prediction thereby resulting in a more accurate forecast.

ii. Support Vector Machine: SVM worked by constructing an optima hyperplane in a multidimensional space where the hyperplane separates the collected QoS sample data into three (3) different classes. Consider the QoS training sample consisting of N patterns $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ where x is the feature vector, and target $y_i \in \{-1, 0, +1\}$ with corresponding multi-class labels. The SVM parameters are determined by maximizing the margin hyperplane [3]:

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i . \alpha_j) \quad (4)$$

Subject to the constraints [3]:

$$\sum_{i=1}^{N} \alpha_i \text{ and } 0 \leq \alpha_i \leq C \quad (5)$$

where $K(x_i, x_j)$ is the kernel function used to map the data from the input space to the feature space and item C is a cost/slack parameter.

iii. XGBoost: is a decision tree ensemble based on gradient boosting, designed to be highly scalable and combines multiple weak learners (typically decision trees) to create a strong predictive model [16]. XGBoost aims to minimize a loss function, which is typically a combination of a training loss and a regularization term:

$$Objective = Loss + Regularization \quad (6)$$

Table 1 shows the details of the hyperparameters tuned in order to achieve the three predicted QoS classes.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

Table 1: ML Models Hyperparameters

ML Model	Hyperparameter	Value
k-means clustering	n_clusters	3
	random_state	0
	n_init	auto
	max_iter	300
	tol	0.0001
	random_state	0
RF classifier	n_estimators	1000
	criterion	log_loss
	max_depth	6
	random_state	42
SVM classifier	kernel	linear
	probability	TRUE
	random_state	42
XGBoost classifier	objective	multi: softmax
	num_class	3
	missing	1
	gamma	0
	learning_rate	0.1
	max_depth	3
	reg_lambda	1
	subsample	1
	colsample_bytree	1
	early_stopping_ rounds	10
	eval_metric	['merror,' 'mlogloss']
	seed	42

3.2.8. MODELS EVALUATION

To assess the effectiveness of the chosen ML models, selecting the appropriate evaluation metrics is crucial. Using an incorrect metric can lead to poor real-world performance of the ML model. Commonly used evaluation metrics include:

iv. Accuracy: This is used to deduce which ML model has the best capacity to recognize relationships and patterns between variables in the training dataset. It can be calculated by dividing the total number of correct predictions by the total number of predictions and then multiplying by 100 [9].

$$A = \frac{TP+TN}{TP+TN+FP+FN} * 100\% \qquad (7)$$

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

where True Positive (TP) denotes that the actual value is true and the model predicts true; False Positive (FP) means the actual value is false and the model predicts true; True Negative (TN) means the actual value is false and the model predicts false and False Negative (FN) means the actual value is true, and the model predicts false.

v. Precision: This is the ratio of true positives and total positives predicted. The formula for Precision is given as [9]:

$$P = \frac{TP}{TP+FP} \quad (8)$$

vi. Recall/Sensitivity/Hit-Rate: The recall metric focuses on type-II errors, i.e., FN. However, it cannot measure the existence of type-I errors, which are false positives. It is calculated as [9]:

$$\mathrm{R} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \quad (9)$$

vii. F1-score: This is the combination of precision and recall. The F1- score is the harmonic mean of Precision and Recall. The formula of the two essentially is [9]:

$$\mathrm{F1score} = \frac{\mathrm{Precision}}{\mathrm{Precision+Recall}} \quad (10)$$

3.2.9. BEST PERFORMING MODEL

Comparing the results from the evaluation metrics, the best-performing model/classifier, i.e., XGBoost, out of the three (3) previously trained models, was picked for deployment and integration with MBPerf mobile app.

3.2.10. MODEL DEPLOYMENT

Deploying the best-performing model to a production environment, i.e., integrating the ML model with MBPerf, involves the following steps: first, the trained model is saved via Pickle library based on Python, and a model serving platform (Render) is chosen. Second, the Flask web framework was used to develop a RESTful API that accepts MBPerf data, processes it through the ML model and returns the predicted QoS classes and notifications. Third, Render was used to deploy the serialized model. Also, the API was connected to it (Render) so that it could receive the required data, process it through the ML model, and return predictions. Fourth, the developed API is interfaced with MBPerf, allowing it to send measurement data to the API and then receive predicted QoS values and notifications. A notification system was implemented to alert users of the predicted QoS values, implemented using the Firebase Cloud Messaging (FCM) platform. The UI of the MBPerf QoS application, after being integrated with the ML classifier, gives notifications on the predicted QoS, as shown in supplementary Figure 12.

4. RESULTS AND DISCUSSION

In this section, the graphical results for the data validation, k-means clustering and models' performances are presented.

4.1. MBPERF DATA VALIDATION

MBPerf has to be validated to ensure that the accuracy, and reliability of its dataset meets the NCC/FCC standard. Measurement results from the Ookla Speedtest app, a widely recognized and reliable benchmark, were used to validate MBPerf measurement results. Ookla Speedtest is recognized by the Federal Communications Commission (FCC) in the US, the International Telecommunications Union (ITU), and several ISPs and Telcos [17]. For seven

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

(7) days, bi-hourly measurements of the download speed, upload speed, and latency were collected via MBPerf and Ookla Speedtest apps for each of the three MNOs under evaluation. Correlation analysis was carried out on the measurement results and presented in Table 2. The correlation analysis results reveal strong positive correlations between MBPerf and Ookla Speedtest measurements. For download speed, the correlation coefficient, r = 0.856; for upload speed, r = 0.884, while for latency metric, r = 0.755. These results show that MBPerf measurements are significant and, to a very large extent, accurate and reliable.

Table 2: Correlation Results between MBPerf and Ookla Speedtest Measurements

	MBPerf Download (Mbps)	*Ookla Download (Mbps)*	*MBPerf Upload (Mbps)*	*Ookla Upload (Mbps)*	*MBPerf Latency (ms)*	*Ookla Latency (ms)*
MBPerf Download (Mbps)	1					
Ookla Download (Mbps)	**0.856**	1				
MBPerf Upload (Mbps)	0.667	0.595	1			
Ookla Upload (Mbps)	0.775	0.723	**0.884**	1		
MBPerf Latency (ms)	-0.600	-0.621	-0.691	-0.773	1	
Ookla Latency (ms)	-0.572	-0.607	-0.603	-0.697	**0.755**	1

4.2. CLUSTERING RESULT

The result of the k-means clustering is shown in Figure 3, where X denotes the centroids picked by Python's k-means estimator with two principal components ($n\_clusters$ = 2) and three clusters (k = 3) representing the three classified QoS categories: average, above average and below average. To effectively map the k-means clusters to the three QoS classes (Figure 4), clusters' centroids were analyzed. In addition, data distributions across relevant QoS metrics, taking into cognizance the industry benchmarks, were also analyzed to identify clusters representing each of the QoS classes. The mapping was validated using domain knowledge and user feedback.

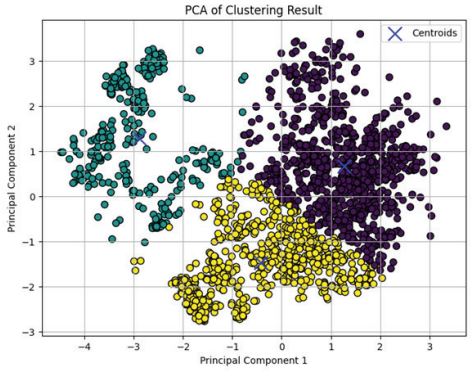


Fig. 3: Scatter Plot Showing Clustered QoS Data

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

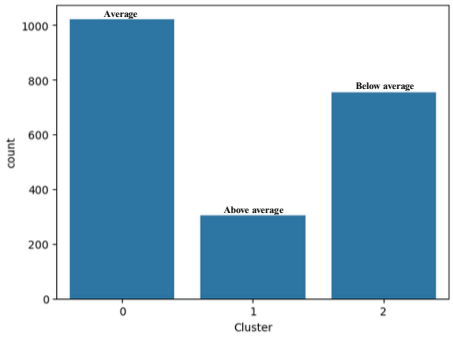


Fig. 4: Count Plotted against Clusters

4.3. RANDOM FOREST

The confusion matrix of Figure 5 shows that the RF classifier performed exceptionally well in identifying the average QoS class (256 correct classifications) but is challenged in distinguishing the above-average and below-average categories with significant misclassifications into the above-average and below-average QoS classes, particularly the average class (98 misclassifications). As shown in the performance comparison results of the ML models in Figure 9, RF achieved a balanced accuracy of 59%, micro precision of 74%, micro recall of 71%, and a micro F1-score of 68%. The high ROC-AUC scores (Figure 6) reveal that RF has the capability to accurately make predictions about the three QoS categories (classes) but is not as accurate as SVM and XGBoost.

4.4. SUPPORT VECTOR MACHINE

The confusion matrix in Figure 7 shows that the SVM classifier performed greatly in identifying all the QoS classes, with only four misclassifications in the average and above-average QoS classes and 15 misclassifications in the below-average QoS class. As shown in the performance comparison results of the ML models in Figure 10, SVM achieved a balanced accuracy of 95%, micro precision of 96%, micro recall of 95%, and a micro F1-score of 96%. These performance results, reveal that SVM has excellent capability to accurately make predictions about the three QoS categories (classes) better than the RF model.

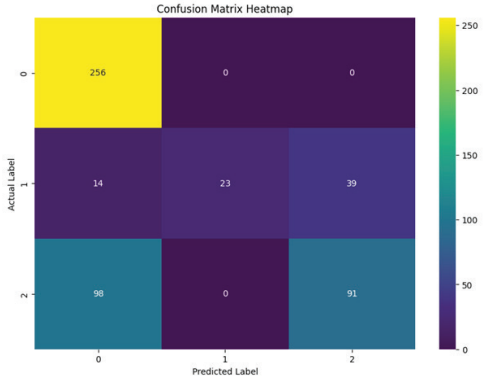


Fig. 5: Random Forest Confusion Matrix Heatmap

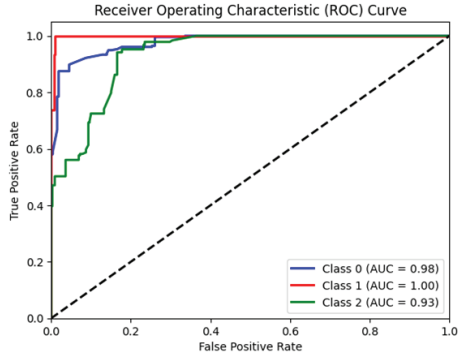


Fig. 6: Random Forest ROC Curve Matrix Heatmap

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

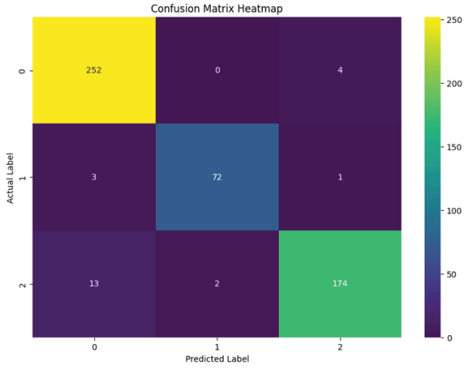


Fig. 7: SVM Confusion Matrix Heatmap

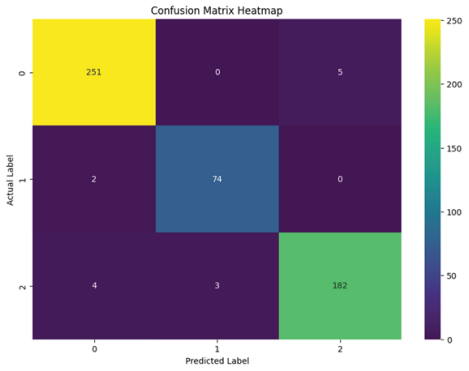


Fig. 8: XGBoost Confusion Matrix Heatmap

4.5. XGBOOST

The confusion matrix in Figure 8 shows that the XGBoost classifier, just like SVM, also performed greatly in identifying all the QoS classes with only five misclassifications in the average QoS class, two misclassifications in the above average class, and seven misclassifications in the below average QoS class. The performance comparison results of the ML models in Figure 9 reveal that XGBoost achieved a balanced accuracy of 97%, micro precision of 97%, micro recall of 96%, and a micro F1-score of 96%. The mlogloss curve presented in Figure 10 indicates a very good model performance. This is true because the curve is seen to decrease as the number of iterations increases, indicating that the model is improving. In summary, the highlighted results show that XGBoost, just like SVM, possesses excellent capability in predicting the three QoS classes. The performance difference between SVM and XGBoost is not significant.

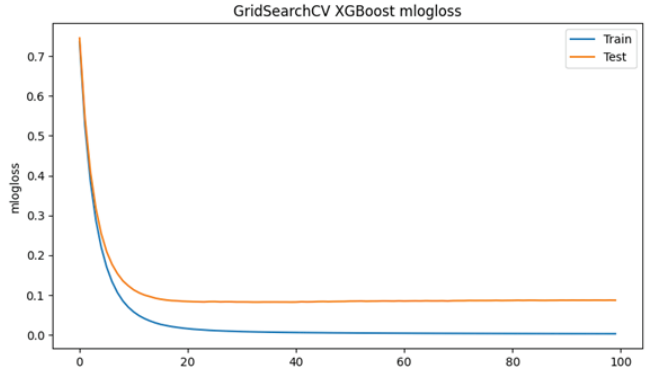


Fig. 9: XGBoost mlogloss Curve

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

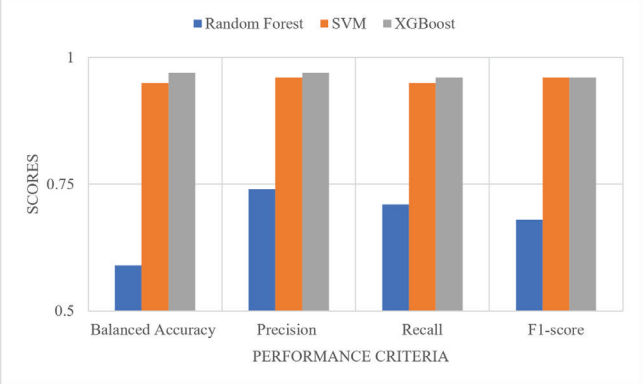


Fig. 10: performance Comparison of RF, SVM, and XGBoost ML Models

The performance comparison of the three (3) ML classifiers is presented in Table 3.

Table 3: Performance Comparison RF, SVM, and XGBoost classifiers

Model	Evaluation Metrics			
	Balanced Accuracy	Precision	Recall	F1-score
RF	0.59	0.74	0.71	0.68
SVM	0.95	0.96	0.95	0.96
XGBoost	0.97	0.97	0.96	0.96

4.6. FEATURE IMPORTANCE

One way to explain a model's behavior is to use feature importance, which measures the marginal contribution of each feature to a model's decisions. Each feature is assigned a score indicating its relative importance in the model, where higher scores indicate greater importance. Figure 11 shows that the OS version, device model, location, operator name, manufacturer, and brand name are the five most prominent features for predicting QoS in this research. A critical look at these features shows that they are the independent variables that are needed to explain the dependent variables, which are the QoS metrics. This important finding conforms with the ITU- T definition of QoS from the end user's perspective that: "QoS is the collective effect of service performance (not measured based on QoS metrics only) which determines the degree of satisfaction of a user of the service [18].

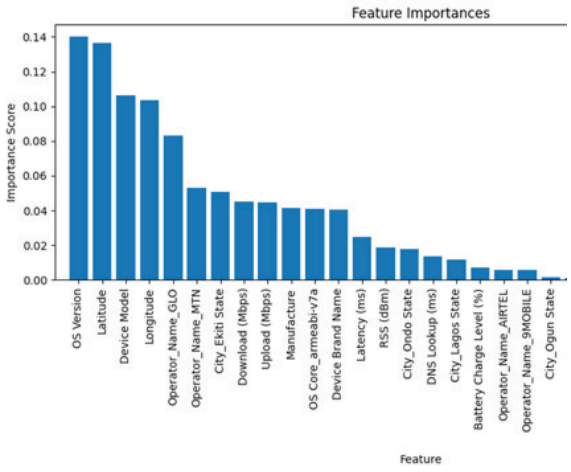


Fig. 11: Feature Importance Bar Chart

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

5. CONCLUSION

This research presents a real-time framework for predicting and classifying MBB QoS, utilizing a customer-centric and AI-driven approach to bridge the information gap between mobile customers and MNOs, as well as empower customers to evaluate the actual quality of MBB services provided by MNOs in Nigeria. By employing a host and crowdsourced based methodology for collecting real-time QoS metrics data directly from a growing network of volunteer devices, the research has demonstrated the possibility of employing machine learning techniques in classifying network performance into "Above Average," "Average," and "Below Average" classes. This QoS system not only provides accurate predictive insights but also provides mobile customers with real-time notifications tailored to their current network conditions. The developed system benefits all stakeholders in Nigeria's telecom industry. The regulator can gain valuable insights into how well networks function in several geographical areas and among different MNOs. These insights will aid evidence-based policymaking. MNOs can be compared, and services are offered in an equitable way, which makes the industry more transparent and accountable. For the MNOs, this system allows for the early identification of congestion hotspots, optimization of resource allocation, and targeted infrastructure investments, leading to improved operational efficiency and customer loyalty. Most importantly, mobile customers are offered more than just the numerical results of Internet performance; they also receive a personalized network QoS prediction that help them improve their network activities and easily manage any problems. Moving forward, as the volunteer base keeps increasing and thousands of measurement instances are collected, attention will now shift to even more advanced prediction techniques. Future works will adopt Deep Learning models, including Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN). Harnessing the power of deep learning enables the discovery of more intricate patterns with better predictive accuracy for dynamic network fluctuations. Out of the three ML models – "employed in this study for the QoS classification task;" experimental results indicate that XGBoost achieved the highest performance, followed by SVM and then RF.

COMPETING INTERESTS

The authors declare that no competing interests exist.

SUPPLEMENTARY FIGURES

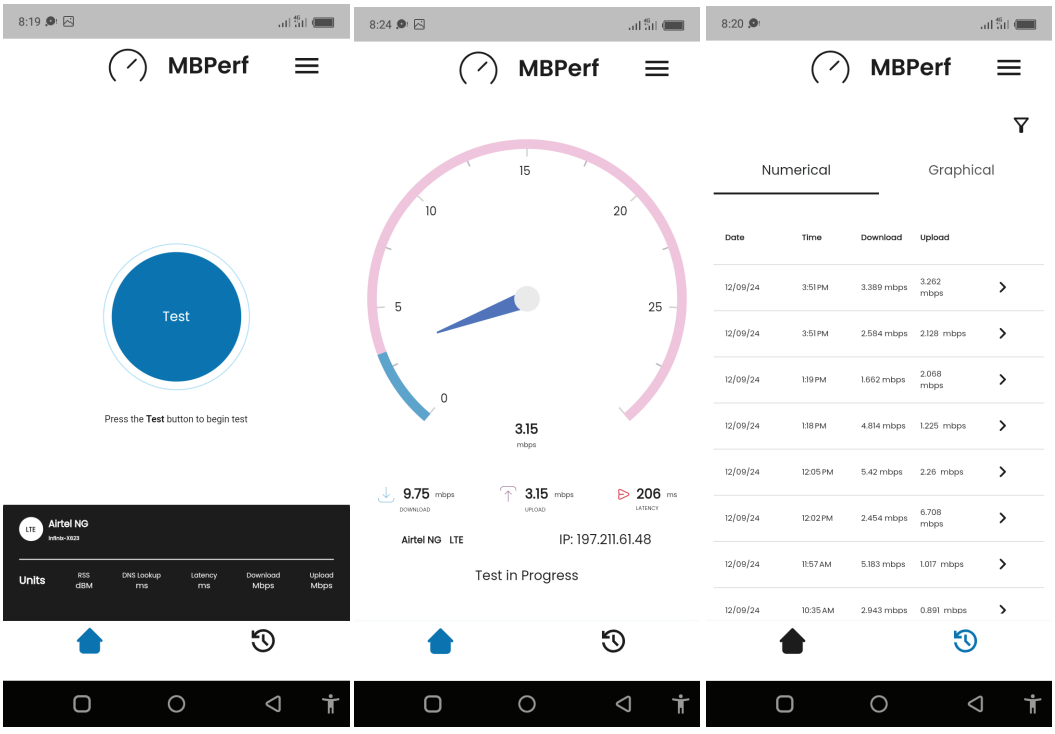


Fig. 11a: MBPerf Home Page Fig. 11b: MBPerf Test Page Fig. 11c: MBPerf Results Page

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

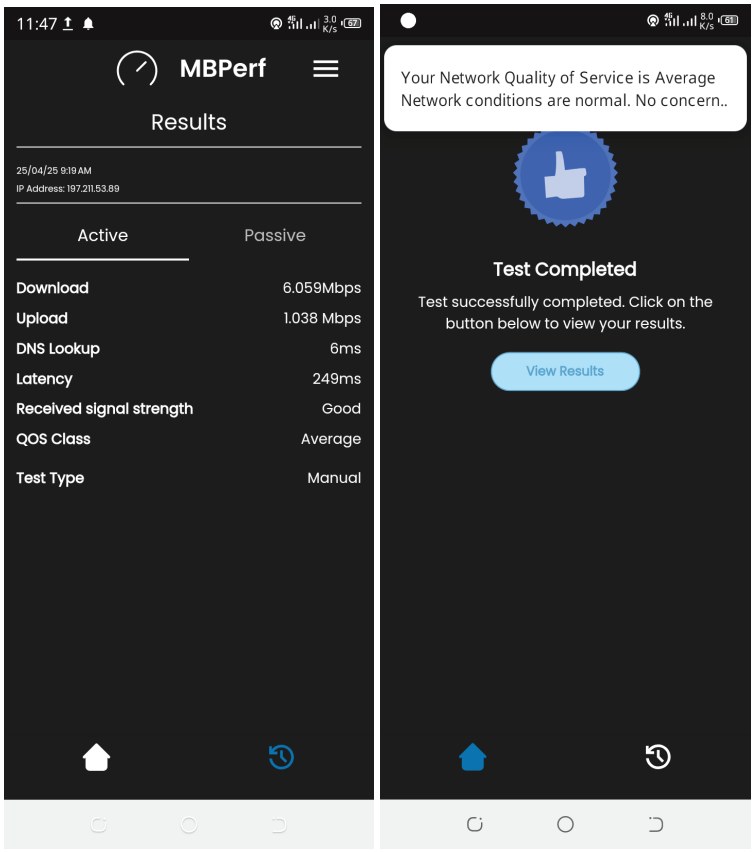


Fig. 12: MBPerf QoS Prediction and Notification

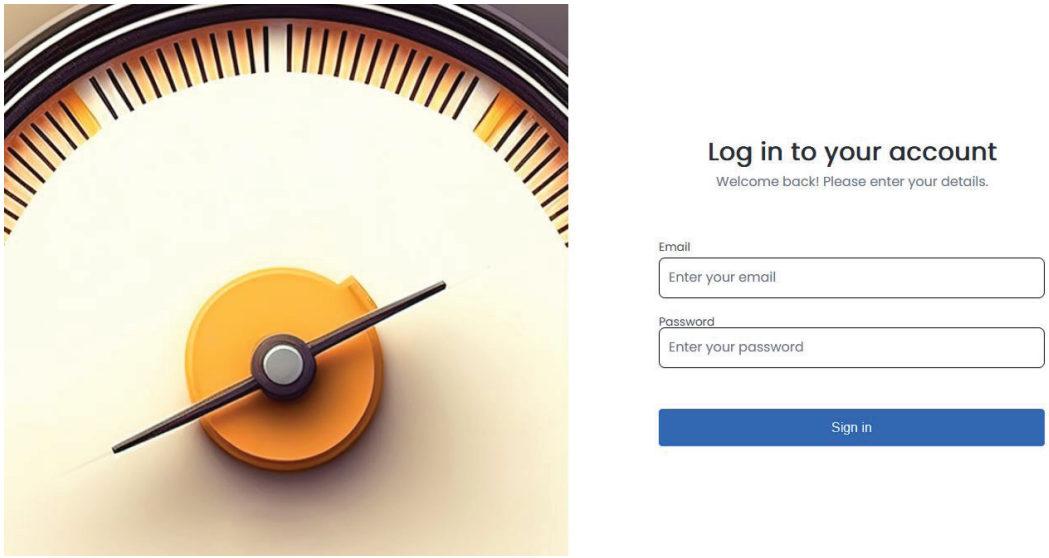


Fig. 13: Login Page

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

Fig. 14: Dashboard Page

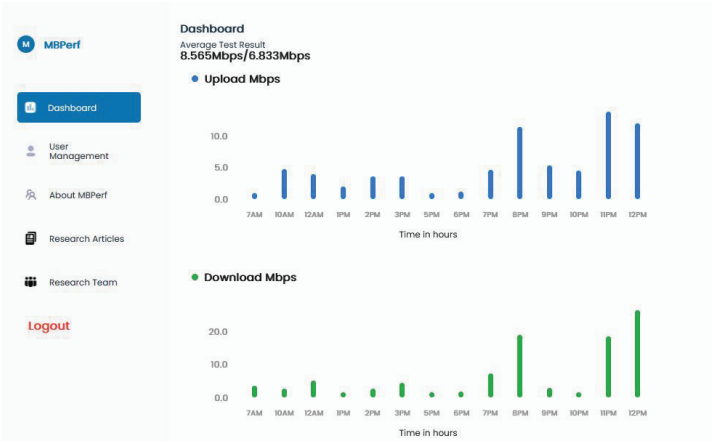


Fig. 15: User Management Page

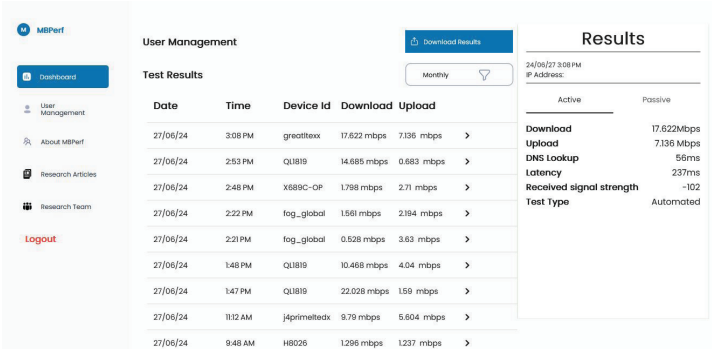


Fig. 16: User Management Page

REFERENCES

[1] A. A. El-Saleh, M. A. Al Jahdhami, A. Alhammadi, Z. A. Shamsan, I. Shayea, and W. H. Hassan, "Measurements and Analyses of 4G/5G Mobile Broadband Networks: An Overview and a Case Study," Wireless Communications and Mobile Computing (Hindawi), vol. 2023, pp. 1–24, Apr. 2023, doi: 10.1155/2023/6205689.

[2] H. Remmert, "When Is 6G Coming, and What Does It Mean for 5G and 4G LTE?" Accessed: May 28, 2024. [Online]. Available: https://www.digi.com/blog/post/when-is-6g-coming-what-does-it-mean-for-5g-4g

[3] O. Theobald, Machine Learning for Absolute Beginners, Second. 2017.

http://dx.doi.org/10.21622/ACE.2025.05.1.1332

[4] A. Chen, J. Law, and M. Aibin, "A Survey on Traffic Prediction Techniques Using Artificial Intelligence for Communication Networks," Telecom, pp. 518–535, Dec. 2021, doi: 10.3390/telecom2040029.

[5] H. Jha and V. Vijayarajan, "Mobile Internet Throughput Prediction using Machine Learning Techniques," in Proceedings of the International Conference on Smart Electronics and Communication (ICOSEC 2020), IEEE Xplore, 2020, pp. 253–257. doi: 10.1109/ICOSEC49089.2020.9215436.

[6] J. Riihijärvi and P. Mähönen, "Machine Learning for Performance Prediction in Mobile Cellular Networks," IEEE Computational Intelligence Magazine, Jan. 2018, doi: 10.1109/MCI.2017.2773824.

[7] I. J. Umoren and S. J. Inyang, "Methodical Performance Modelling of Mobile Broadband Networks with Soft Computing Model," International Journal of Computer Applications, vol. 174, no. 25, pp. 7–21, 2021.

[8] L. Alho, A. Burian, J. Helenius, and J. Pajarinen, "Machine Learning Based Mobile Network Throughput Classification," in 2021 IEEE Wireless Communications and Networking Conference (WCNC), Nanjing, China: IEEE Xplore, 2021, p. 6. doi: 10.1109/WCNC49053.2021.9417365.

[9] G. Sam, P. Asuquo, and B. Stephen, "Customer Churn Prediction using Machine Learning Models," JERR, vol. 26, no. 2, pp. 181–193, Feb. 2024, doi: 10.9734/jerr/2024/v26i21081.

[10] A. De Masi and K. Wac, "Towards accurate models for predicting smartphone applications' QoE with data from a living lab study," Quality and User Experience, Springer Nature, vol. 5, no. 10, pp. 1–18, 2020, doi: https://doi.org/10.1007/s41233-020-00039-w.

[11] M. Ghuge, N. Ranjan, R. A. Mahajan, P. A. Upadhye, S. T. Shirkande, and D. Bhamare, "Deep Learning Driven QoS Anomaly Detection for Network Performance Optimization," J. Electrical Systems, vol. 19, no. 2, pp. 97–104, 2023.

[12] M. I. Smahi, f. Hadjila, C. Tibermacine, and A. Benamar, "A deep learning approach for collaborative prediction of Web Service QoS," SOCA, vol. 15, pp. 5–20, 2021, doi: 10.1007/s11761-020-00304-y.

[13] D. Rahmayanti, "Predicting Quality of Service on Cellular Networks Using Artificial Intelligence," Journal of Engineering, Electrical and Informatics, vol. 5, no. 2, pp. 28–35, 25AD, doi: https://doi.org/10.55606/jeei.v5i2.3901.

[14] Google, "Google Colaboratory," Google colab. Accessed: Nov. 24, 2024. [Online]. Available: https://colab.google/

[15] H. Samulowitz, U. Khurana, and S. D. Turaga, "Learning Feature Engineering for Classification," in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17), Aug. 2017, pp. 2529–2534. doi: 10.24963/ijcai.2017/352.

[16] A. Csorg, G. Martınez-Munoz, and C. Bentejac, "A Comparative Analysis of XGBoost," Feb. 17, 2020, Spain. doi: 10.48550/arXiv.1911.01914.

[17] Ookla, "Speedtest," Speedtest. Accessed: Sep. 23, 2024. [Online]. Available: https://www.speedtest.net/about

[18] ITU-T Study Group, "Communications quality of service: A framework and definitions," Series G: Transmission Systems and Media, Digital Systems and Networks. ITU-T G-Series Recommendations (G.1000), 2001.

[19] Arnab, A.A., Shuvro, A.A., Ma, K.F., and Leung, H. (2023). A Deep Learning Approach for a QoS Prediction System in Cellular Networks. 2023 IEEE 9th World Forum on Internet of Things (WF-IoT), 1-6.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

AN AI-BASED FRAMEWORK FOR IMPROVING EFFICIENCY AND FAIRNESS IN THE INTERVIEW PROCESS

Mohannad Taman[1,3], Yahia Khaled[2,3], Dalia Sobhy[3]

[1] AI Engineer, Obeikan Digital Solutions, Cairo, Egypt.

[2] SOC Analyst, Banque Misr, Cairo, Egypt.

[3] Arab Academy of Science and Technology and Maritime Transport, Alexandria, Egypt.

Emails: {m.taman@obeikan.com.sa, yahiakhaled49@gmail.com, dalia.sobhi@aast.edu }

Received on, 27 April 2025 - Accepted on, 25 May 2025 - Published on, 18 June 2024

ABSTRACT

Artificial intelligence (AI) technologies have advanced to the point where they can assist human resource specialists, such as recruiters, by automating major aspects of the hiring process and efficiently filtering candidate pools. However, limited research has evaluated the effectiveness of AI systems in virtual interviews. This paper presents InstaJob, an AI-powered framework designed to enhance both efficiency and fairness in the hiring pipeline. The system integrates multiple deep-learning components to analyze candidate responses during interviews. Facial emotion recognition is performed using a convolutional neural network (CNN) trained on the FER2013 dataset, achieving a validation accuracy of 77% and outperforming several state-of-the-art approaches. For speech processing, IBM Watson is used to convert spoken responses into text. The transcribed text is then analyzed using EmoRoBERTa, a transformer-based model, to detect emotional signals from verbal content. In addition, IBM Watson is employed to detect filler words and assess speech fluency. These components collectively enable InstaJob to assess candidates' soft skills in a structured and unbiased manner, offering a comprehensive and data-driven evaluation of interview performance.

Index words: Artificial intelligence, virtual interviews, Facial emotion recognition, speech processing, Deep learning applications.

1. INTRODUCTION

Emotion is a complex and dynamic state that arises in response to various stimuli, including experiences, thoughts, or social interactions. It includes subjective feelings, cognitive processing, behavioral responses, physiological changes, and communication cues. Therefore, the ability to identify emotions is essential in a variety of domains, including marketing, human-robot interaction, mental health evaluation, and employment interviews [1, 2]. Using virtual interviews in employment processes has become more common recently.

Virtual video interviews offer several advantages to interviewers and interviewees alike: they enable HR personnel to assess a large number of job applications; (2) they enable HR personnel to review and make decisions offline, and (3) facilitate cost-effective long-distance interviews. However, virtual video interviews pose several challenges, particularly in areas like assessing non-verbal communication, building rapport, and ensuring consistency across candidates. Studies have indicated that

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

in virtual environments, biases related to race, gender, and socioeconomic class May develop [3, 4]. In order to minimize bias by standardizing the interview process and Delivering data-driven insights for more equitable decisions, there is growing interest in the usage of AI-driven tools that objectively analyze applicant responses and non-verbal cues. These technologies can help make virtual interview recruitment fairer by ensuring that hiring decisions are based on qualifications rather than opinions. According to recent surveys [5, 6], AI in recruiting is becoming more widely accepted, with 60% of HR professionals believing it will have a major impact. The global market for AI recruiting is expected to reach $7.1 billion by 2025, highlighting its growing influence in the field. HR professionals can save up to 80% of their time during the hiring process by implementing AI, while candidates who practice using AI-powered interview tools have a higher chance of securing employment opportunities [5, 6]. In this context, the use of automated systems for assessing video interviews to evaluate applicants have grown in popularity in recent years across a range of industries [7].

Hiring managers can save time with these tools, and candidates can schedule interviews at their convenience [8]. Nowadays, automatic video interview assessment systems are in use all over the world. More than 80 million interviews have been assessed according to HireVue is a well-known startup in the automated hiring sector [9]. Emerging applications have made use of social signal processing (SSP) [10] to assist in the study and assessment of interview performance. Nevertheless, emotional cues The voice and face have not been fully investigated in prior studies. This paper addresses the following research question: ***RQ:*** How can artificial intelligence (AI) be applied to ensure the fairness of the applicant's evaluation and the efficiency of the interview process using facial and speech cues?

The remainder of the paper is structured as follows: The related work section presents the literature review related to AI in recruitment and emotion detection techniques. The proposed model section describes the proposed AI-driven framework. The experimental evaluation section provides the set of experiments conducted. Finally, the conclusion section provides conclusions and future work.

RELATED WORK

AI-based recruitment techniques can promote better hiring results, decrease bias, and accelerate the hiring process [11–13]. However, the effectiveness of AI in hiring depends on on the approaches adopted and the implementation environment. This section presents an overview of research on AI-based recruitment practices.

AI IN RECRUITMENT

The use of AI in recruitment has grown significantly over the years. Various studies have explored automated systems for CV parsing, skill assessment, and candidate filtering, with most approaches focusing on text analysis [14, 15]. However, few work have focused on emotional detection from video interviews. In recent years, several recruitment platforms have been proposed [16–18]. For example, a multi-fine-grained method based on recurrent neural networks (RNN) and keyword-question attention mechanisms are proposed for interviews [16]. This method scores different personality characteristics of interviewees, obtaining a comprehensive interview score. The approach leverages a two-stage model learning mechanism and a keyword-question-level attention mechanism to reliably predict personality traits after removing words and phrases that are associated with those traits. This work has focused on determining personality traits based on speech without considering facial emotion recognition. Further, "vRecruit" [17], a machine learning-based web application for virtual recruitment was proposed, which incorporates a client-specific interview process and a text-based sentiment analysis engine. In [18], the authors used CNN to classify a candidate's emotions based on images and nervousness from blink rate during a virtual interview process, with an accuracy of 60%. However, the focus of these works was on

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

evaluating candidates using still images, which may introduce significant bias compared to video interview analysis.

EMOTION DETECTION IN AI

Emotion detection has been a significant area of AI research, especially with the use of deep learning models such as convolutional neural networks (CNN) for facial expression analysis. The 2013 Facial Expression Recognition (FER) Challenge data set (FER-2013 dataset) [19], a widely used benchmark in this field. It provides a diverse range of facial expressions necessary for training and evaluating emotion detection models. The first model, the Local Learning Bag of Words (BOW), was designed for competition application [20]. This was later improved by the Local Learning Deep+BOW model, which combined deep learning features with traditional BOW and was applied to mental disorder detection [21]. In 2020, Ensemble ResMaskingNet, an ensemble learning approach demonstrated the effectiveness of combining multiple CNNs to improve CNN performance [22]. Another CNN-based model from the same year, aimed at enhancing human-computer interaction [23]. By 2022, efforts to improve emotion detection methods using data augmentation with the VGG16 model was introduced, which was used to improve the quality of images in the FER-2013 dataset [24]. However, it achieved a low accuracy as compared to the to previous approaches. In 2023, custom architectures and other methods achieved moderate improvements, with one such custom architecture reaching a considerable accuracy, though the application details were unspecified [25].

Other studies from 2023, reported proposed some models with applications remaining unspecified [26] [27]. To the best of our knowledge, none of these approaches have used facial emotion detection to enhance the recruitment process. In this context, we aim to introduce an novel deep learning model specifically tailored for this purpose. In parallel with facial emotion recognition, text-based emotion recognition has evolved significantly. Models like BERT and EmoRoBERTa have been employed for analyze sentiment and emotional tone in textual data. These models have been particularly useful in assessing emotional intelligence in scenarios such as interviews, where the analysis of both facial expressions and text plays a pivotal role. The evolution of emotion detection models from traditional handcrafted Features extraction techniques to advanced deep learning and ensemble methods underscore the continual advancements in this field. The shift towards deeper architectures and hybrid approaches, along with the integration of multiple datasets, represents a significant leap forward in the accuracy and reliability of emotion detection systems. Ensemble models, in particular, have proven highly effective, combining diverse architectures to achieve superior performance. As emotion detection continues to mature, it will remain a essential component in applications ranging from human-computer interaction (HCI) to personalized content recommendation systems.

2. PROPOSED FRAMEWORK

FULL SYSTEM ARCHITECTURE

This system analyzes video interviews through two parallel paths (Fig. 1). The left path focuses on facial analysis, using HaarCascade for face detection and CNN techniques to assess candidate expressions. The right path processes audio, transcribing it with IBM Watson, then analyzing it for fillers and emotions using IBM Watson and EmoRoBERTa. Each analysis produces a score, which is combined to generate a comprehensive assessment of the candidate's performance.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

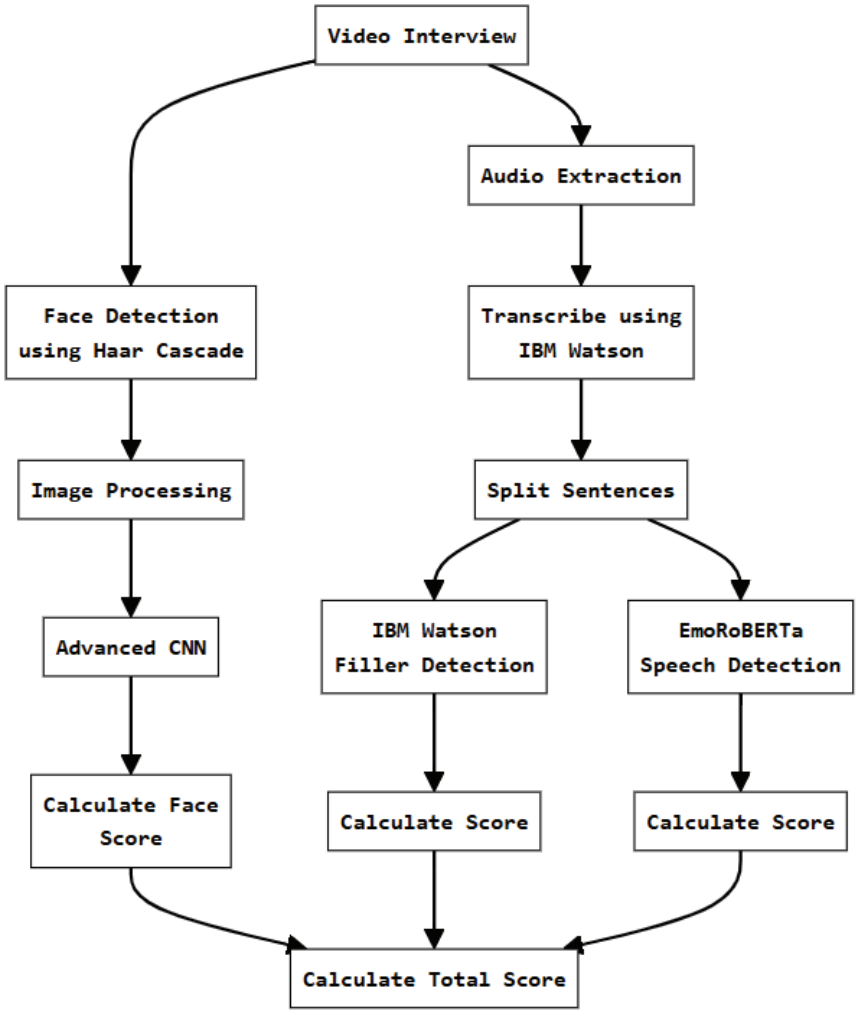


Fig 1. Interview process system Architecture.

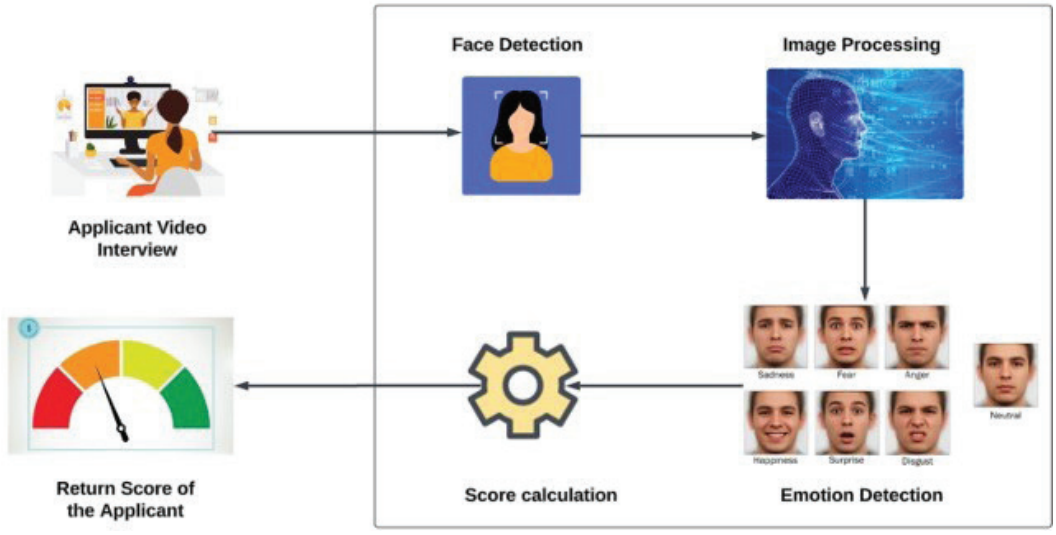


Fig 2. Face Emotion Detection System Architecture.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

3. FACE EMOTION DETECTION

SYSTEM ARCHITECTURE

The system architecture is shown in Fig. 2. The Face Emotion Detection module aims to analyze facial expressions to detect emotions such as happiness, sadness, anger, and neutrality. It provides users with feedback on their emotional responses during video interviews or interactions, enhancing self-awareness and interview performance. This

The analysis is performed offline after the candidate uploads the video. Finally, the recruiter receives a score indicating the candidate's performance.

HAARCASCADE FOR FACE DETECTION

In our video processing pipeline, we utilize haar cascade frontal face default.xml from the OpenCV library for face detection. This pre-trained HaarCascade classifier is specifically designed to identify frontal faces within images or video frames (Fig. 3).

The classifier operates by evaluating patterns of pixel intensities that correspond to the characteristic features of a human face, such as the arrangement of eyes, nose, and mouth. Upon detection, the classifier marks the location of each identified face with a bounding box, effectively isolating facial regions from the rest of the image or frame.

This bounding box ensures that only the essential facial features are extracted as inputs for further processing, such as emotion detection using machine learning models. Integrating HaarCascade face detection enhances the precision and efficiency of our system in capturing and analyzing facial expressions within video streams.

EMOTION DETECTION

After detecting faces using haar cascade frontal face default.xml in the video processing pipeline, the extracted facial regions (bounded by boxes) are passed to the emotion detection model. These regions, containing key facial features (Fig. 4), are preprocessed through resizing and pixel normalization to ensure consistent input formatting. We developed a Convolutional Neural Network (CNN) model for face emotion detection.

The CNN consists of multiple convolutional and pooling layers followed by a fully

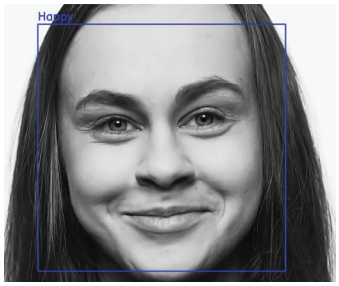


Fig 3. Sample of Face Detection using Haar Cascade.

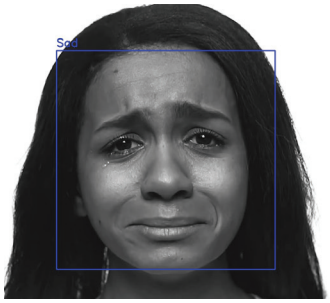


Fig 4. Face Detection Detection.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

Connected layers. The input shape of images is 48x48 pixels, grayscale. The architecture of our CNN model is shown in Fig. 5. The deep learning model then classifies emotions—such as happiness, sadness, anger, surprise, and neutrality—based on these features. By combining Haar Cascade face detection with emotion recognition model, the system efficiently captures and interprets emotional cues from video streams, enabling meaningful real-time insights.

SPEECH ANALYSIS

It consists of three modules. First, the Speech-to-Text Conversion module, which is implemented using the IBM Watson Speech-to-Text API. It inputs the candidate's speech performs some preprocessing (e.g., sentence segmentation using punctuation (.,?)), and then provides the transcribed text. Second, the Speech Emotion Detection the module uses the pre-trained EmoRoBERTa (28 emotion classes). It performs three main processes: contraction expansion (e.g., "can't" → "cannot"), sentence-level emotion classification, and dominant emotion flagging. The third module is Filler Word Detection, which uses regex-based pattern matching. It searches for some fillers (e.g., 'uh,' 'um,' 'like,' etc.) and provides the timestamped filler occurrences. At the end, it calculates a score for speech detection and filler detection.

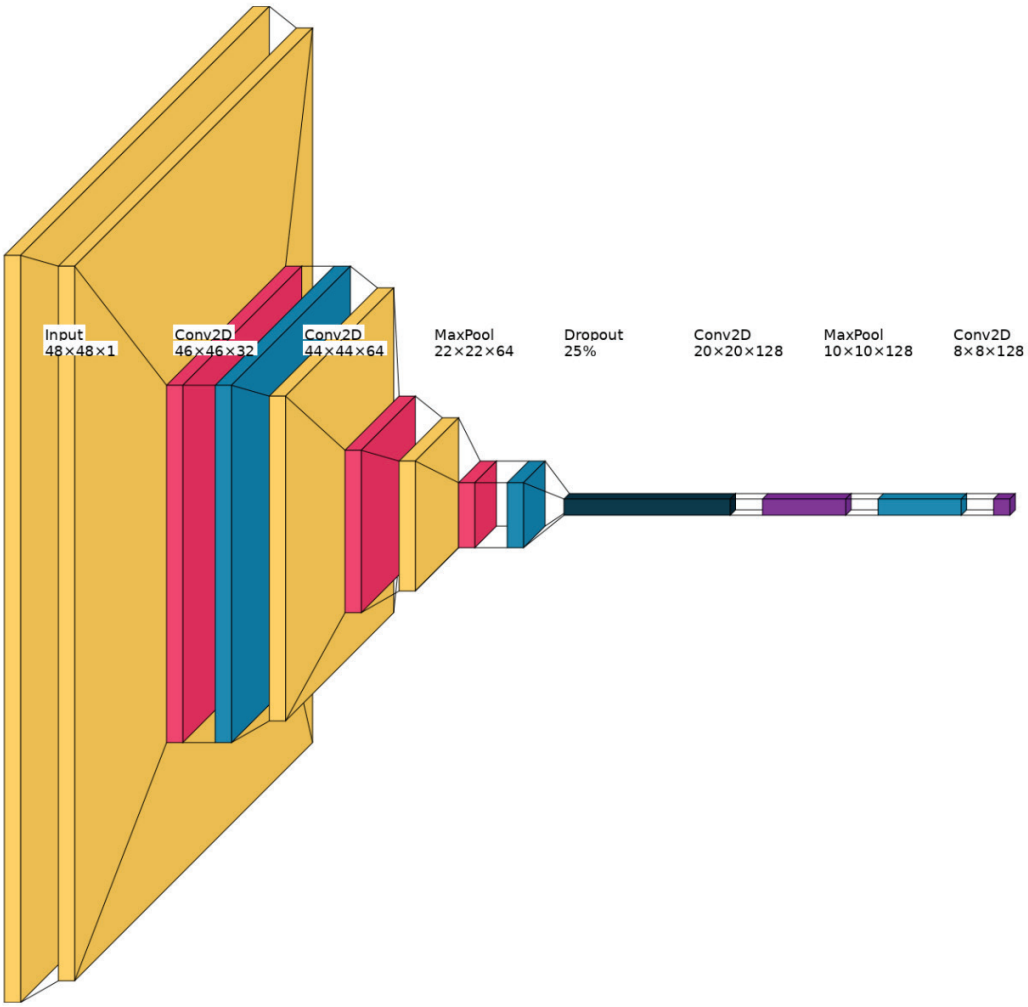


Fig 5. The CNN model for the face emotion detection feature.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

SCORING METHODOLOGY

The score S is calculated as a weighted sum of negative emotion instances (E) and filler word occurrences (F) from three independent models ($S = \sum_{i=1}^{3} w_i \cdot T_i$). T_i representsthe count of triggers (emotions/filler words) for model i , whereas w_i is the weight per 159 models (inverse of triggers needed to increment the score by 1). Table 1 shows the scoring o f 160 rules per model.

Table 1. Scoring rules per model.

Model	Trigger	Score Increment
Face Emotions Detection	4 negative emotions	+1
Speech-to-Text Emotions AI	6 filler words	+1
Text Emotions AI	2 negative emotions	+1

FACE EMOTION DETECTION EXPERIMENTAL EVALUATION

In this section, we will discuss the experimental evaluation.

4. DATASET

The dataset consists of facial expressions obtained from the Facial Expression Recognition 2013 (FER2013) dataset [28], which was provided by Kaggle at the International Conference on Machine Learning (ICML). The FER-2013 dataset contains a set of 35,887 grayscale face images, all standardized to 48×48 pixels. Each image is annotated with one of seven emotion categories: Angry, Disgust, Fear, Happy, Sad, Surprise, or Neutral (Fig. 6). These emotions cover a wide spectrum of affective states that are essential for examining human reactions in interview situations, where nonverbal clues such as facial expressions offer crucial information about applicants' emotional reactions, degrees of engagement, and behavioral indicators.

Fig 6. FER2013 Seven Emotions [28].

The dataset is divided into training, validation, and test sets, facilitating Model development and performance evaluation across different stages. There are 28,709 images in the training set and 3,589 images in the test set. In the training set, there are 4,953 images of anger, 547 images of disgust, 5,121 images of fear, 8,989 images of happiness, 6,077 images of sadness, 4,002 images of surprise, and 6,198 images of neutrality. This imbalance presents an additional challenge in training models for perform accurately across all classes. The dataset's low resolution and grayscale format add further complexity, as models must learn to identify subtle emotional cues from limited visual data.

DATASET PREPROCESSING AND AUGMENTATION

To fit the interview context, data cleaning was essential. Label inaccuracies, particularly in the "happy" class, where images were mislabeled as "angry," were identified, and

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

removed. Additionally, emotions such as fear, surprise, and disgust were excluded, focusing on four expressions: angry, sad, happy, and neutral (Fig. 7). After cleaning, the dataset showed an imbalance in emotional state distribution, with "happy" having the most instances, followed by "neutral" and "sad," while "angry" had the least (Fig. 8). This imbalance could bias the model towards recognizing dominant emotions like happiness and neutrality, potentially affecting its ability to accurately identify less common emotions like anger. Addressing this imbalance is crucial for the model's performance in real interview contexts.

The train data generator is an instance of ImageDataGenerator configured to augment image data during the training phase of our machine learning model. This configuration includes several augmentation techniques aimed at enhancing the model's ability to generalize from the training data. The images are first rescaled to a range of 0 to 1 to normalize the pixel values. During training, the generator randomly applies a rotation of up to 15 degrees, zooms images by up to 10%, performs horizontal flips, and shifts images horizontally and vertically by up to 5% of the total width and height, respectively, using a wrapping fill mode to handle gaps. These augmentations not only help expose the model to a wider variety of image variations, reducing overfitting and improving its ability to learn robust features from the data, but also effectively address the initial class imbalance by generating synthetic samples for under-represented classes like "angry." This approach ensures a more balanced distribution of training instances across different emotional states, thereby enhancing the model's overall performance and generalization capability.

Fig 7. Cleaned FER2013.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

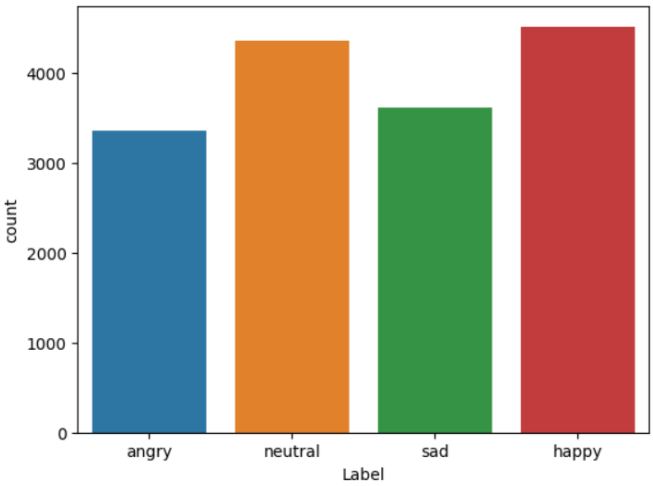


Fig 8. The cleaned FER2013 class distribution.

Fig 9. Examples of the augmented dataset.

5. EXPERIMENTAL EVALUATION

The optimization of the neural network architecture for the FER2013 dataset involved a systematic exploration of hyperparameters and training configurations to maximize model performance. Initial experimental iterations employed traditional optimization algorithms, including Stochastic Gradient Descent (SGD), Adagrad, and RMSprop, coupled with relatively shallow dense layers and limited dropout regularization. These baseline models yielded moderate accuracies, ranging from 52% to 62%, highlighting the necessity for further architectural and training refinements. Subsequently, the focus

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

shifted towards the Adam optimizer due to its adaptive learning rate capabilities and superior convergence properties, which are well-documented in deep learning literature. Progressive adjustments were made by increasing the dense layer capacity to 1024 neurons, incorporating dropout layers with rates of 0.25 and 0.5 to mitigate overfitting and expand the convolutional stack to include four layers with filter sizes of (32, 64, 128, 128), all utilizing ReLU activation functions to introduce non-linearity and improve gradient flow. The training was conducted with a batch size of 64 over 100 epochs, a configuration empirically determined to balance convergence speed and generalization.

This rigorous tuning process culminated in the final model, which achieved a peak accuracy of 77%, representing a significant improvement over the earlier attempts. Additional post-final experiments, which explored alternative activation functions such as LeakyReLU and alternative optimizers like RMSprop at comparable training lengths failed to surpass the established performance threshold, thereby validating the robustness and optimality of the selected hyperparameter set. This comprehensive experimentation underscores the critical role of optimizer selection, network depth, dropout regularization, and training regimen in enhancing the predictive

Table 2. Progressive Model Tuning Experiments for FER2013 Dataset: Optimizer, Layers, and Hyperparameters.

Edition	Neurons	Dropout	Conv Layers (Filter Size)	Training Params	Accuracy
Exp 1	Dense: 64	None	(32), (3,3)	SGD, 64 batch, 10 epochs, ReLU	52%
Exp 2	Dense: 128	0.1	(32, 64), (3,3)	SGD, 64 batch, 20 epochs, ReLU	57%
Exp 3	Dense: 256	0.25	(32, 64), (3,3)	Adagrad, 64 batch, 30 epochs, ReLU	60%
Exp 4	Dense: 256	0.25	(32, 64), (3,3)	RMSprop, 64 batch, 40 epochs, ReLU	62%
Exp 5	Dense: 256	0.25	(32, 64), (3,3)	RMSprop, 32 batch, 60 epochs, Tanh	61%
Initial Adam Model	Dense: 256	None	(32, 64), (3,3)	Adam, 64 batch, 20 epochs, ReLU	65%
Edition 1	Dense: 512	0.25	(32, 64), (3,3)	Adam, 64 batch, 40 epochs, ReLU	68%
Edition 2	Dense: 512	0.25, 0.5	(32, 64, 128), (3,3)	Adam, 64 batch, 60 epochs, ReLU	71%
Edition 3	Dense: 1024	0.25	(32, 64, 128), (3,3)	Adam, 32 batch, 80 epochs, ReLU	73%
Edition 4	Dense: 1024	0.25, 0.5	(32, 64, 128), (3,3)	Adam, 64 batch, 90 epochs, ReLU	75%
Final Model	**Dense: 1024**	**0.25, 0.5**	**(32, 64, 128, 128), (3,3)**	**Adam, 64 batch, 100 epochs, ReLU**	**77%**
Post-Exp 1	Dense: 1024	0.25, 0.5	(32, 64, 128, 128), (3,3)	RMSprop, 64 batch, 100 epochs, ReLU	75%
Post-Exp 2	Dense: 1024	0.25, 0.5	(32, 64, 128, 128), (3,3)	Adam, 64 batch, 100 epochs, LeakyReLU	74%

Accuracy of deep neural networks on the FER2013 dataset.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

6. RESULTS AND DISCUSSION

To evaluate the effectiveness and robustness of our face emotion detection model, we conducted an extensive series of simulations utilizing the FER-2013 dataset. The FER-2013 dataset is well-regarded in the field for its diverse range of facial expressions and comprehensive annotations, making it an ideal benchmark for assessing Model performance. Our simulations were designed to rigorously test the model's ability to accurately identify and classify a wide array of facial emotions across various conditions, including different lighting, angles, and occlusions. Through these simulations, we aimed to visualize and analyze the model's predictions in comparison to the actual ground truth labels. This process not only helps in quantifying the model's accuracy but also in identifying specific instances where the Model excels or struggles. By showcasing these visualizations, we provide a detailed insight into the operational dynamics of our face emotion detection system.

Fig. 10 illustrates sample results from our simulations, demonstrating the model's capability to interpret and categorize facial emotions. Each image is accompanied by the predicted emotion label and the corresponding ground truth, allowing for a clear comparison. These visualizations highlight the practical applicability of our model i real-world scenarios, underscoring its potential for integration into various applications such as AI-powered interviews, mental health assessments, and human-computer interaction systems.

In addition to evaluating the model's accuracy, these simulations also shed light on its limitations and areas for improvement. For instance, certain emotions may be more than challenging to detect under low-light conditions or when the face is partially obscured.

Understanding these limitations is crucial for guiding future enhancements and ensuring that the model performs reliably across all intended use cases. Our model achieved a validation accuracy of 77%, a validation loss of 0.6, a validation precision of 0.82 and a validation recall of 0.71. Table 3 shows that our the model outperforms the other state-of-art approaches.

Table 3. Summary of the state-of-the-art approaches and their performance as compared to the proposed model on the FER2013 dataset.

Study	Year	Method	Accuracy	Application
[20]	2014	Local Learning BOW	67.48%	Competition
[21]	2018	Local Learning Deep+BOW	75.42%	Mental Disorder Detection
[22]	2020	Ensemble ResMask- ingNet	76%	Boost the performance of CNNs
[23]	2020	CNN	70.14%	Enhance Human Computer Interaction
[24]	2022	VGG16	58.6%	Improve Quality of Images in FER2013
[27]	2023	N/A	61.88%	N/A
[25]	2023	Custom Architecture	66.6%	N/A
[29]	2024	Deep CNN	65.68%	VR, Robotics, Marketing and Mental Health
Ours	**2025**	**Advanced CNN**	**77%**	**AI Recruitment**

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

While validation accuracy provides a general measure of the model's performance, precision and recall offer more nuanced insights into its behavior. Precision, the ratio of correctly predicted positive observations to the total predicted positives is crucial in scenarios where the cost of false positives is high. Recall that the ratio of correctly predicted positive observations to all actual positives is essential in situations where it is important to capture as many true positives as possible. These metrics help us understand the trade-offs between different types of errors, guiding us to optimize the model for its intended applications.

Hence, to provide a balanced measure of the model's performance, especially when dealing with imbalanced datasets or when the costs of false positives and false negatives

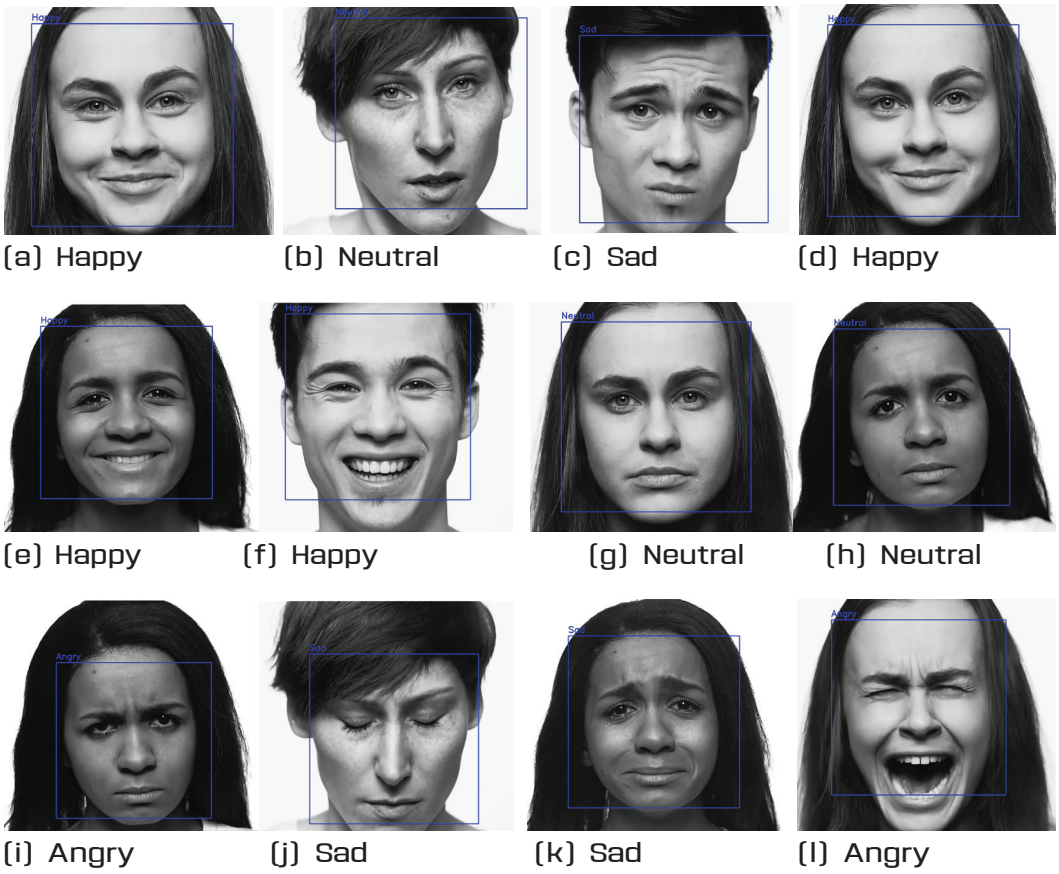


Fig 10. Face Emotion model examples.

Are different, we compute the F1 score. The F1 score is the harmonic mean of precision and recall, calculated as

$$F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = 0.761$$

The F1 score balances precision and recall, offering a single metric that accounts for both types of errors. It is especially important in applications like emotion detection, where both false positives and false negatives can have significant implications. For instance, incorrectly identifying a negative emotion when there is none (false positive) can cause unnecessary concern while missing a genuine negative emotion (false negative) can overlook someone in need of help. Thus, the F1 score provides a reliable metric for assessing the overall effectiveness of our face emotion detection model.

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

IMPLEMENTATION EXAMPLE ON SCORING METHODOLOGY

For a session with eight negative emotions (Face), 12 filler words (Speech), and three negative emotions (Text), the score is computed as follows (with respect to Table 1):

$$S = \left\lfloor \frac{8}{4} \right\rfloor + \left\lfloor \frac{12}{6} \right\rfloor + \left\lfloor \frac{3}{2} \right\rfloor = 2 + 2 + 1 = 5$$

Generally, higher scores reflect cumulative performance issues. The main limitation is that it assumes uniform impact of triggers; calibration may vary by context, where other advanced scoring methodologies will be addressed in the future.

7. CONCLUSION

This paper presents InstaJob, an AI-powered framework that enhances fairness and efficiency in the recruitment process by integrating facial emotion detection, speech analysis, and AI-based technical assessments. The system utilizes datasets such as FER-2013 and GoEmotions to power its models and offer a scalable solution for conducting initial candidate evaluations.

While the platform demonstrates promising results, several limitations must be acknowledged. The variability in video and audio quality can affect the accuracy of the models, and there is a risk of biased outcomes due to cultural differences not fully represented in the training data. Additionally, the virtual interview format may not be perfectly replicate in-person interactions, potentially impacting candidate behavior and model performance.

Future work will focus on improving model generalizability, expanding dataset diversity to reduce cultural and linguistic bias, and testing the platform further in real-world scenarios across various industries. It is also essential to address the ethic considerations of using AI in recruitment by ensuring transparency, fairness, and inclusivity in the decision-making process.

REFERENCES

[1] D. Jekauc et al., "Recognizing affective states from the expressive behavior of tennis players using convolutional neural networks," Knowl Based Syst, vol. 295, p. 111856, Jul. 2024, doi: 10.1016/j.knosys.2024.111856.

[2] S. K. Khare, V. Blanes-Vidal, E. S. Nadimi, and U. R. Acharya, "Emotion recognition and artificial intelligence: A systematic review (2014–2023) and research recommendations," Information Fusion, vol. 102, p. 102019, Feb. 2024, doi: 10.1016/j.inffus.2023.102019.

[3] C. Dando, D. A. Taylor, A. Caso, Z. Nahouli, and C. Adam, "Interviewing in virtual environments: Towards understanding the impact of rapport-building behaviours and retrieval context on eyewitness memory," Mem Cognit, vol. 51, no. 2, pp. 404–421, Feb. 2023, doi: 10.3758/s13421-022-01362-7.

[4] V. Coleman et al., "Lessons Learned From Conducting Virtual Multiple Mini Interviews During the COVID-19 Pandemic," The Journal of Physician Assistant Education, vol. 35, no. 3, pp. 287–292, Sep. 2024, doi: 10.1097/JPA.0000000000000606.

[5] "Artificial Intelligence Insights & Articles | QuantumBlack | McKinsey & Company," 2023. [Online]. Available: https://www.mckinsey.com/capabilities/quantumblack/our-insights/

[6] S. Yadav and S. Kapoor, "RETRACTED ARTICLE: Adopting artificial intelligence (AI) for employee recruitment: the influence of contextual factors," International Journal of System Assurance Engineering and Management, vol. 15, no. 5, pp. 1828–1840, May

2024, doi: 10.1007/s13198-023-02163-0.

[7] C. Fernández-Martínez and A. Fernández, "AI and recruiting software: Ethical and legal implications," Paladyn, vol. 11, no. 1, pp. 199–216, May 2020, doi: 10.1515/pjbr-2020-0030.

[8] E.-R. Lukacik, J. S. Bourdage, and N. Roulin, "Into the void: A conceptual model and research agenda for the design and use of asynchronous video interviews," Human Resource Management Review, vol. 32, no. 1, p. 100789, Mar. 2022, doi: 10.1016/j.hrmr.2020.100789.

[9] HireVue, "About the Company | Leadership & CEO | HireVue." [Online]. Available: https://www.hirevue.com/about

[10] A. Vinciarelli et al., "Bridging the Gap between Social Animal and Unsocial Machine: A Survey of Social Signal Processing," IEEE Trans Affect Comput, vol. 3, no. 1, pp. 69–87, Jan. 2012, doi: 10.1109/T-AFFC.2011.27.

[11] C. Signore, B. Della Piana, and F. Di Vincenzo, "Digital Job Searching and Recruitment Platforms: A Semi-systematic Literature Review," 2023, pp. 313–322. doi: 10.1007/978-3-031-42134-1_31.

[12] A. Fabris et al., "Fairness and Bias in Algorithmic Hiring: A Multidisciplinary Survey," ACM Trans Intell Syst Technol, vol. 16, no. 1, pp. 1–54, Feb. 2025, doi: 10.1145/3696457.

[13] C. Qin, L. Zhang, R. Zha, and D. Shen, "A Comprehensive Survey of Artificial Intelligence Techniques for Talent Analytics," arXiv preprint , 2023, doi: http://dx.doi.org/10.48550/arXiv.2307.03195.

[14] E. T. Albert, "AI in talent acquisition: a review of AI-applications used in recruitment and selection," Strategic HR Review, vol. 18, no. 5, pp. 215–221, Oct. 2019, doi: 10.1108/SHR-04-2019-0024.

[15] M. Kathiravan, M. Madhurani, S. Kalyan, R. Raj, and S. Jayan, "A modern online interview platform for recruitment system," Mater Today Proc, vol. 80, pp. 3022–3027, 2023, doi: 10.1016/j.matpr.2021.06.459.

[16] I. Naim, Md. I. Tanveer, D. Gildea, and M. E. Hoque, "Automated Analysis and Prediction of Job Interview Performance," IEEE Trans Affect Comput, vol. 9, no. 2, pp. 191–204, Apr. 2018, doi: 10.1109/TAFFC.2016.2614299.

[17] S. Mhadgut, N. Koppikar, N. Chouhan, P. Dharadhar, and P. Mehta, "vRecruit: An Automated Smart Recruitment Webapp using Machine Learning," in 2022 International Conference on Innovative Trends in Information Technology (ICITIIT), IEEE, Feb. 2022, pp. 1–6. doi: 10.1109/ICITIIT54346.2022.9744135.

[18] A. K. A, A. H, N. P. Nair, V. A, and A. T, "Interview Performance Analysis using Emotion Detection," in 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA), IEEE, Sep. 2022, pp. 1424–1427. doi: 10.1109/ICIRCA54612.2022.9985667.

[19] I. J. Goodfellow et al., "Challenges in Representation Learning: A Report on Three Machine Learning Contests," 2013, pp. 117–124. doi: 10.1007/978-3-642-42051-1_16.

[20] I. J. Goodfellow et al., "Challenges in Representation Learning: A report on three machine learning contests," arXiv:1307.0414 [cs, stat], May 2013, [Online]. Available: https://arxiv.org/abs/1307.0414

[21] M.-I. Georgescu, R. T. Ionescu, and M. Popescu, "Local Learning With Deep and Handcrafted Features for Facial Expression Recognition," IEEE Access, vol. 7, pp. 64827–64836, 2019, doi: 10.1109/ACCESS.2019.2917266.

[22] L. Pham, T. H. Vu, and T. A. Tran, "Facial Expression Recognition Using Residual Masking Network," in 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, Jan. 2021, pp. 4513–4519. doi: 10.1109/ICPR48806.2021.9411919.

[23] A. Jaiswal, A. Krishnama Raju, and S. Deb, "Facial Emotion Detection Using Deep Learning," in 2020 International Conference for Emerging Technology (INCET), IEEE, Jun. 2020, pp. 1–5. doi: 10.1109/INCET49848.2020.9154121.

[24] J. Luo, Z. Xie, F. Zhu, and X. Zhu, "Facial Expression Recognition using Machine Learning models in FER2013," in 2021 IEEE 3rd International Conference on Frontiers Technology

http://dx.doi.org/10.21622/ACE.2025.05.1.1317

of Information and Computer (ICFTIC), IEEE, Nov. 2021, pp. 231–235. doi: 10.1109/ICFTIC54370.2021.9647334.

[25] S. Hassan, M. Ullah, A. S. Imran, and F. A. Cheikh, "Attention-Guided Self-supervised Framework for Facial Emotion Recognition," 2024, pp. 294–306. doi: 10.1007/978-981-99-7025-4_26.

[26] Y. Wu, L. Zhang, Z. Gu, H. Lu, and S. Wan, "Edge-AI-Driven Framework with Efficient Mobile Network Design for Facial Expression Recognition," ACM Transactions on Embedded Computing Systems, vol. 22, no. 3, pp. 1–17, May 2023, doi: 10.1145/3587038.

[27] Y. Mao, "Optimization of Facial Expression Recgonition on ResNet-18 using Focal Loss and CosFace Loss," in 2022 International Symposium on Advances in Informatics, Electronics and Education (ISAIEE), IEEE, Dec. 2022, pp. 161–163. doi: 10.1109/ISAIEE57420.2022.00041.

[28] Dumitru, I. Goodfellow, W. Cukierski, and Y. Bengio, "Challenges in Representation Learning: Facial Expression Recognition Challenge," 2013. [Online]. Available: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge

[29] D. Bhagat, A. Vakil, R. K. Gupta, and A. Kumar, "Facial Emotion Recognition (FER) using Convolutional Neural Network (CNN)," Procedia Comput Sci, vol. 235, pp. 2079–2089, 2024, doi: 10.1016/j.procs.2024.04.197.

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

FOG COMPUTING-ENABLED SMART SEATING SYSTEMS: OPTIMIZING LATENCY AND NETWORK BANDWIDTH EFFICIENCY IN CLASSROOMS

Evans Obu[1], Michael Asante[2] and Eric Opoku Osei[3]

[1]Department of Computer Science and Engineering, SRID, UMaT, Tarkwa, Ghana

[2]Department of Computer Science, CoS, KNUST, Kumasi, Ghana

[3]Department of Computer Science, CoS, KNUST, Kumasi, Ghana

Emails: {eobu@umat.edu.gh, mickasst@yahoo.com, eric.opoku.osei@knust.edu.gh}

Received on, 06 May 2025 - Accepted on, 03 June 2025 - Published on, 19 June 2025

ABSTRACT

In modern educational settings, overcrowded classrooms challenge student engagement and learning efficiency. To address these issues, we propose a novel smart seating system powered by Fog Computing that leverages Wireless Sensor Networks (WSN), Internet of Things (IoT), Fog Computing (FC), and Cloud Computing (CC) technologies. Our work introduces the first fog computing-driven smart seating system for classroom settings. It demonstrates significant improvements in latency (3.29 ms in Fog-based vs. 108.69 ms in cloud-based systems) while maintaining comparable network efficiency. Our findings highlight fog computing's potential to transform real-time classroom management. Using iFogSim, we conducted a comparative study between traditional cloud-centric architectures and our fog-based system across various classroom scenarios. Results demonstrate that the fog-based architecture delivers superior real-time responsiveness, making it particularly suitable for dynamic educational environments. This research provides both technical insights into performance improvements and practical implementation guidelines for educational institutions seeking to optimize classroom management systems.

Index words: Cloud Computing (CC), Fog Computing (FC), iFogSim, Latency, Network efficiency, Smart Seating System.

1. INTRODUCTION

University enrolment has been progressively increasing over the years. While this is a positive development, it also brings new challenges, including challenges with event attendance and seating infrastructure [1]. One common issue faced by several institutions is the lack of adequate seating spaces in lecture halls. Traditional seating arrangements often fall short, as they lead to distractions that can affect both students and lecturers [2], overcrowded classrooms, and time wastage as students search for available seats. In some cases, these concerns even discourage attendance completely. They could disrupt the learning process and, as such, highlight the need for smarter, more adaptable classroom solutions that can keep pace with growing student populations and evolving educational demands.

This study introduces a novel fog-based smart classroom seating system and uses iFogSim to simulate our architecture. It is designed to improve the management of lecture hall spaces. By integrating technologies like WSNs, IoT, CC, and, most notably, Fog Computing, the system offers a more responsive and efficient way to organize classroom seating. The idea

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

is to move beyond fixed arrangements and towards an intelligent system that can detect available seats, respond to occupancy in real-time, and assist both students and lecturers in making better use of the space while enhancing the speed and network bandwidth usage. Out of the various available simulation tools reviewed by [3], iFogSim has gained enormous attention from many Fog Computing researchers. [4], discuss the various components of the iFogSim to assist researchers in implementing various scenarios of fog computing. [5], For instance, I have modeled and simulated smart surveillance systems in two environments, Cloud Only Network and Fog-Based Cloud Network, using iFogsim.

Our goal is to tackle the limitations found in cloud-based smart systems, chiefly problems with latency, network congestion, and real-time performance. By using fog computing, which brings computation closer to the devices collecting data, we aim to create a faster, more reliable, and energy-efficient solution. This system will improve technical performance and create a smoother classroom experience for stakeholders.

The following research questions serve as a guide to this work:

1. How can a fog computing-based architecture be designed to detect and manage classroom seat occupancy in real-time?

2. What are the measurable performance differences (in terms of latency and bandwidth usage) between fog-based and cloud-based smart seating systems in simulated classroom environments?

3. How does the proposed fog-based system improve responsiveness and scalability compared to traditional cloud-based solutions?

There has been a lot of talk about smart classrooms in recent years. However, there is a lack of studies on how fog computing could be used specifically for seating management in the classroom. This study fills that gap by exploring a practical, tech-driven approach to an everyday problem in higher education. We unite emerging technologies with real-world classroom needs and hope to contribute a solution that improves technical efficiency and also enhances the learning environment in a meaningful way.

1.1. RELATED WORKS

Generally, seating arrangements, whether smart or traditional, have been proven to influence student engagement and behavior. Some researchers have explored this premise.

For example, a study by [6], utilised wearable physiological sensors to examine the impact of individual and group sitting experiences on student engagement and participation. Their findings indicate that, students who were seated in close proximity to each other exhibited greater physiological synchrony, and hence an enhanced cooperative participation. This means that students' interaction and attention levels may be influenced by traditional seating arrangements such as rows or table groups.

Also, in a study on the impact of different seating arrangements (i.e. circles vs rows) on the interaction levels among university students, [7] used wearable Sociometric badges to study speech rate & speaking segment length. They noted that sitting in rows led to more intensive interactions than sitting in circles. They however noted that, the field of study and the facilitator's involvement also had an effect on the result. These findings stress the importance of deliberate seating configurations to improve learning outcomes.

Smart classrooms have introduced dynamic seating techniques designed to enhance situational engagement. A work published by [8] in Sustainability (2023) indicate that, students positioned at the periphery of smart classrooms often exhibit lower engagement levels, as compared to those seated in central locations. The study advocates for deliberate seating configurations to minimize the use of peripheral seats, thereby promoting equitable participation.

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

To also minimise student distractions, innovative proposals such as fuzzy logic-based seating configurations have been presented by some researchers. A 2025 study by [9] introduced "CUB", a Fuzzy Inference-Based Tool for Classroom Seating Arrangement to Minimize Distraction, which actually resulted in a reduction in classroom distractions.

Building upon these insights, this study proposes a novel fog computing-enabled smart seating system that not only considers seat availability but also enables real-time and automated seat allocation based on sensor-driven occupancy detection. Unlike traditional seating systems or standalone AI tools, our system integrates edge-level processing (via fog nodes) with real-time input from classroom IoT devices (e.g., cameras and microcontrollers). This makes room for immediate feedback, equitable seating optimization, and scalability across multiple classrooms and hence bridges the gap between theoretical seating strategies and fully functional, dynamic seat management systems.

Recent research has shown that smart seating systems have the potential to improve classroom management, enhance student engagement, and promote more inclusive learning experiences [1], [10], [11]. Most of these works have delved into how these smart seating systems can be enhanced using cloud computing, mostly for storing, processing, and analyzing classroom data [12]. Some other studies have explored how these systems impact the dynamics of classroom interaction, student participation, and even the effectiveness of instructors [1], [2]. This research demonstrates the strengths and limitations of our proposed fog-based smart seating solutions compared to cloud-based designs in academic settings.

Currently, fog computing has gained traction as a powerful approach for managing resources in distributed systems. Unlike traditional cloud computing, where data is processed in distant data centers, fog computing brings processing closer to the source where the data is generated [13], [14]. This allows for faster data processing, reduced latency, and increased system responses. This makes fog computing particularly useful in situations that demand real-time interaction, such as smart seating in lecture halls.

Because Fog Computing places processing power closer to users, it shortens the distance data needs to travel. This leads to faster responses and smoother system performance [14], [15]. Additionally, fog nodes can handle tasks locally, which implies that less data needs to be sent to the cloud. This not only reduces network load but also cuts down on energy usage [14]. Also, the decentralized nature of fog computing allows institutions to scale up easily by adding more local nodes where necessary [16]. This makes it a great fit for expanding smart seating systems.

Previous research on smart classroom technologies has focused on enhancing student engagement through interactive displays and personalized learning environments [17], [18] but has often overlooked the fundamental logistical challenges of classroom management. Studies by [1] and [19] explored IoT-enabled classrooms but employed cloud-centric architectures that inherently suffer from latency issues unsuitable for real-time applications. Our work extends these efforts by specifically examining the computational advantages of fog computing for addressing the practical challenge of seat allocation in overcrowded lecture halls. Unlike [19] and [20], which primarily discussed the theoretical benefits of fog computing in educational contexts, our work provides empirical evidence through simulation-based performance comparisons between fog and cloud architectures. This empirical focus represents a significant advancement over existing literature, which has largely remained conceptual or limited to small-scale proof-of-concept implementations.

Despite all the recent interest in fog computing, there is still a noticeable gap in research on its application in educational environments. Very few studies have directly compared fog based systems with traditional cloud-based ones exclusively in the context of smart classroom seating [19]. While individual studies point to fog computing's advantages, which include reduced latency, better energy usage, and improved scalability, there is a lack of head-to-head comparisons under similar conditions [19]. These kinds of comparative studies are crucial. They provide solid evidence on whether fog really outperforms cloud systems in educational settings and also help guide the design of the next generation of smart seating solutions [20].

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

2. METHODS

We explore the design and performance evaluation of a smart seating system powered by Fog Computing (FC), Wireless Sensor Networks (WSN), Cloud Computing (CC), and Internet of Things (IoT) technologies.

Simulations are conducted using iFogSim to assess the effectiveness of the proposed fog-based smart seating system over traditional cloud based systems. The simulation models a university lecture hall environment, where data is collected in real-time from cameras embedded in the lecture halls and processed by nearby fog nodes.

The simulation environment was configured as follows: Each classroom was modeled with varied numbers of sensors (1-5 cameras). Each sensor generated data at 30 frames per second, with 1080p resolution. Processing requirements were considered with respect to image analysis algorithms requiring approximately 4800 MIPS (for Cloud devices), 2800 MIPS (for Proxy devices), and 500 MIPS (for Classroom-level devices) per frame. Network bandwidth was configured according to typical educational institution specifications (i.e., 100 Mbps uplink, 10 Gbps downlink).

The simulation was run about 10 times for each configuration to ensure consistency and statistical validity, with each run representing a 60-minute classroom session. Statistical analyses included Mean, Median, and independent t-tests and their related p-values for both latency and bandwidth utilization comparisons. This helped to determine the statistical significance of these differences, particularly in latency, which is critical for real-time applications. Sensitivity analysis was also performed to assess the system's robustness under varying workloads and network conditions, ensuring the reliability of the results.

The findings demonstrate that the fog-based system significantly outperforms the cloud-based system in terms of latency while maintaining comparable network usage. This suggests that fog computing offers substantial advantages in the real-time management of smart classroom environments. However, the study also acknowledges limitations related to the simulation environment and the scalability of the proposed system, suggesting areas for future research. Table 1 presents a Summary of the Configuration Setup for this simulation.

Table 1. Summary of Configuration Setup

Component	Description
CPU	Intel Core i7
RAM	16 GB
Operating System	Windows 11
Simulation Tool	iFogSim
Programming Language	Java (for iFogSim simulations)
Network Bandwidth	Uplink: 100 Mbps, Downlink: 10 Gbps
Fog Device Processing Power	4800 MIPS (Cloud); 2800 MIPS (Proxy); 500 MIPS (Classroom-level devices)
Edge Node Components	IoT devices (e.g., environmental sensors, cameras), Raspberry Pi, Mobile Phones
Latency Configurations	Proxy-to-Cloud: 100 ms; Sensor-to-Fog: 1 ms
Sensors	Cameras, PTZ Controls, Environmental Sensors. (Generated data at 30 frames per second with 1080p resolution)
Fog Node Components	Proxy servers, routers, smart cameras
IoT Framework	Sensor data transmitted to Fog and Edge nodes
Deployment Configuration	Cloud deployment (for comparative analysis), Fog deployment (for proposed framework evaluation)

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

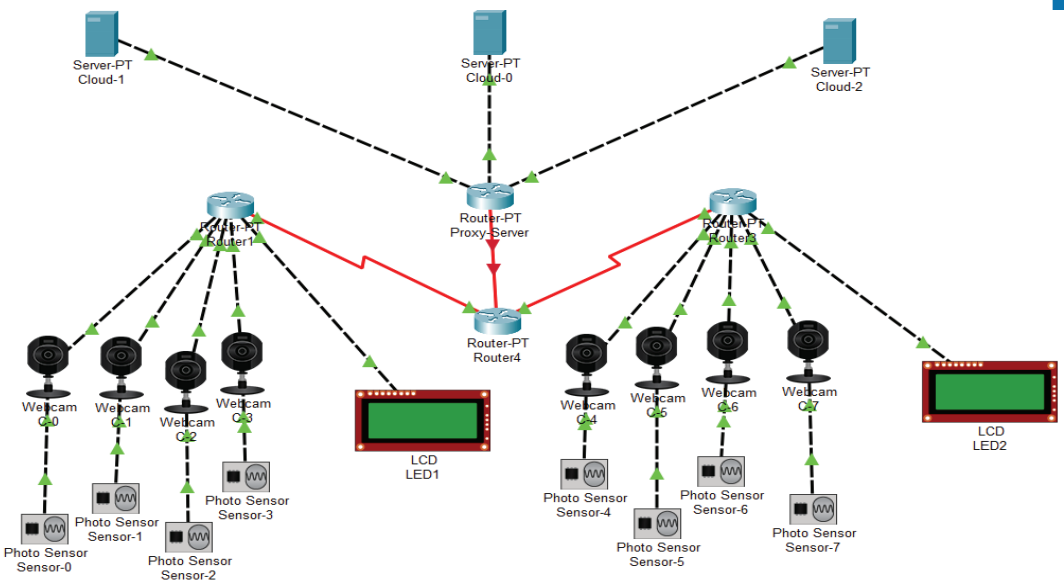


Figure 1. A logical design of a fog-based smart lecture hall

2.1.1. THE FUNCTION AND POSITION OF THE CLOUD IN THE PROPOSED SYSTEM.

In our proposed smart seating architecture, the cloud server is strategically positioned as a centralized platform for storage and analytics. The deployed Fog nodes manage real-time processing tasks, including seat detection and user interface updates in the classroom, whereas the cloud facilitates long-term data archiving, historical analytics, system-wide monitoring, and remote administrative access. Cloud servers analyze seating usage trends over time, produce reports for institutional decision-making, and enable remote updates to system software.

The cloud infrastructure is presumed to be located off-premises, potentially at a university-operated data center or with a commercial, public cloud provider such as AWS or Google Cloud in a nearby regional data zone. This geolocation is critical as it introduces network latency when handling time-sensitive operations like real-time seat detection. Due to this physical separation, it is projected that round-trip delays can exceed 100 milliseconds, as confirmed in our simulations. In contrast, fog nodes positioned locally within campus networks drastically reduce this latency, enabling near-instantaneous feedback and responsiveness. Therefore, fog computing is intentionally prioritized for latency-sensitive tasks, while the cloud is retained for its strength in storage capacity, backup, and system-wide coordination.

2.1.2. GLOBAL VIEW OF PROPOSED SMART SEATING SYSTEM FOR LECTURE HALLS

Implementing the proposed architecture involves five major layers, as presented in Figure 3 (i.e., the cyber-physical layer, the data management layer, the data processing layer, the domain application layer, and the cloud). These layers can be further summarized into three (3) as presented in Figure 4 (i.e., the Cyber-physical layer, Fog-layer, and the Cloud layer). The cyber-physical layer consists of various sensors like GPS, RFID tags, and surveillance cameras, which enable data collection from multiple sources. IoT technology facilitates direct interactions among these sensors, routers, and gateways. Data management, positioned between the cyber-physical and data processing layers, handles tasks such as data description and fusion to manage collected data efficiently, removing redundancy and integrating data for consistency. Central to fog-based computing, the data processing layer processes data types via fog computing, handling tasks locally and transferring overloaded tasks to cloud data centers when necessary. The domain application layer provides specific intelligent applications and services, such as guiding students to available seats in smart seating systems and enhancing efficiency and convenience. Figure 2 and Figure 3 present

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

the "Detailed Hierarchical structure of fog computing-based smart seating system" and the "Condensed high-level Hierarchical structure of fog computing-based smart seating system," respectively.

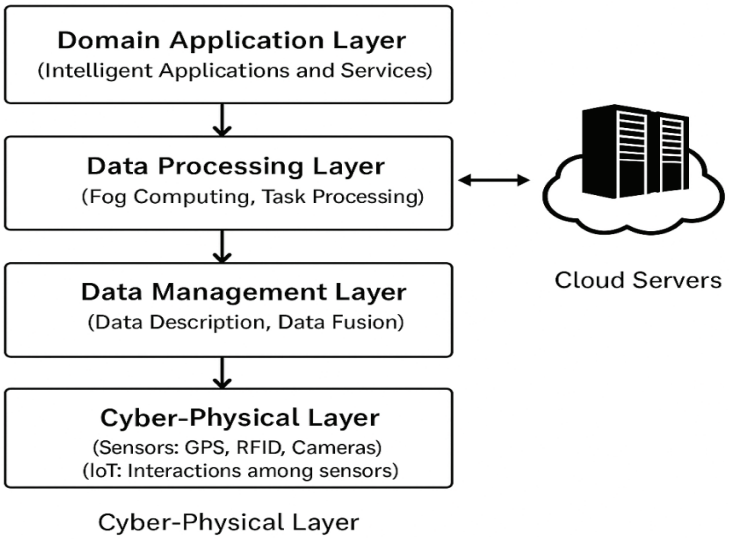


Figure 2. Detailed Hierarchical structure of fog computing-based smart seating system

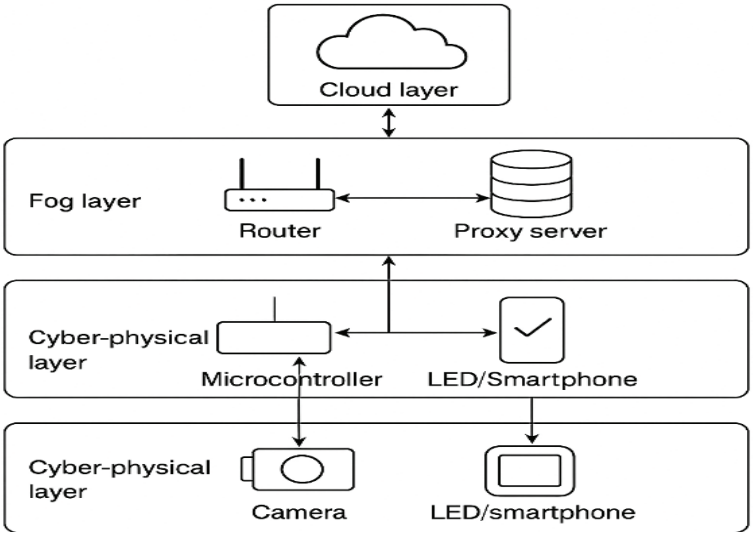


Figure 3. Condensed high-level Hierarchical structure of fog computing-based smart seating system.

2.1.3. IMAGE PROCESSING WORKFLOW FOR SEAT DETECTION

Although the simulation environment in this study (iFogSim) does not support direct integration of image processing modules, the proposed fog-based smart seating system is designed to conceptually support real-time computer vision tasks. The system assumes that each classroom is equipped with IP surveillance cameras capable of capturing video frames at 1080p resolution and 30 frames per second. The captured frames are processed locally at the fog nodes to detect available or occupied seats using a lightweight object detection pipeline.

A typical image processing workflow for seat detection would involve the use of pre-trained object detection models, such as YOLOv5 or YOLOv7, which are capable of detecting chair outlines and distinguishing occupancy based on motion and form, as discussed in detail by

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

[21]. The first step involves frame acquisition and preprocessing (e.g., resizing, normalization, and background subtraction). Following this, real-time object detection algorithms are applied to identify human presence in proximity to predefined seat locations. Motion detection and frame differencing can also be employed to enhance accuracy, especially in low-light or static conditions.

Thresholding techniques are then applied to determine occupancy. For instance, if a seat region contains a detected object (i.e. a person) for more than a specific number of consecutive frames, the seat is classified as "occupied"; otherwise, it is marked as "available." This processed information is transmitted to the fog node, which handles further decision-making and updates the smart classroom interface in real-time.

In iFogSim, this behavior is abstracted by modeling data generation at the sensor level and processing tasks at fog nodes using AppModules. The modules simulate the computational demand of image processing (configured at 500 MIPS per frame for classroom-level devices) while omitting the image content itself. The abstraction ensures scalability testing and performance benchmarking without actual video processing yet aligns conceptually with how the system would operate when deployed with real image analytics capabilities. Figure 4 presents a pictorial description of the Image Acquisition workflow for seat detection in this study.

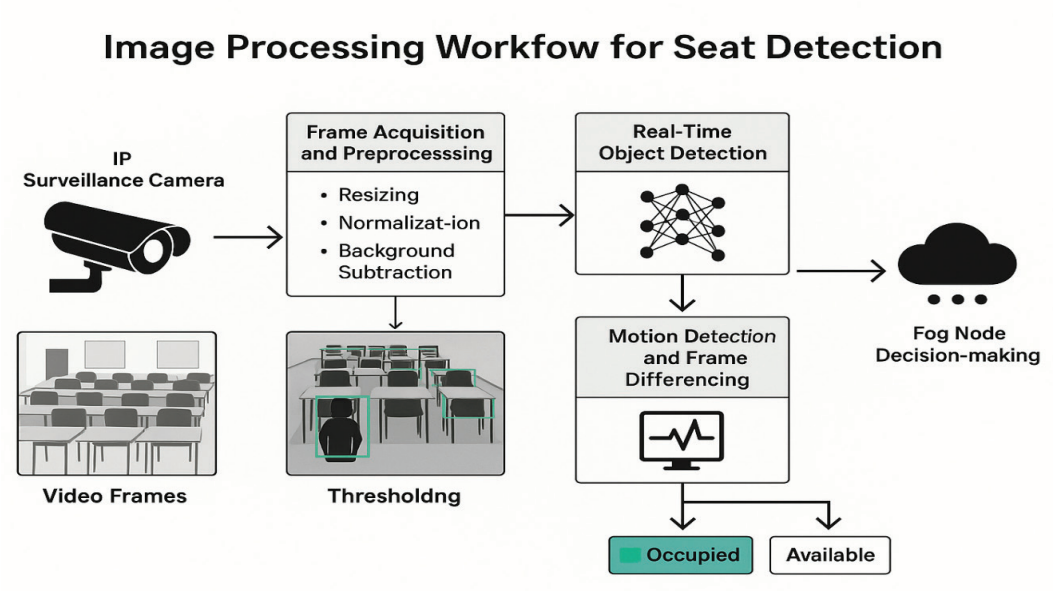


Figure 4. Image Acquisition workflow for seat detection

3. SIMULATION SET UP FOR SMART SEATING SYSTEM IN IFOGSIM

Ifogsim comprises three primary components, i.e., physical, logical, and management components. These components are discussed below. The codes used in creating the various components can be found on the authors' GitHub repository at:

https://github.com/NobleITSoultions/SmartCampusContract2024/blob/main/SmartSeatingSystemSimulation.

A. PHYSICAL COMPONENT

The physical components include all fog devices (fog nodes). The fog devices are made in a hierarchical order. The lower fog devices are directly connected to the sensors and actuators. Fog devices act as data centers in the cloud computing paradigm by offering memory, network, and computational resources. Each fog device must have specific

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

parameters such as processing power, memory capacity (MIPS), computational power, uplink and downlink bandwidth.

Physical components must be created with specific parameters, including RAM, processing capability in a million instructions per second (MIPS), cost per million instructions processed, uplink and downlink bandwidth, and busy and idle power, along with the hierarchical order. While creating the lower-level fog devices, the associated IoT devices, sensors, and actuators need to be created. The particular value in the transmit Distribution object that is set in creating an IoT sensor refers to its sensing interval. In addition, the creation of sensors and actuators requires the reference of application ID and broker ID.

B. LOGICAL COMPONENT

Application modules (AppModules) and Application edges (AppEdges) are the logical components of iFogSim. AppModules represent the application components or modules of the fog computing application. These modules can include different tasks or functionalities that must be executed within the fog computing application. AppEdges represent the communication link or edges between different application modules within fog computing applications. They define the data flow and communication patterns between various modules, capturing how data is exchanged among application parts.

Logical components such as AppModule, AppEdge, and AppLoop are required to be created. While creating the AppModules, their configurations are provided, and the AppEdge objects include information regarding the tuple's type, its direction, CPU, and networking length, along with the reference of the source and destination module. In the background, distinct types of tuples are created based on the specifications given for AppEdge objects.

C. MANAGEMENT COMPONENTS

The management component of iFogSim consists of the Controller and Module Mapping objects. This is responsible for managing and coordinating various aspects of the simulated fog computing environment, overseeing the execution of applications, and handling the dynamic nature of resources in fog computing. These functionalities include resource management, task scheduling, communication management, monitoring and logging, and fault tolerance.

Management Components (Module Mapping) are initiated to define different scheduling and AppModule placement policies. Users can consider total energy consumption, service latency, network bandwidth usage, operational cost, and device heterogeneity while assigning AppModules to Fog devices. They can extend the abstraction of the Module Mapping class accordingly. Based on the information of AppEdges, the requirements of an AppModule need to be aligned with the specification of the corresponding tuple type and satisfied by the available Fog resources. Once AppModules and Fog devices are mapped, the information on physical and logical components is forwarded to the Controller object. The Controller object later submits the whole system to the CloudSim engine for simulation.

The simulation results are compared in terms of latency and network usage. The **"private static boolean CLOUD = false;"** section of the code helps us achieve this result by assigning the values true or false, as it is directly linked to the cloud or fog. Depending on what execution is taking place, the data collected is sent directly to the cloud or fog nodes for processing.

Our simulation framework was specifically designed to replicate real-world classroom conditions. The selection of parameters such as camera frame rates (30 fps) and the resolution (1080p) was based on typical specifications of commercial surveillance systems used in educational settings. The processing requirements (i.e., 4800 MIPS for Cloud devices, 2800 MIPS for Proxy devices, and 500 MIPS for Classroom-level devices) were calibrated to accurately reflect the computational demands of real-time image processing algorithms

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

needed for seat detection. These parameters were validated through preliminary testing to ensure they represented a realistic operational environment. This approach allows our findings to be directly applicable to practical implementations in university lecture halls.

4. RESULTS

The simulation was performed for both Fog-based and cloud-based smart seating systems to compare their performance in lecture hall environments. The experiments focused on evaluating key performance metrics, including latency and network bandwidth usage efficiency. Table 2 summarizes the simulation results.

Table 2. Simulation results

	Latency (Milliseconds, ms)		Network Bandwidth Usage (Kilobytes, kb)	
Number of cameras	Average Fog Latency for all 10 simulations	Average Cloud Latency for all 10 Simulations	Average Fog Bandwidth Usage for all 10 Simulations	Average Cloud Bandwidth Usage for all 10 Simulations
1	3.2	108.64	41553.8	41559.0
2	3.3	108.67	83107.6	83118.0
3	3.3	108.69	124656.4	124677
4	3.309	108.72	166205.2	166236.0
5	3.319	108.75	207754.0	207795.0

5. GRAPHICAL INTERPRETATION OF RESULTS

Based on the results presented in Table 2, Figure 5 and Figure 6 are graphical representations of the simulation results for latency and network usage, respectively.

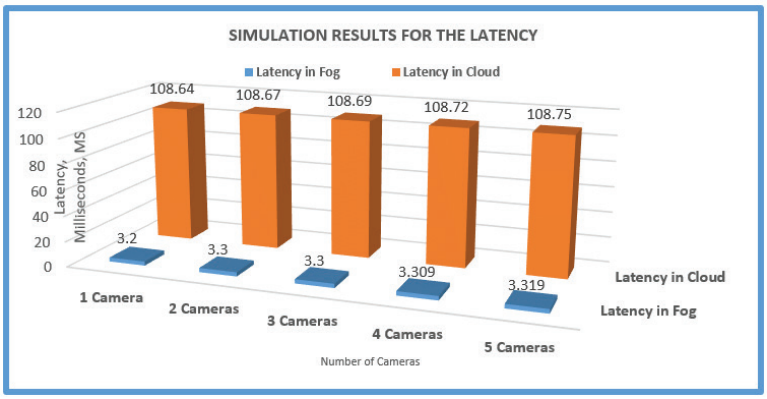


Figure 5. Graphical Representation of the simulation results on the latency

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

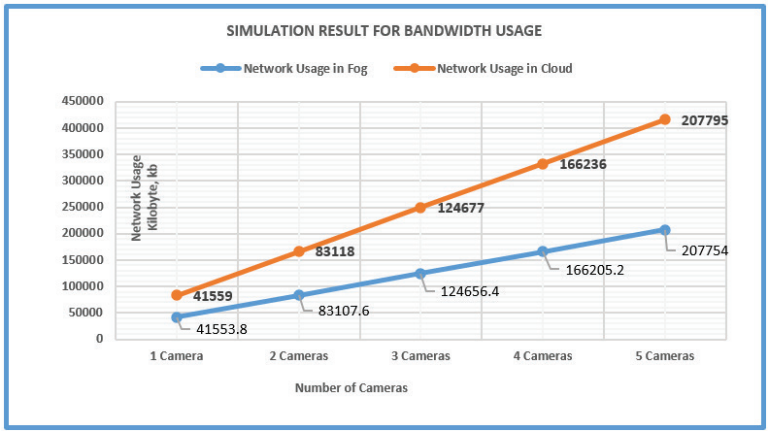


Figure 6. Graphical Representation of the Simulation Results on Network Bandwidth Usage

6. STATISTICAL ANALYSIS

A. Latency

a) Mean Calculation: The mean is calculated as:

Mean = $$Mean = \frac{\sum X}{n} \quad (1)$$

- **Fog-Based System Latency: X = [3.2, 3.3, 3.3, 3.309, 3.319] milliseconds and n = 5.**

Fog Latency Mean = 3.2856ms, which means that, on average, the fog-based system had a latency/delay of 3.29 milliseconds.

- **Cloud-Based System Latency: X = [108.64, 108.67, 108.69, 108.72, 108.75] milliseconds and n = 5.**

Cloud Latency Mean: = 108.694ms; this implies the cloud-based system had a much higher latency/delay of 108.69 milliseconds.

b) Standard Deviation Calculation: The standard deviation is calculated as:

$$\frac{\sqrt{\sum(X-Mean)^2}}{n-1} \quad (2)$$

Therefore, using Equation 2 and the simulation results in Table 2, the calculated standard deviations for network usage are:

- **Fog-Based System: 0.0485ms**
- **Cloud-Based System: 0.0428ms.**

Interpretation: Standard Deviation for latency in the Fog environment **= 0.0485ms.** This indicates slightly more variation, even though it is still very stable, with only minor fluctuations. Cloud Latency Standard Deviation: **= 0.0428ms.** This indicates that the latency in the cloud-based system is slightly more consistent (lower standard deviation) but at a much higher mean value.

B. Network Usage

a) Mean Calculations:

- **Fog-Based System Network Usage:** X = [41553.8, 83107.6, 124656.4, 166205.2, 207754.0] KB and n = 5

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

- **Fog Network Bandwidth Mean = 124655.4KB**; this is the average amount of data transmitted in the fog-based system.

- **Cloud-Based System Network Usage:** X = [41559.0, 83118.0, 124677.0, 166236.0, 207795.0] KB and n = 5

Cloud Network Bandwidth Mean = 124677.0 KB; this is the average amount of data transmitted in the cloud-based system.

b) Standard Deviation Calculation:

Therefore, using Equation 2 and the simulation results in Table 2, the calculated standard deviations for network usage are:

- Fog-Based System: 65,696.00 KB
- Cloud-Based System: 65,710.55 KB

Interpretation: Neither system maintains a stable network usage. The demand increases with time or workload. However, the difference in variation between fog and cloud is minimal, which suggests that both scale similarly in network load.

C. Independent Samples T-Test

The T-test is used to compare the means of two independent groups to see if there is a statistically significant difference between them.

The formula for the t-statistic in an independent samples t-test is:

$$t = \frac{Mean_1 - Mean_2}{\sqrt{\frac{SD^2}{n_1} + \frac{SD_2^2}{n_2}}} \qquad (3)$$

Where:

- $Mean_1$ and $Mean_2$ are the means of the two groups.
- SD_1 and SD_2 are the standard deviations of the two groups.
- n_1 and n_2 are the sample sizes of the two groups.

$$\textbf{Latency T-Test=} \quad \frac{3.2856 - 108.694}{\sqrt{\frac{0.0485^2}{5} + \frac{0.0428^2}{5}}}$$

T-Statistic: t=−3645.06t. This value indicates how far the difference between the two means is from zero, measured in standard error units. As observed here, a very large (absolute) t-value typically indicates a significant difference between the groups.

$$\textbf{Network Bandwidth Usage T-Test =} \quad \frac{124655.4 - 124677.0}{\sqrt{\frac{65696.0^2}{5} + \frac{65710.5^2}{5}}}$$

T-Statistic: t = −0.00052t. This small t-value suggests that the difference between the means of the two groups (fog-based and cloud-based systems) is minimal. This is, however, subject to increase as the number of IoT devices increases.

7. DISCUSSION

The results indicate that the fog-based system consistently maintains latency below 3.5ms across all tested camera configurations. The cloud-based system, however, exhibits latencies that exceed 108ms. Statistical analysis confirms this difference is highly significant (t=3645.06, p<0.0001). Notably, as the number of cameras increased from 1 to 5, the latency difference between cloud and fog architectures remained consistently noteworthy. This points out fog architecture's scalability advantage. Both architectures showed similar

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

network bandwidth usage in absolute terms. However, fog architecture demonstrated superior efficiency in terms of advantages in processing distribution. This is because the computational load was distributed across edge devices rather than concentrated in centralized cloud resources.

Our findings align with theoretical expectations regarding latency reduction in fog computing architectures, as recorded by [13]. Our approach demonstrates particular advantages in bandwidth utilization compared to cloud-based approaches presented by [3], which reported higher bandwidth consumption. However, speed slightly reduces as the camera count increases. This is a result of the increasing amount of real-time videos that are processed in smart environments.

Following the results the following differences between fog-based and cloud-based deployments presented in table 3 can be highlighted around these metrics to support future research works.

Table 3. Edge-based systems vs. Traditional cloud-based systems

Metric	Fog/Edge-Based Systems	Traditional Cloud-Based Systems
Latency	Reduced	Potentially Higher
Mobility	Explicit Mobility	Limited Mobility
Architecture	Decentralized	Centralized
Local Awareness	Yes	Limited
Geographic Distribution	Yes	Limited
Scalability	High	Scalable
Availability	High	High
Service Access	Edge/Handheld Devices	Limited to Internet Access
Remote Work	Enables Remote Work	Limited Remote Work
Real-time Processing	Better Support for Real-time Processing	May Require High Bandwidth
Analytics	Better Support for Real-time Analytics	Analytics May be Centralized
Classroom Management	Enhances Efficiency	May Require Additional Resources
Access to Resources	Easier and More Convenient	Dependent on Internet Connection

7.1. PRIVACY AND ETHICAL CONSIDERATIONS IN CAMERA-BASED SMART SEATING SYSTEMS.

The implementation of our proposed camera-based smart seating system in educational environments raises important privacy and ethical considerations that must be addressed proactively. Our proposed system relies on camera feeds to detect seat occupancy, which introduces several privacy challenges:

First, there is the question of student consent and awareness. Educational institutions implementing such systems should develop clear policies requiring informed consent from students, with transparent disclosure about what data is being collected, how it is processed, and where it is stored. It should also clarify that the system is designed for seat detection rather than individual identification.

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

Second, the technical implementation must incorporate privacy-by-design principles. In our system, this could be achieved through:

- Edge-based processing that extracts only occupancy data and discards raw video footage
- Intentional reduction of image resolution to prevent facial recognition
- Implementation of data minimization techniques that store only aggregated occupancy statistics rather than individual seating patterns
- Encryption of any data transmitted between fog nodes and cloud storage

Third, the system should comply with relevant data protection regulations, such as GDPR in Europe or FERPA in the United States, which may require data protection impact assessments before deployment.

Finally, educational institutions should establish ethical governance frameworks that prevent function creep, where a system designed for one purpose (seat management) evolves to serve other functions (such as student attendance monitoring or behavior tracking). Such frameworks should include regular audits, stakeholder consultations, and clear limitations on data usage.

These considerations should be integrated into the early design phases of smart classroom implementations. It should not be treated as an afterthought in order to ensure that technological innovations enhance student privacy and autonomy instead of compromising it.

8. CONCLUSION

Our findings suggest that fog-based smart seating systems could be introduced in university classrooms to significantly reduce classroom management overhead in terms of data processing speed and bandwidth utilization. This potentially translates into several additional instructional minutes per class session. Also, beyond the performance metrics of latency and bandwidth, the proposed system enhances classroom seating by enabling real-time seat detection and allocation. This reduces delays, ensures equitable student distribution, minimizes distractions, and promotes a more focused and engaging learning environment.

The work identified several scalability limitations. As the number of cameras increased from 1 to 5 per classroom, we observed a minor but consistent increase in latency (from 3.2ms to 3.319ms). This suggests potential processing bottlenecks at higher sensor densities. When scaling this system to cover an entire campus with hundreds of classrooms, the hierarchical fog architecture would require careful optimization in order to prevent overloading intermediate fog nodes. While fog computing improves real-time responsiveness, effective fog node deployment is essential to sustain performance.

Additionally, while our simulations demonstrated comparable network bandwidth usage between fog and cloud architectures comprising five (5) cameras, the difference may become more pronounced in larger deployments. The fog-based approach would likely maintain its latency advantage. It might, however, require additional edge nodes to distribute computational load effectively. These limitations underscore the importance of adaptive resource allocation frameworks, which can respond dynamically to varying classroom conditions and student populations.

For practical implementation in educational institutions, we recommend the following hardware specifications: (1) Entry-level IP cameras with 1080p resolution and basic motion detection capabilities would provide sufficient input data while minimizing costs; (2) Fog nodes can be effectively implemented using mid-range edge computing devices such as

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

Intel NUC mini PCs or equivalent ARM-based systems with at least 8GB RAM and quad-core processors; (3) Network infrastructure should support at least 100 Mbps within buildings, with redundant connections to ensure system reliability. This hardware configuration would support deployment costs of approximately $1,500-2,000 per classroom, with potential ROI realized through improved space utilization and reduced administrative overhead. These specifications represent a balance between performance requirements identified in our simulations and practical budget constraints faced by educational institutions.

Future research should focus on three key areas: (1) optimizing fog node placement algorithms to balance computational load across distributed educational environments, (2) developing adaptive resource allocation frameworks that respond to dynamic classroom conditions, and (3) establishing standardized benchmarks for smart classroom performance evaluation, (4) integrating artificial intelligence (AI) and machine learning (ML) algorithms for predictive analytics and personalized adaptive learning experiences to support students' individual needs. Educational institutions implementing these systems should begin with small-scale pilots in high-density classrooms, establish clear metrics for success (e.g., 95% seat allocation efficiency, 5ms response times), and develop comprehensive privacy policies before deployment.

ACKNOWLEDGMENTS

This research work is a section of Evans Obu's PhD work under the supervision and support of the co-authors, which is under the sponsorship of the Ghana Education Trust Fund (GetFund) and the University of Mines and Technology, Ghana - Staff Development Fund (UMaT-SDF). Many thanks go to the author's research assistant, Richemond Kofi Gbekley of UMaT-SRID, for his assistance with some sections of the simulation processes. I am also grateful to Mrs. Mary Adonma Obu for proofreading this article. In July 2024, an earlier version of this work was presented at the E-Learning Conference held at the KNUST Great Hall. In January 2025, it was also presented at a seminar at UMaT-SRID, Ghana. Much appreciation also goes to Ing. Dr. Albert Kwansah Ansah, Ing. Dr. Ezekiel Mensah Martey, and the Staff of UMaT-SRID for their valuable comments and suggestions during the seminars.

REFERENCES

[1] T. Aggarwal, Y. Lohumi, D. Gangodkar, and P. Srivastava, "Comprehensive Review of Recent Trends, Challenges, Applications, and Case Studies in Fog Computing," in Integration of Cloud Computing and IoT, Boca Raton: Chapman and Hall/CRC, 2024, pp. 480–496. doi: 10.1201/9781032656694-27.

[2] G. Lu, Q. Liu, K. Xie, C. Zhang, X. He, and Y. Shi, "Does the Seat Matter? The Influence of Seating Factors and Motivational Factors on Situational Engagement and Satisfaction in the Smart Classroom," Sustainability, vol. 15, no. 23, p. 16393, Nov. 2023, doi: 10.3390/su152316393.

[3] M. Fahimullah, G. Philippe, S. Ahvar, and M. Trocan, "Simulation Tools for Fog Computing: A Comparative Analysis," Sensors, vol. 23, no. 7, p. 3492, Mar. 2023, doi: 10.3390/s23073492.

[4] K. S. Awaisi, A. Abbas, S. U. Khan, R. Mahmud, and R. Buyya, "Simulating Fog Computing Applications Using iFogSim Toolkit," in Mobile Edge Computing, Cham: Springer International Publishing, 2021, pp. 565–590. doi: 10.1007/978-3-030-69893-5_22.

[5] S. Shrestha and S. Shakya, "A Comparative Performance Analysis of Fog-Based Smart Surveillance System," Journal of Trends in Computer Science and Smart Technology, vol. 2, no. 2, pp. 78–88, May 2020, doi: 10.36548/jtcsst.2020.2.002.

[6] N. Gao, M. S. Rahaman, W. Shao, K. Ji, and F. D. Salim, "Individual and Group-wise Classroom Seating Experience," Proc ACM Interact Mob Wearable Ubiquitous Technol, vol. 6, no. 3, pp. 1–23, Sep. 2022, doi: 10.1145/3550335.

[7] J. Nehyba, L. Juhaňák, and J. Cigán, "Effects of Seating Arrangement on Students' Interaction in Group Reflective Practice," The Journal of Experimental Education, vol.

http://dx.doi.org/10.21622/ACE.2025.05.1.1335

91, no. 2, pp. 249–277, Apr. 2023, doi: 10.1080/00220973.2021.1954865.

[8] G. Lu, Q. Liu, K. Xie, C. Zhang, X. He, and Y. Shi, "Does the Seat Matter? The Influence of Seating Factors and Motivational Factors on Situational Engagement and Satisfaction in the Smart Classroom," Sustainability, vol. 15, no. 23, p. 16393, Nov. 2023, doi: 10.3390/su152316393.

[9] G. Olges and K. Cohen, "Reducing Student Distraction Through Fuzzy Logic Based Seating Arrangements," 2025. [Online]. Available: https://arxiv.org/abs/2505.00545

[10] E. Gilman et al., "Internet of Things for Smart Spaces: A University Campus Case Study," Sensors, vol. 20, no. 13, p. 3716, Jul. 2020, doi: 10.3390/s20133716.

[11] M. A. Hassanain, M. O. Sanni-Anibire, and A. S. Mahmoud, "An assessment of users' satisfaction with a smart building on university campus through post-occupancy evaluation," Journal of Engineering, Design and Technology, vol. 22, no. 4, pp. 1119–1135, Jun. 2024, doi: 10.1108/JEDT-12-2021-0714.

[12] X. Wang, "Construction and application of a high-quality smart classroom education simulation platform in a cloud system environment," Service Oriented Computing and Applications, Aug. 2024, doi: 10.1007/s11761-024-00424-9.

[13] X. Wang, "Construction and application of a high-quality smart classroom education simulation platform in a cloud system environment," Service Oriented Computing and Applications, Aug. 2024, doi: 10.1007/s11761-024-00424-9.

[14] S. Mammadov and E. Kucukkulahli, "A User-Centric Smart Library System: IoT-Driven Environmental Monitoring and ML-Based Optimization with Future Fog–Cloud Architecture," Applied Sciences, vol. 15, no. 7, p. 3792, Mar. 2025, doi: 10.3390/app15073792.

[15] Y. Wu, H.-N. Dai, H. Wang, Z. Xiong, and S. Guo, "A Survey of Intelligent Network Slicing Management for Industrial IoT: Integrated Approaches for Smart Transportation, Smart Energy, and Smart Factory," IEEE Communications Surveys & Tutorials, vol. 24, no. 2, pp. 1175–1211, 2022, doi: 10.1109/COMST.2022.3158270.

[16] E. Çela, M. M. Fonkam, P. Eappen, and N. R. Vajjhala, "Current Trends in Smart Classrooms and Sustainable Internet of Things," 2024, pp. 1–26. doi: 10.4018/979-8-3693-5498-8.ch001.

[17] J. Xu, J. Li, and J. Yang, "Self-regulated learning strategies, self-efficacy, and learning engagement of EFL students in smart classrooms: A structural equation modeling analysis," System, vol. 125, p. 103451, Oct. 2024, doi: 10.1016/j.system.2024.103451.

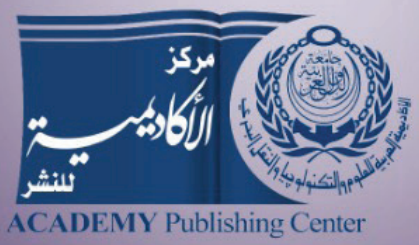
[18] L.-S. Huang, J.-Y. Su, and T.-L. Pao, "A Context Aware Smart Classroom Architecture for Smart Campuses," Applied Sciences, vol. 9, no. 9, p. 1837, May 2019, doi: 10.3390/app9091837.

[19] M. Yağanoğlu et al., "Design and validation of IoT based smart classroom," Multimed Tools Appl, vol. 83, no. 22, pp. 62019–62043, Jul. 2023, doi: 10.1007/s11042-023-15872-2.

[20] S. M. Dickson, "OVERVIEW OF INTERNET OF THINGS (IoT) NETWORK ARCHITECTURE FOR DIGITAL LEARNING AND DISTANCE EDUCATION," Pakistan Journal of Educational Research , vol. 7, no. 3, pp. 13–22, 2024, doi: 10.52337/pjer.v7i3.1144.

[21] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv:2207.02696 [cs], Jun. 2022, [Online]. Available: https://arxiv.org/abs/2207.02696

VOLUME 5, ISSUE 1, JUNE 2025



ACE

ADVANCES in COMPUTING & ENGINEERING

JOURNAL

E-ISSN: 2735-5985