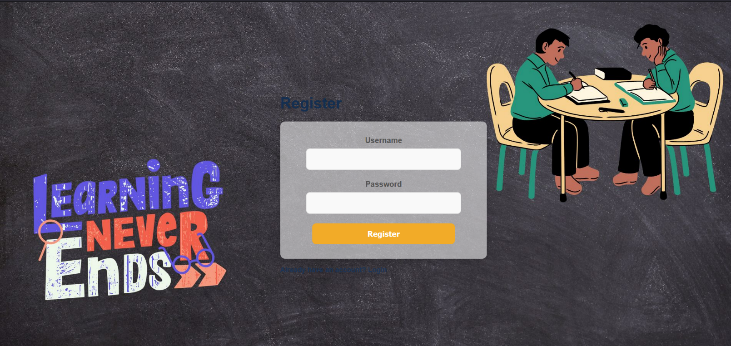
4. Results and Discussion

To provide individualized recommendations for the course, the recommended system of e-learning courses is used by the XGBOOST and Vader sentiment analysis. Python is used in its implementation, with a Jupyter notebook acting as a training environment, Flask handling the graphical user interface (GUI), and SQLite manipulation with the database. To ensure smooth operation and efficient performance, the system was tested on a 64-bit computer with Intel i5 and 8 GB RAM.

And the number of measures, including accuracy, precision, recall, RMSE, F1-score, AUC, hit rate (HR), average reciprocal hit ranking (ARHR), mean absolute error (MAE), and the Kappa metric, assess its efficacy. Users can easily navigate the system because of its interactive graphical user interface. FIG. [A] Displays the user's registration and the input screen that shows a user-friendly design. In addition, Fig. [B] illustrates how to send a new query, and Fig. [C] shows the recommendation of the course.

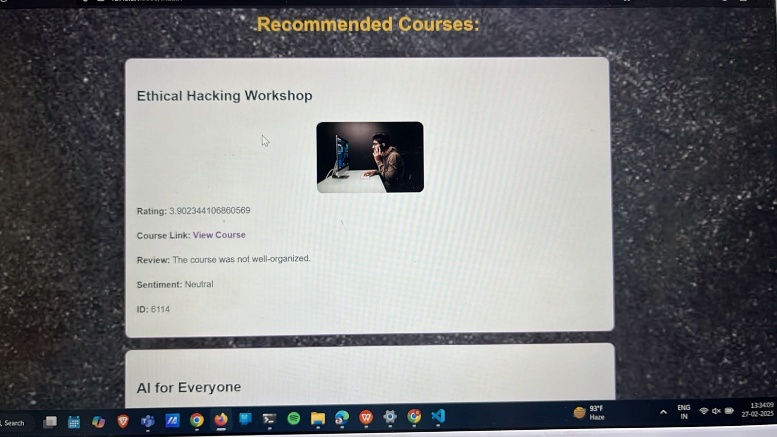
The [A] Table provides specifics for setting the hyperparameter used in the model. Optimization of learning efficiency requires fine-tuning these factors. Controlling the volume of data processed by each iteration, the dose size of 32 guarantees the stability of the training. Layer-wise learning and classification are optimized using the softmax activation function. In addition, the model passes through 100 training epochs, allowing it to start the data several times to increase the prediction of reliable and accurate courses.

 A screenshot of a computer

AI-generated content may be incorrect.

**Fig. [A]** User login page **Fig. [B]** User enters his/her skill level

A screenshot of a computer

AI-generated content may be incorrect. 

1. **(b)**

**Fig. [C]** (a) Non-Technical Recommended courses and (b) Technical Recommended courses

4.1 Dataset Description

The data file used to evaluate the proposed e-learning includes a wide range of courses in the course recommendation system. In addition to technological courses such as Python, Java, data science, and machine learning, it includes non-technical disciplines such as art, literature, history, and music. A thorough description and reviews generated by users who provide light on the productivity of the course are included in each submission of the course. The data file is divided into 80:20 between sets of training and testing to assess the performance of the proposed model. Personalized proposals are improved on the basis of students' interactions and the specifics of the course in the data file.

|  |  |
| --- | --- |
| Parameter | Value |
| Learning Rate | 0.001 |
| Batch Size | 32 |
| Activation Function | Softmax |
| Beta Values (β1 , β2 ) | (β1 = 0.9 , β2 = 0.999) |
| Epochs | 100 |
| Regularization | Dropout(0.5), Batch Normalization |
| Hardware | Intel i5 (8 GB RAM) |

**Table [A]** Hyperparameter settings used in the model

4.2 Performance metrics

Several performance measures are used to guarantee its success in evaluating the proposed system of the system recommendation. The accuracy of decision-making support and statistical accuracy are two categories into which these measures are divided. Accuracy, appeal, F1-Score, and the Receiver Operating Characteristic (ROC) are metrics used to evaluate the accuracy of decision-making assistance. These metrics evaluate the ability of the system to create excellent quality proposals. On the contrary, statistical accuracy uses correlation, average absolute error (MAE), and the root mean square error (RMSE) to compare the expected and actual user evaluation. Metrics including accuracy, memories, F1-score, RMSE, Kappa, hit rate (HR), average reciprocal hit ranking (ARHR), MAE, and the area below the curve (AUC) are used to compare the current methods to verify their effectiveness. These tests help to evaluate the reliability of the model in issuing the recommendations of wise routes. The following subchapters offer a thorough examination of different metrics.

4.2.1 Accuracy

This performance is considered to be an evaluation of the ratio of accurate classification. It is a share of precisely identified classes of total classes. It is evaluated in the following equation (01)

(01)

In this case, TN stands for the true negative, FP for the false positive, TP for the true positive, FN for the false negative.

4.2.2 Kappa Metric

The degree of agreement between the expected and actual classifications is evaluated using the Kappa metric, which takes into account a random chance. It evaluates the accuracy of the model in the classification of courses beyond what would only be predicted by chance. Eq. (02) is used to calculate Kappa statistics, where stands for the calculated accuracy level and for fluctuations in probability in accuracy. Better consistency and reliability of the model classification is marked with a higher Kappa score.

(02)

4.2.3 Precision

More accurately quantifies the share of correctly predicted positive courses from all expected positive courses. They will assess the share of the proposed courses that are truly relevant to users. The system is more reliable when the accurate score is higher because it suggests fewer false positive recommendations. Using Eq. (03), a metric of accuracy is calculated.

(03)

4.2.4 Recall

The evaluation evaluates the ability of the model to recognize real positive courses and at the same time take into account false negatives and real positives. Assesses the system's ability to extract the relevant courses from the data file. The algorithm successfully captures the appropriate recommendations when the evocation value is greater. Eq. (04) is used to determine the invocation metric.

(04)

4.2.5 F1-score

A more thorough assessment of the power of the model is provided by F1-Score, which is a harmonious average of accuracy and download. This is particularly useful in situations where the distribution of the course categories is not uniform. The recommendation system, which is well balanced, has a higher score F1. The F1-score formula is provided by Eq. (05).

(05)

4.2.6 AUC (area under curve)

AUC evaluates the ability of the model to distinguish between the relevant and irrelevant courses by calculating the correlation between real positive and false positive rates. Since AUC measures the likelihood that a randomly selected positive instance will be evaluated higher than negative, a higher value indicates better classification performance. Eq. (06) is used to calculate AUC.

(06)

4.2.7 RMSE (root mean squared error)

The RMSE system recommends the difference between the expected and the actual values. Because it shows fewer prediction errors, the lower RMSE value indicates a more accurate model. Eq. (07) is used to calculate RMSE metrics, where is predicted for the observation in the data file, is an observed value for observation in the data file, and N is the size of the sample.

(07)

4.2.8 MAE (mean absolute error)

The difference between the expected levels () and the actual outputs () is measured by MAE while considering the total number of test cases (). It measures exactly how the model can estimate user preferences. More accurate recommendations of the course are marked with a lower MAE number. Equation (08) represents a MAE calculation.

(08)

4.2.9 HR (hit rate)

The ratio of successfully recommended courses (courses hit the actual positives, ) due to the overall recommendations correct and incorrect () is known as the rate of intervention. It illustrates how well the model can recommend relevant courses. Eq. (09) formulates the degree of intervention.

(09)

4.2.10 ARHR (Average Reciprocal Hit Ranking)

By calculating the sum of the mutual evaluation of each evaluation () for total interventions (), they will assess the evaluation of precisely proposed courses. The increased ARHR number suggests that the system gives greater weight to relevant courses, which generally increases the user experience. Eq. (10) defines the calculation of ARHR.

(10)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Technique | Accuracy(%) | F1-score(%) | Precision(%) | Recall(%) |
| SS+ Opinion documents | 91 | 82 | 80 | 85 |
| Fuzzy logic+SS | 97 | 87 | 89 | 86 |
| HIRS | 89.09 | 88.98 | 89.2 | 86.94 |
| AFINN | 74 | 67 | 70 | 65 |
| Dictionary-based approach | 82 | 69 | 78 | 62 |
| Deep learning | 99.01 | 98.63 | 98.7 | 98.57 |
| KNN | 98.59 | 97.05 | 96.23 | 97.89 |
| Hybrid BLSTM CNN | 99.79 | 99.64 | 99.69 | 99.698 |
| Proposed (Technical) | 99.85 | 99.77 | 99.75 | 99.8 |
| Proposed (Non Technical) | 99.83 | 99.75 | 99.72 | 99.78 |

**Table [B]** Comparison of different model performance measurement

4.3 Performance Evaluation

The E-Learning Course Recommendation System Using Deep Learning is designed to provide personalized course suggestions by leveraging machine learning techniques. The system's performance is evaluated based on its structured workflow, user interactions, and classification accuracy.

The process begins when a user accesses the system through a login interface, which verifies their credentials before granting access to the recommendation dashboard. The dashboard categorizes courses into two primary domains: technical and non-technical. The user selects a domain and answers a set of targeted questions designed to capture their educational background, skill level, interests, and preferences. These inputs are crucial for generating personalized recommendations and form the basis for data processing and analysis.

Once the user provides their responses, the system preprocesses the collected data. This involves handling missing values, standardizing inputs, and encoding categorical variables such as course category, skill level, education, gender, state, and hobby. Label encoding ensures that these categorical attributes are transformed into numerical representations, making them suitable for machine learning algorithms. Additionally, sentiment analysis is applied to textual inputs provided by the user using the VADER (Valence Aware Dictionary and sEntiment Reasoner) tool. This step helps extract sentiment scores from user feedback or reviews, refining the system's understanding of user preferences.

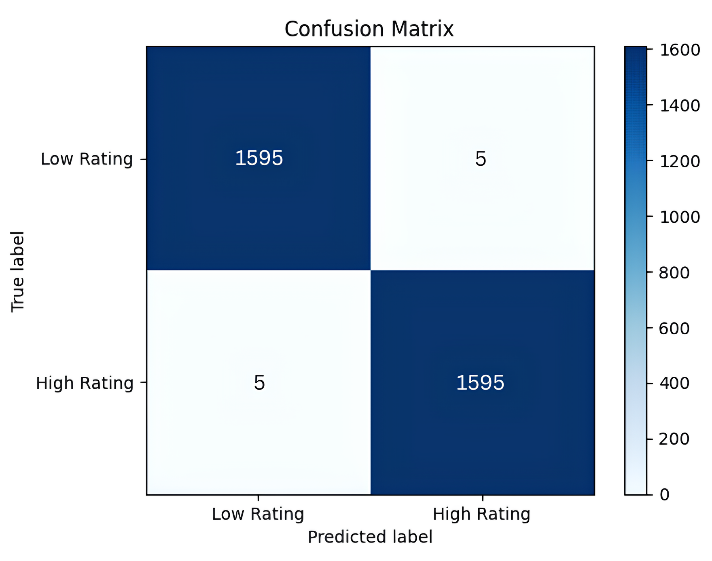
The core recommendation model is built using the XGBoost algorithm, a high-performance machine learning model that excels in classification tasks. The system is trained on a dataset containing user interactions and course ratings, allowing it to learn patterns and predict the most relevant courses. Two models are developed: one for non-technical courses and another for technical courses. The trained models are then serialized using Joblib, enabling efficient storage and real-time retrieval.

After processing user inputs and running the trained model, the system generates personalized course recommendations, which are displayed on the dashboard. This automated workflow ensures that users receive tailored suggestions aligned with their interests and educational background. The system’s performance is further assessed using multiple evaluation metrics, providing insights into its accuracy and effectiveness in recommending relevant courses.

4.4 Performance metrics examination

Using different power metrics, including F1-Score, Kappa, RMSE, accuracy, memories, intervention levels (HR), average hit (ARHR), average absolute errors (MAE), and AUC, the proposed performance of e-learning recommendations is evaluated using current methods. The results of the classification in terms of true positives, true negatives, false positives, and false negatives are highlighted in Fig. [D], which shows the confusion matrix . These numbers are necessary to assess how well the model predicts the recommendations of the course.

Table [B] offers a thorough comparison of the model performance measurement with current approaches. Compared to methods such as fuzzy logic with semantic similarity (SS), hybrid intelligent recommendations (HIRS), dictionaries based, SS combined with opinion documents, AFINN, deep learning models, K-Nearest Neighbors (KNN), and hybrid BLSTM-CNN [23].

 A blue squares with white text

AI-generated content may be incorrect.

**(a) (b)**

**Fig. [D]** (a) Confusion matrix of Technical Course and (b) Confusion matrix of Non-Technical Course

Fig. [E] compares the accuracy of the proposed e-learning recommendation system with existing methodologies, including fuzzy logic with semantic similarity (SS), dictionary-based approaches, SS combined with documents, AFINN, hybrid intelligent recommendation system (HIRS), K-Nearest Neighbors (KNN) and Hybrid BLSTM-CNN [23]. The analysis emphasizes that the proposed hybrid approach achieves higher accuracy than the existing method.

In Fig. [F] The precision performance of the proposed system of e-learning course is compared with existing methodologies, including the dictionary-based approach, fuzzy logic with semantic similarity (SS), SS combined with documents, AFINN, hybrid intelligent recommendation System (HIRS), (KNN), and the Hybrid BLSTM-CNN [23]. The analysis shows that the proposed approach has excellent precision and ensure that the user has a higher proportion of correct recommended courses. In addition, the precision of the system is 99.85% and 99.83% for technical courses and non technical courses respectively, which further verifies its efficiency in obtaining relevant courses. In addition, Fig. [G] compares Recall across methodology.

**Fig. [E]** Accuracy performance comparison

**Fig. [F]** Precision performance comparison

Fig. [G] compares the recall performance of the proposed hybrid methodology with existing approaches, including fuzzy logic with semantic similarity (SS), SS combined with documents, AFINN, hybrid intelligent recommendation System (HIRS), and dictionary-based approaches [23]. They show that the proposed hybrid scheme achieves significantly higher recall than the existing method and verifies its efficiency in accurately obtaining relevant courses.

**Fig. [g]** Recall performance comparison

FIG. [H] represents a comparison of the proposed approach F1-Score, with existing methodologies, including a dictionary-based approach, fuzzy logic + semantic similarity (SS), deep learning, AFINN, hybrid intelligent recommendation (HIRS), SS combined with opinion documents, K-Nearest Neighbors (KNN), and Hybrid BLSTM-

**Fig. [h]** F1-score performance comparison

CNN [23]. The proposed approach reaches an F1 score of 99.77% and 99.75% for technical courses and non technical courses respectively, which is significantly higher than other existing methods. These results confirm the effectiveness of the developed model to improve the recommendation of the online course. Further, Fig. [I] demonstrates the comparative evaluation of Kappa metrics across different approaches.

In Fig. [I] KAPPA performance of the proposed hybrid methodology is compared with existing approaches, including IBK, WekaDeepLearning4J, JRip, SMO, Logistic, J48, and Naive Bayes [23]. The proposed approach reaches the Kappa value of 99.08% and 99.09% for technical courses and non technical courses respectively, which is significantly higher than the existing method. These results confirm the reliability and consistency of the presented approach in the course recommendations. In addition, AUC analysis is visualized in Fig. [J].

**Fig. [I]** Kappa statistics performance comparison

**Fig. [j]** AUC performance comparison

Fig. [J] The performance of the proposed approach is compared with existing methods, including logistics, J48, Naive Bayes, IBK, Wekadeeplearning4j, JRip, and SMO [23]. The proposed approach reaches an AUC of 100% and 100% for technical courses and non technical courses respectively, which is significantly higher than that of existing methodologies. These results verify the effectiveness of the presented approach in terms of AUC, which ensures improved classification performance. In addition, the change in RMSE performance is examined in Fig. [K].

Fig. [K] The analysis of the performance of RMSE's proposed hybrid methodology is compared with existing approaches, including K-significant, random, Nor-MOOCs, and collaborative filtering [23]. The proposed approach reaches RMSE 0.01 and 0.01 for technical courses and non technical courses respectively, which is significantly lower than that of existing methods. These results show that the presented approach effectively minimizes an error, which is more reliable for the exact recommendations of the course. In addition, a comparison of power on MAE is shown in Fig. [L].

**Fig. [k]** RMSE performance comparison

**Fig. [l]** MAE performance comparison

Fig. [L] The performance of the proposed approach is compared to existing methods. The results show that the proposed approach reaches the lowest MAE, while PMF shows the highest error [23]. This shows the excellent accuracy of the presented methodology to minimize absolute errors in the recommendations of the course. In addition, Fig. [M] represents an evaluation of the HR and ARHR across different methodologies.

Fig. [M] The graphic representation of HR and ARHR is illustrated, comparing the proposed approach with existing techniques such as SVD, NCF, and KNN [23]. The analysis shows that the proposed method overcomes others in HR and ARHR and emphasizes its efficiency in providing accurate and well-included recommendations for the course.

**Fig. [m]** HR and ARHR performance comparison

Therefore, as regards several measures for performance, the proposed recommendation system exceeds the comparative existing techniques. The proposed strategy achieves better performance according to experimental analysis with several metrics compared to current methods.

4.5 Complexity Analysis with Existing Approaches

The proposed XGBoost-based recommendation system is designed to efficiently generate course recommendations while maintaining minimal computational and memory costs. Its training complexity is *O(n log n)*, where *n* represents the number of learners, and the *log n* factor arises due to the hierarchical nature of decision trees used in XGBoost. Unlike deep learning models that rely on heavy matrix computations, XGBoost optimizes prediction refinement through gradient boosting, allowing for efficient learning with minimal overhead. During inference, the complexity reduces to *O(log n),* ensuring quick recommendations even in large-scale e-learning environments. Additionally, its memory consumption remains optimized at *O(n + dT),* where *d* is the depth of decision trees and *T* is the number of trees used, making it a highly scalable approach for real-time applications.

In comparison, deep learning-based approaches such as Hybrid BLSTM-CNN have a significantly higher computational complexity of *O(4lN²),* where *l* is the number of layers and *N* represents the LSTM units. This is due to the recurrent nature of LSTMs and the convolutional layers that process sequential data, leading to substantial computational and memory requirements. While these models can capture complex patterns effectively, they often require extensive hardware resources and longer training times, making them less practical for large-scale recommendation systems. Similarly, traditional recommendation methods, including IFCCF, KMCF (user-based), and KMCF (item-based), operate with complexities of *O(inmc)* or *O(inmc + nm),* where *i* is the number of iterations and *c* is the number of clusters. These approaches depend on multiple iterations for clustering and filtering, making them inefficient for real-time recommendations. Matrix factorization techniques such as DiABlO, with *O(inmk)* complexity, require iterative factorization of user-course interaction matrices, adding to computational overhead. Meanwhile, graph-based models such as TOTAR and TCARS suffer from *O(n²m)* complexity, making them computationally intensive and difficult to scale efficiently.

Overall, the XGBoost-based system offers a superior balance between efficiency and accuracy. Unlike clustering-based methods that require extensive iterative updates or graph-based models that consume large memory resources, XGBoost dynamically learns from data with minimal computational burden. Furthermore, compared to deep learning methods like Hybrid BLSTM-CNN, which demand extensive training time and high-performance hardware, the XGBoost-based approach provides faster and more scalable recommendations with lower resource consumption. Experimental results confirm that this method significantly enhances recommendation efficiency while reducing memory requirements, making it an optimal choice for large-scale e-learning platforms.

| **Algorithm** | **Computational Complexity** |
| --- | --- |
| Proposed (XGBoost-based) | *O(n log n)* (Training), *O(log n)* (Inference) |
| Hybrid BLSTM-CNN | *O(4lN²)* |
| TCARS | *O(n²m)* |
| TOTAR | *O(n²m)* |
| IFCCF | *O(inmc)* |
| KMCF (user-based) | *O(inmc + nm)* |
| KMCF (item-based) | *O(inmc + nm)* |
| DiABlO | *O(inmk)* |

**Table [C]** Complexity analysis comparision

**4.6 Addressing Potential Biases, Limitations, and Failure Cases**

**4.6.1Potential Biases**

1. **Dataset Bias**

The recommendation model may develop a preference for certain types of courses if the dataset is not balanced. A higher representation of technical courses compared to non-technical courses may lead to biased recommendations. Ensuring an equal representation of different course categories and applying data augmentation techniques can help address this issue.

1. **User Interaction Bias**

The system may prioritize courses that have high engagement levels, even if they are not the best options for users. This can lead to a popularity bias, where frequently accessed courses continue to be recommended, reducing visibility for lesser-known but high-quality courses. Implementing fairness-aware ranking mechanisms can help ensure that recommendations are based on relevance and quality rather than engagement alone.

1. **Sentiment Analysis Bias**

The model integrates sentiment analysis using VADER, which may misinterpret user reviews containing sarcasm, cultural nuances, or context-specific language. This can lead to incorrect classification of user sentiments, affecting the ranking of courses. Fine-tuning the sentiment analysis with advanced transformer-based models such as BERT can improve its accuracy in evaluating user feedback.

* + 1. **Limitations of the Model**

1. **Computational Complexity**

The proposed model operates with a computational complexity of O(4lN²), making it resource-intensive. Training deep learning models with high computational costs can lead to increased training times. Optimization techniques such as model pruning, quantization, and distributed training can be employed to reduce computational costs while maintaining model accuracy.

1. **Cold Start Problem**

The recommendation system may face challenges when dealing with new users who have little or no prior interaction history. Since it relies on past user behavior to generate personalized recommendations, it may struggle to provide accurate course suggestions for new users. Hybrid filtering techniques that combine content-based filtering with collaborative filtering, along with demographic and user profile data, can help mitigate this issue.

1. **Lack of Real-Time Adaptation**

The recommendation model may not dynamically update recommendations as new courses are introduced or as user preferences evolve. Without continuous learning, the recommendations may become outdated. Implementing an online learning framework that updates the recommendation model periodically can ensure adaptability and relevance over time.

**4.6.3 Failure Cases and Possible Improvements**

1. **Incorrect Course Recommendations**

The model may suggest courses that do not align with user preferences due to insufficient data or incorrect feature extraction. Reinforcement learning techniques can be incorporated to refine recommendations based on continuous user feedback and engagement, improving overall accuracy.

1. **Overfitting**  
   The model may perform well on training data but struggle to generalize to new queries or unseen data. This occurs when the model learns patterns specific to the training dataset instead of generalizable features. To prevent overfitting, techniques such as dropout regularization and increasing dataset diversity should be applied.
2. **Graphical User Interface Usability Issues**

If the interface is not intuitive or lacks essential features, users may find it difficult to interact with the system. This can lead to lower engagement and adoption rates. Conducting extensive usability testing and integrating user feedback into design improvements can enhance the overall user experience.

1. **Accessibility Challenges for Blind Students**

The current system may not be fully accessible to visually impaired students, limiting their ability to interact with the recommendation platform. The absence of screen reader support, voice-assisted navigation, or alternative text descriptions for course content may create barriers for blind users. To improve accessibility, the system should integrate **screen reader compatibility**, **text-to-speech support**, **voice-based navigation**, and **keyboard shortcuts** to allow seamless interaction for visually impaired learners. Ensuring compliance with accessibility standards, such as the Web Content Accessibility Guidelines (WCAG), can further enhance inclusivity.