NLP Hybrid Deep Learning Model For E-Learning System Prediction Classifier System

Dr. Sudha Vemaraju[1], Dr K. Sarvani[2], Dr. Satya Vani Bethapudi[3], Dr. Venkateswarlu Chandu[4,*], Dr.Ch. Sahyaja[5] ,Dr.K. Kiran Kumar Varma[6] and Ankam Dhilli Babu[7]

[1]Associate Professor, GITAM School of Business, GITAM University (Deemed To Be University)- Hyderabad, AP, India
[2]Assistant Professor, Amritha School of Business, Amrita Viswa Vidyapeetam, Amaravati, AP, India
[3]Assistant Professor, Department of Entrepreneurship, School of Business, GITAM University, Hyderabad,India
[4]Assistant Professor, KL Business School, Koneru Lakshmaiah Education Foundation Greenfields, Vaddeswaraam, AP, India
[5]Assistant Professor, Amritha School of Business, Amrita Viswa Vidyapeetam, Amaravati A.P,India
[6]Associate Professor, Dept of EM&H, SRKR Engineering College, Bhimavaram, A.P,India
[7]Assistant Professor, Vijaya Institute of Technology for Women, Department of Information Technology,A.P,India

Abstract

Natural Language Processing (NLP) with deep learning transforms how machines understand and generate human language by leveraging powerful neural networks, such as Recurrent Neural Networks (RNNs) and Transformers. Deep learning models in NLP can process vast amounts of unstructured text data, enabling highly accurate tasks like sentiment analysis, machine translation, text summarization, and question-answering. Course construction refers to designing, organizing, and implementing educational content to achieve specific learning objectives. A Learning Management System (LMS) integrated with deep learning in e-learning revolutionizes personalized education by analyzing student data to provide customized learning paths and experiences. Deep learning models in the LMS can assess a student's learning style, pace, and areas of difficulty by processing large volumes of user interactions, such as quizzes, assignments, and engagement metrics. The proposed n-gram-LMS-MC-DL model integrates n-gram language modeling, Markov Chain, and Deep Learning (DL) techniques to enhance the prediction accuracy in Learning Management Systems (LMS). This hybrid approach aims to predict students' next learning states based on their interactions within the LMS. The system achieves significant improvements in prediction accuracy across various learning stages. For instance, the n-gram-LMS-MC-DL

model outperforms both the standalone Markov Chain and Deep Learning models, reaching an average accuracy of 92.82%, compared to 85.52% for Deep Learning and 77.78% for the Markov Chain. In individual stages, the model predicts with 92.1% accuracy for transitioning from "Lesson Completed" to "Quiz Started" and 95.2% accuracy for progressing from "New Lesson" to "Discussion." In addition to enhanced accuracy, the system maintains high precision (average 0.87), recall (average 0.85), and F1-score (average 0.855) across various learning activities, with manageable time complexities ranging from 120 ms to 155 ms.

Keywords: Markov Chain, Learning Management System (LMS), E-learning, Deep Learning, Prediction, Classification

1. Introduction

E-learning management involves the strategic planning, implementation, and evaluation of online educational programs to enhance learning outcomes. This includes the use of Learning Management Systems (LMS) that facilitate course delivery, track student progress, and provide resources for both educators and learners [1]. Effective e-learning management incorporates user-friendly interfaces, interactive content, and robust assessment tools to engage students and promote active learning. Additionally, it emphasizes the importance of data analytics to monitor performance and identify areas for improvement, ensuring that educational goals are met efficiently [2]. A Learning Management System (LMS) is a powerful software application designed to facilitate the administration, delivery, and tracking of educational courses and training programs. It allows educators to create and manage courses efficiently, offering a variety of content formats such as videos, quizzes, and interactive simulations to cater to different learning styles [3-5]. With user management capabilities, administrators can track enrollment and monitor student progress, while built-in assessment tools streamline evaluation through automated grading and analytics. Communication features, including discussion forums and chat functions, foster collaboration between learners and instructors, enhancing engagement and peer interaction [6-8]. LMS platforms are typically web-based, providing flexible access to course materials anytime and anywhere, making education more accommodating. Additionally, many LMS solutions integrate seamlessly with other educational tools and are scalable to suit both small institutions and large organizations.

A prediction model for e-learning leverages advanced analytics and machine learning techniques to forecast student performance, engagement, and course outcomes [9-10]. By

analyzing historical data, including participation rates, assessment scores, and interaction metrics, the model identifies patterns and trends that can inform interventions and personalized learning strategies [11]. For instance, it can predict which students may struggle with specific concepts, allowing educators to provide targeted support before issues escalate. Additionally, the model can assess the effectiveness of different teaching methods and course materials, guiding content improvements to enhance learning experiences [12-13]. By employing techniques such as regression analysis, decision trees, or neural networks, the prediction model helps educational institutions make data-driven decisions, ultimately leading to improved student retention, satisfaction, and overall success in e-learning environments [14].

Moreover, the prediction model can enhance the design of adaptive learning pathways, tailoring content delivery based on individual learner needs and preferences [15]. This customization fosters greater engagement and motivation, as students receive materials and assessments aligned with their learning pace and style. The model can also integrate demographic factors, such as age, educational background, and prior knowledge, to refine predictions and identify at-risk populations [16]. Furthermore, by continuously updating its algorithms with new data, the model evolves over time, increasing its accuracy and relevance. This dynamic approach not only supports educators in timely interventions but also empowers students by promoting self-directed learning [17]. Ultimately, a well-implemented prediction model in e-learning contributes to a more responsive educational environment, maximizing learning outcomes and fostering a culture of continuous improvement. Deep learning plays a transformative role in Learning Management Systems (LMS) by enhancing personalized learning experiences and improving educational outcomes [18]. By utilizing neural networks, deep learning algorithms can analyze vast amounts of data generated by learners' interactions with course materials, assessments, and collaborative activities [19]. This analysis helps in identifying patterns and predicting individual learning behaviors, allowing the LMS to adapt content and learning paths to suit each student's unique needs. For example, deep learning can power recommendation systems that suggest relevant resources or courses based on a learner's previous performance and interests [20]. Additionally, it can facilitate automated grading and feedback through natural language processing, enabling quicker assessments of written assignments and discussions. As a result, deep learning not only enriches the interactivity and adaptability of LMS but also empowers educators with insights to refine their teaching strategies and enhance student engagement and retention [21].

This paper presents significant contributions to the field of e-learning through the development of the n-gram-LMS-MC-DL model, which integrates n-gram language modeling, Markov Chain processes, and Deep Learning techniques for enhanced prediction accuracy in Learning Management Systems (LMS). The proposed model achieved an impressive average prediction accuracy of 92.82%, surpassing the accuracy of traditional models, which stood at 77.78% for the Markov Chain and 85.52% for Deep Learning alone. Additionally, the model demonstrated strong precision, recall, and F1-score metrics, with averages of 0.87, 0.85, and 0.855, respectively. This model's effectiveness is further evidenced by its capacity to accurately predict students' next actions, with specific state transitions showing accuracy rates as high as 95.2% for moving from "New Lesson" to "Discussion." Furthermore, the integration of these methodologies allows for timely and accurate feedback in the e-learning process, thus improving the overall student experience and engagement. Overall, this research advances the understanding of how combining various predictive techniques can lead to more effective and responsive educational environments.

.

2. Proposed n-gram Learning Management System (n-gram-LMS-MC-DL) with deep learning with the Markov Chain

The proposed n-gram Learning Management System with Markov Chain and Deep Learning (n-gram-LMS-MC-DL) is designed to enhance the adaptability and personalization of e-learning by leveraging both linguistic patterns and probabilistic modeling. The n-gram model captures sequences of words (n-grams) from student interactions, such as text inputs, to analyze language patterns and predict the next word or phrase based on prior occurrences. This approach enables the system to better understand student queries, responses, or content engagement, improving the system's language comprehension. The integration of the Markov Chain further enhances this model by introducing a probabilistic framework that predicts future states (learning paths or actions) based on current ones. In the LMS context, this means that the system can predict a learner’s next action or topic based on their current activity, improving the flow and adaptability of content delivery. The student has consistently performed well in previous topics, the system may predict that they are ready for more advanced content, adjusting the learning path dynamically. Deep learning complements these techniques by providing the computational power to analyze large amounts of data,

identifying complex patterns, and improving the accuracy of predictions. The deep learning component helps in understanding student behavior on a more granular level, enabling the LMS to offer highly personalized learning experiences. The proposed method for enhancing a Learning Management System (LMS) utilizes min-max normalization combined with a deep learning-based voting classifier to improve the accuracy and efficiency of student performance predictions. The first step involves applying min-max normalization to the input data, which scales the features to a uniform range, typically [0, 1]. This is defined by the equation (1)

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

In equation (1) X is the original feature value, X_{min} and X_{max} are the minimum and maximum values of the feature, respectively, and X′ is the normalized value. Once the data is normalized, a deep learning model, such as a neural network, is trained on the dataset to capture complex patterns in student interactions and performance metrics. This model can consist of multiple layers, including input, hidden, and output layers, where each layer's neurons are activated using functions like ReLU or sigmoid. The output of the deep learning model can be combined using a voting classifier, which aggregates predictions from multiple models to enhance overall predictive performance. The final prediction is determined by majority voting or weighted voting based on individual model accuracies. Mathematically, the voting mechanism can be expressed as in equation (2)

$$\hat{Y} = argmax(\sum_{i=1}^{n} w_i . P_i) \quad (2)$$

In equation (2) $\hat{Y}$ is the final predicted class, w_i represents the weight assigned to the iii-th model based on its performance, and w_i is the predicted probability from the i-th model.A Learning Management System (LMS) for e-learning serves as a centralized platform to facilitate, manage, and optimize the educational experience. To effectively evaluate student performance and engagement, an LMS can employ predictive modeling techniques based on historical data. For instance, let's define a dataset DDD consisting of n students, where each student i has features F representing their engagement metrics, assessment scores, and interaction frequencies. The performance can be modeled using a linear regression approach, given by the equation (3)

$$Y_i = \beta_0 + \beta_1 F_{1i} + \beta_2 F_{2i} + \cdots .. + \beta_k F_{ki} + \epsilon_i \quad (3)$$

In equation (3) is the predicted performance score for student i, β_0 is the intercept, β_k are the coefficients for the features F_{ki} , and ϵ_i represents the error term. To enhance the learning experience, the LMS can implement a feedback loop based on performance predictions. For instance, if a student's predicted performance ϵ_i falls below a certain threshold T, targeted interventions can be initiated. The LMS can also apply min-max normalization to the engagement features, scaling them to a range of [0, 1] using the equation (4)

$$F'_{ki} = \frac{F_{ki} - F_{kmin}}{F_{kmax} - F_{kmin}} \tag{4}$$

With normalization ensures that all features contribute equally to the model. Additionally, the LMS can utilize advanced algorithms, such as neural networks, to capture non-linear relationships in the data. The output layer of the neural network can predict the likelihood of student success using a softmax activation function stated in equation (5)

$$P(Y_i = y) = \frac{e^z y}{\sum_j e^{zj}} \tag{5}$$

In equation (5) $e^z y$ is the score for class y. By integrating these methodologies, an LMS can create a data-driven, adaptive learning environment that effectively meets the needs of individual learners and enhances overall educational outcomes. To further elaborate on the LMS for e-learning, we can enhance the predictive model using a deep learning framework. Suppose we utilize a feedforward neural network with one hidden layer containing h neurons. The output for student i can be defined as in equation (6)

$$Y_i = f\left(W^{(2)}.h + b^{(2)}\right) \tag{6}$$

In equation (6) $W^{(2)}$ is the weight matrix for the connections from the hidden layer to the output layer, h represents the hidden layer activations, and $b^{(2)}$ is the bias term. The activation of the hidden layer can be expressed as in equation (7)

$$h = f\left(W^{(1)}.F'_1 + b^{(1)}\right) \tag{7}$$

In equation (7) $W^{(1)}$ is the weight matrix connecting the input features F'_1 (normalized features) to the hidden layer, and $b^{(1)}$ is the corresponding bias. The choice of activation function f could be ReLU (Rectified Linear Unit) for hidden layers, defined as in equation (8)

$$f(x) = max(0, x) \tag{8}$$

After training the model using a dataset with known outcomes, we can optimize the weights $W^{(1)}$ and $W^{(2)}$using backpropagation and a loss function, such as mean squared error (MSE) stated in equation (9)

$$L = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{9}$$

In equation (9) $\hat{Y}_i$ is the predicted output from the neural network. The weights are adjusted based on the gradient of the loss function with respect to the weights, typically using an optimization algorithm like Adam or SGD (Stochastic Gradient Descent). Once the neural network is trained, the LMS can leverage the trained model to make predictions for new students. By integrating these predictions with a feedback mechanism, the LMS can dynamically recommend resources or interventions based on individual student needs. For example, if the predicted performance score falls below a defined threshold T, the system can automatically notify instructors and suggest tailored content or additional support resources.

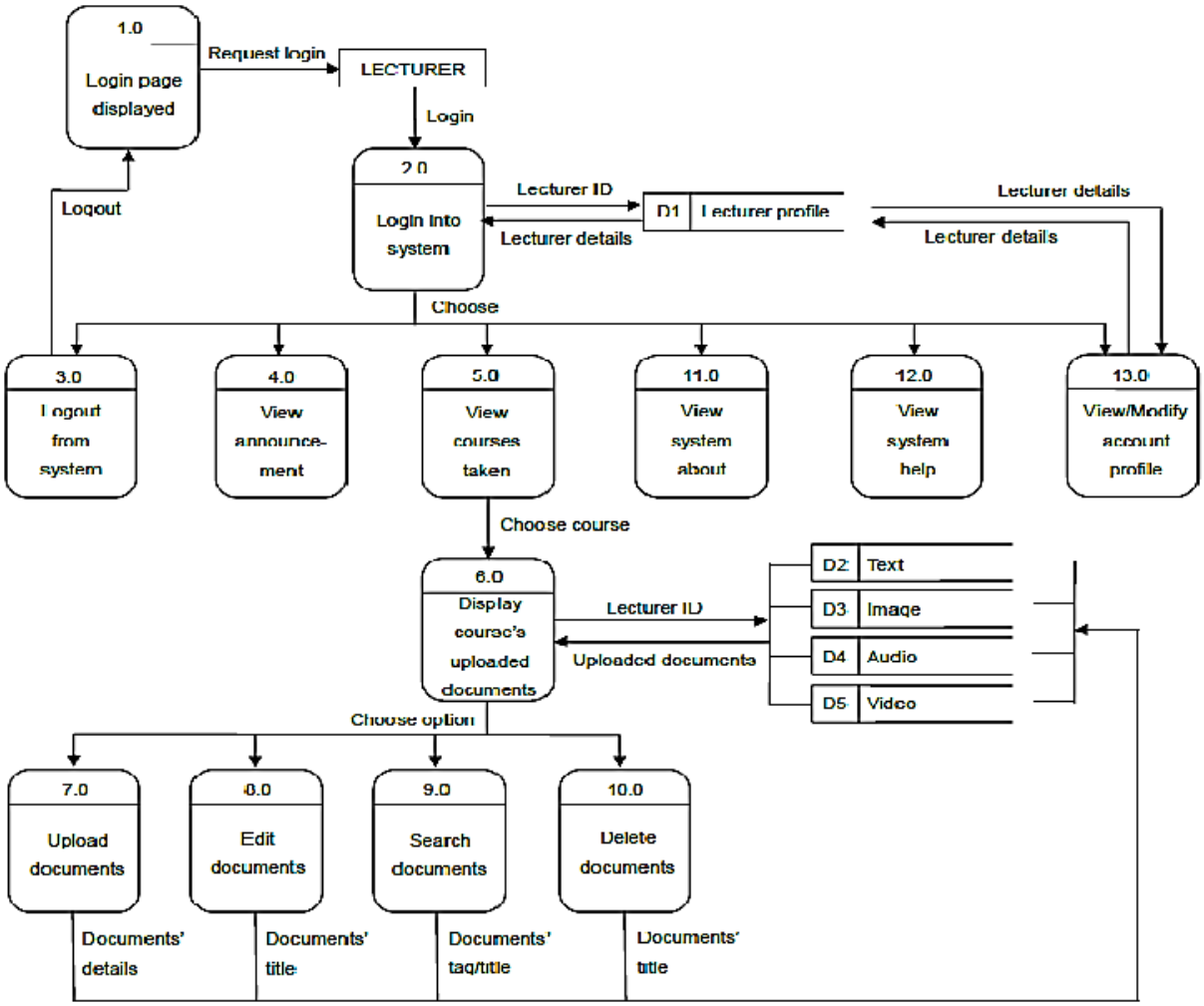

Figure 1: Learning Management System with Deep Learning

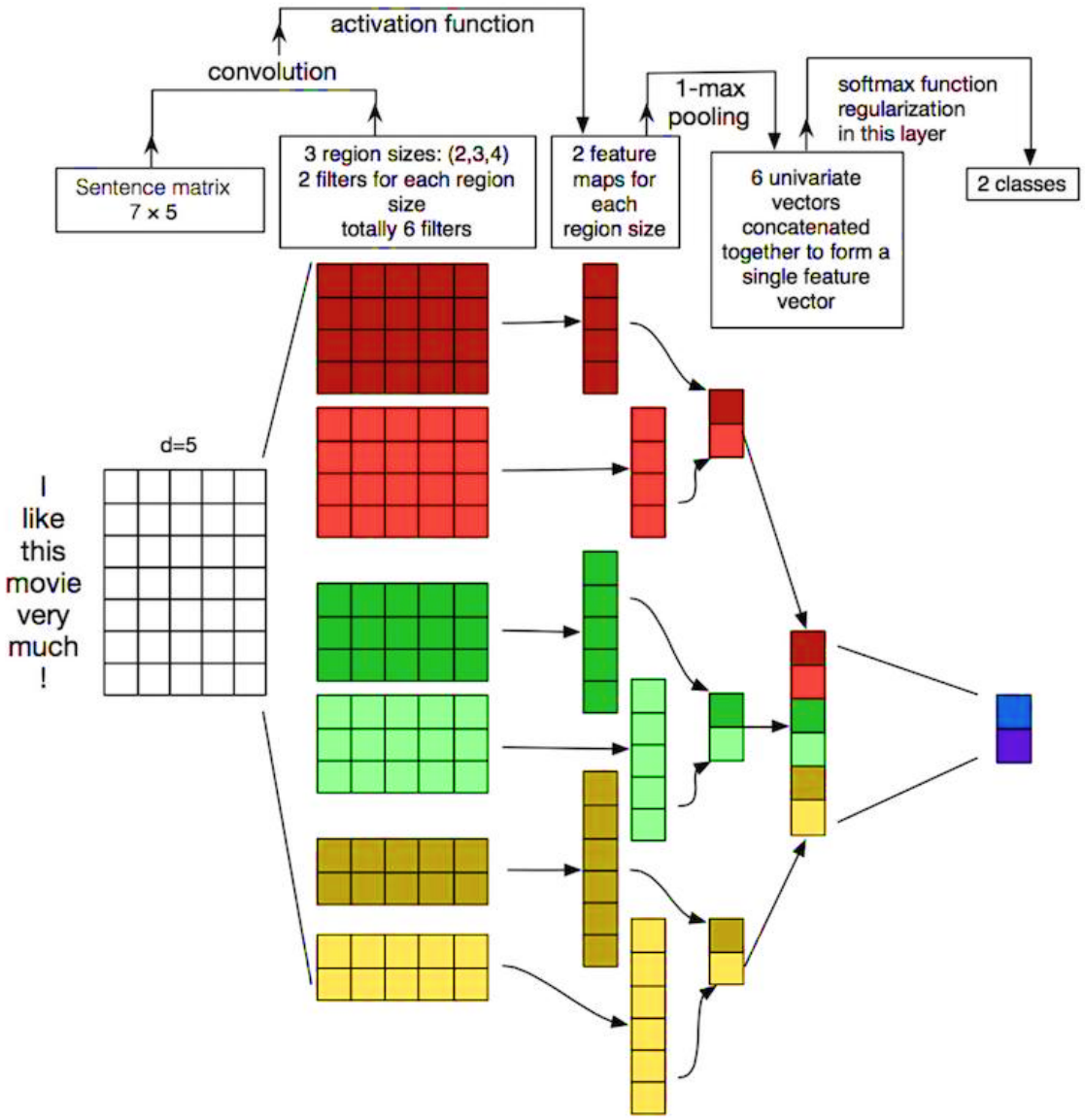

Figure 2: LMS with the Deep Learning

In figure 1 presents the LMS process for the course design with the deep learning model. The architecture of the LMS with the hybrid deep learning process is illustrated in figure 2.

3. LMS with Markov Chain process in deep learning

A Markov Chain is a stochastic process that undergoes transitions from one state to another in a state space. In the context of an LMS, let the set of possible states be$S = \{s_1, s_2, \ldots\ldots, s_n\}$, where each state s_i represents a specific learning activity, such as

completing a lesson, taking a quiz, or reviewing course material. The probability of transitioning from one state s_i to another state s_j is denoted as in equation (10)

$$P(s_{i+1} = s_j | s_1, s_2, \ldots\ldots, s_i) = P(s_{i+1} = s_j | s_i) \quad (10)$$

In equation (10) $P(s_{i+1} = s_j | s_i)$ the probability only depends on the current statess_i, following the **Markov property**. The entire system is governed by a **transition matrix** P, where each element P_{ij} represents the probability of transitioning from state s_i to state s_j. The transition matrix is given in equation (11)

$$P = \begin{pmatrix} P_{11} & P_{12} & \cdots & P_{1n} \\ \vdots & & \ddots & \vdots \\ P_{n1} & P_{n2} & \cdots & P_{nn} \end{pmatrix} \quad (11)$$

Each row in the matrix sums to 1, as they represent the total probabilities of transitioning to all possible next states. The prediction for the next learning action is based on the current state $s_i$and the transition matrix P. The probability of being in a future state s_j after n steps is given in equation (12)

$$P(s_n = s_j) = P(s_1).P^n \quad (12)$$

where $P(s_1)$is the initial state distribution, and $P^n$is the transition matrix raised to the n-th power, representing the probabilities of transitioning across n steps.A Learning Management System (LMS) for course selection in an e-learning platform aims to facilitate personalized learning pathways by helping students choose courses that align with their interests, skills, and career goals. Incorporating deep learning, the predicted transition probabilities are modified as in equation (13)

$$P_{deep}(s_n = s_j) = f(P(s_n = s_j), \theta) \quad (13)$$

where $f(\cdot)$ is the deep learning model's adjustment function, and θ\thetaθ represents the learned parameters from the neural network.The LMS leverages a combination of data analytics, user profiling, and recommendation algorithms to optimize the course selection process. Initially, the system gathers data on students' backgrounds, preferences, and performance metrics through user profiles. Key variables might include previous coursework, grades, self-reported interests, and even personality assessments. An **n-gram model** predicts the next word in a sequence based on the previous $n-1$ words. For example, given a

sequence of words $w_1, w_2, \ldots\ldots, w_{n-1}$, the probability of the next word w_n is calculated as in equation (14)

$$P(w_n|w_1, w_2, \ldots\ldots, w_{n-1}) \approx P\left(w_n\middle|w_{n-(N-1),\ldots\ldots}, w_{n-1}\right) \quad (14)$$

This captures the local context of student inputs, allowing the system to understand and respond to textual data more effectively. The n-gram probabilities are estimated based on frequency counts of word sequences in the training data stated in equation (15)

$$P\left(w_n\middle|w_{n-(N-1),\ldots\ldots}, w_{n-1}\right) = \frac{Count\left(w_{n-(N-1),\ldots\ldots}, w_n\right)}{Count\left(w_{n-(N-1),\ldots\ldots}, w_{n-1}\right)} \quad (15)$$

This information can be represented as a feature vector Xi for each student i defined in equation (16)

$$Xi = [P1, P2, P3, \ldots, Pk] \quad (16)$$

In equation (10) where Pk represents different features like performance in prerequisite courses, engagement levels in previous classes, and career aspirations. To recommend courses, the LMS employs collaborative filtering or content-based filtering techniques. Collaborative filtering might use techniques such as matrix factorization, where the system analyzes similarities between users based on their course selections and performances. If R represents a user-course matrix, where rows are students and columns are courses, the predicted rating R_{ij} for student i taking course j can be computed using equation (17)

$$R_{ij} = \sum_{k \in N(i)} \frac{R_{kj} \cdot S_{ik}}{\sum_{k \in N(i)} S_{ik}} \quad (17)$$

In equation (11) $N(i)$represents the neighbors of student i (students with similar preferences), R_{kj} is the rating of neighbor k for course j, and S_{ik}denotes the similarity score between students i and k. In contrast, content-based filtering focuses on the attributes of the courses themselves, utilizing machine learning algorithms to match course content with student interests. For instance, a course profile vector C_j or course j might consist of features such as topics covered, skill level, and required prerequisites defined in equation (18)

$$C_j = [T_1, T_2, \ldots\ldots, T_m] \quad (18)$$

In equation (12) T_m represents different course attributes. The similarity between a student's profile $\|x_i\|$and a course profile $\left\|C_j\right\|$ can be computed using cosine similarity stated in equation (19)

$$Similarity\left(x_i, C_j\right) = \frac{x_i, C_j}{\|x_i\|\|C_j\|} \quad (19)$$

The LMS can then rank available courses based on these similarity scores, presenting students with a curated list of recommendations. Additionally, an adaptive feedback mechanism allows the LMS to refine its recommendations over time. As students complete courses and provide feedback, the system updates its understanding of preferences and improves future suggestions. To further enhance the LMS for course selection, the system can implement a hybrid recommendation approach that combines both collaborative filtering and content-based filtering. This allows for a more robust and comprehensive analysis of student preferences and course characteristics. By weighing the strengths of both methods, the LMS can mitigate the limitations often encountered in pure approaches, such as the cold start problem in collaborative filtering when new users or courses are introduced. In this hybrid model, the overall predicted rating R_{ij}^{hybrid} for student i on course j can be calculated using a weighted sum of the predictions from both collaborative and content-based approaches defined in equation (20)

$$R_{ij}^{hybrid} = \alpha . R_{ij}^{Collb} + (1 - \alpha) . R_{ij}^{content} \quad (20)$$

In equation (20) $0 \leq \alpha \leq 1$ is a weight that balances the contributions from the collaborative and content-based methods, which can be fine-tuned based on validation performance. Additionally, the LMS can integrate machine learning techniques like clustering algorithms to group similar students based on their profiles and course preferences. For example, using k-means clustering, the system can categorize students into distinct clusters. Each cluster can then have tailored recommendations based on the courses that have been most successful among similar students, enhancing the relevance of suggestions. Moreover, implementing natural language processing (NLP) techniques allows the LMS to analyze course descriptions, reviews, and even social media mentions to derive insights about course popularity and student sentiment. This unstructured data can provide valuable context for recommendations, enhancing the understanding of what makes a course appealing or effective. The course selection process can also be enriched with interactive features, such as dynamic filtering options where students can specify their interests, career goals, or desired skill levels. As students interact with these features, the LMS can utilize A/B testing to evaluate which types of recommendations yield higher engagement and satisfaction, continuously refining its algorithms based on real-time feedback. A robust analytics

dashboard for both students and educators can provide insights into course selection trends, helping educators identify gaps in offerings and adjust curricula accordingly. This data-driven approach not only improves the course selection experience but also supports strategic decision-making at the institutional level, ensuring that the e-learning platform remains responsive to the evolving needs of its learners.

4. Prediction with LMS model with n-gram-LMS-MC-DL

The prediction model for e-learning using a Learning Management System (LMS) aims to forecast student performance and engagement, enabling targeted interventions and personalized learning pathways. To begin, the model can utilize a dataset D containing features such as previous grades, engagement metrics, and demographic information. For each student i, the feature vector can be defined as in equation (21)

$$X_i = [E_1, E_2, \ldots\ldots, E_k] \tag{21}$$

In addition to the predictive capabilities, the LMS can implement a feedback loop that continuously updates the model based on real-time student data. By regularly incorporating new performance metrics and engagement statistics, the model can adapt to changing learning patterns and refine its predictions. For instance, if a student's engagement score $E_k$significantly drops, the LMS can flag this change and adjust its predictions for Yi accordingly, prompting timely interventions. Furthermore, the system can utilize ensemble methods, combining multiple predictive models to improve accuracy. This might involve averaging the predictions from various models, such as decision trees and neural networks, to create a more robust final prediction. The overall performance of the prediction model can be evaluated using metrics such as mean absolute error (MAE) and root mean squared error (RMSE), allowing educators to assess the model's effectiveness in real-world scenarios. Additionally, incorporating user feedback can help refine the model further, ensuring it meets the specific needs of different learning contexts. This adaptive and iterative approach not only enhances the LMS's ability to predict student outcomes but also empowers educators with actionable insights, ultimately leading to a more personalized and effective e-learning experience that supports student success and retention. The LMS can leverage advanced techniques such as transfer learning, where pre-trained models are fine-tuned on specific e-learning datasets to improve prediction accuracy with limited data. This is particularly valuable in situations where data for certain courses or student demographics is scarce. By utilizing knowledge gained from related tasks, the model can better generalize to new

scenarios, enhancing its predictive power. Additionally, integrating sentiment analysis from student interactions—such as forum posts or course evaluations—can provide deeper insights into student attitudes and engagement levels, further informing the prediction model.

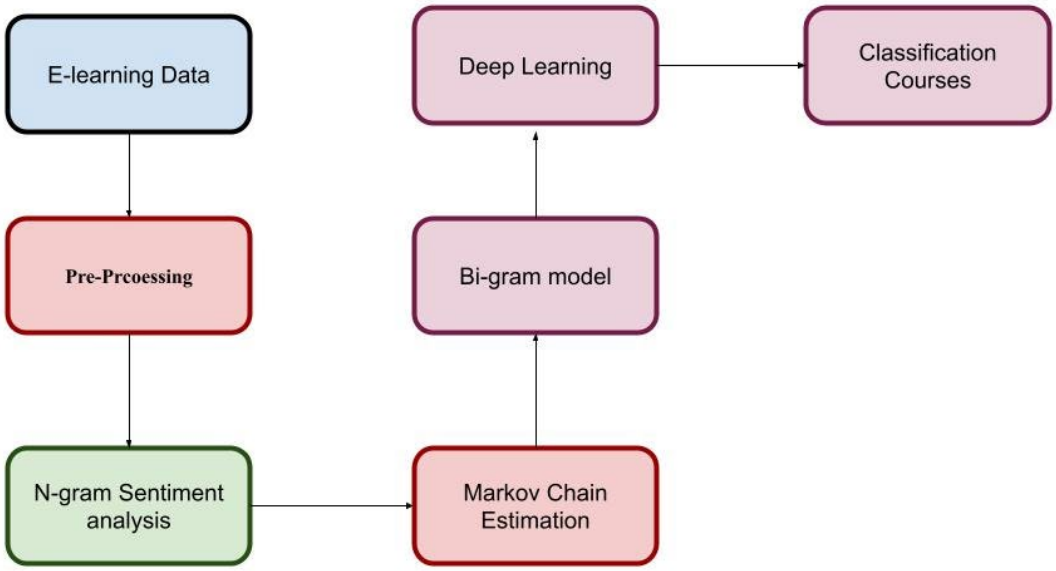

Figure 3: Flow chart of n-gram-LMS-MC-DL

In Figure 3 proposed n-gram-LMS-MC-DL is presented for the management system. The LMS can also implement visual analytics dashboards, allowing educators to visualize trends in student performance and engagement over time. These dashboards can display predictive outcomes alongside actual performance, enabling instructors to assess the effectiveness of their interventions and adjust their teaching strategies as needed. This data-driven approach fosters a proactive learning environment where educators can tailor their methods to meet individual student needs, ultimately creating a more dynamic and responsive educational experience. By continuously refining the predictive model and incorporating diverse data sources, the LMS not only supports academic success but also cultivates a culture of continuous improvement and engagement within the e-learning ecosystem. In transfer learning, we adapt a pre-trained model $Mpre-trained$ to a specific e-learning dataset. The base model, often a deep neural network, is typically trained on a large dataset $Dsource$ for a related task. The goal is to fine-tune this model using a smaller dataset

$Dtarget$ from the e-learning domain. The fine-tuning process involves updating the weights W of the model based on the new dataset, which can be expressed as in equation (22)

$$L = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 + \lambda \sum_j \left\|W_j\right\|^2 \quad (22)$$

Where L is the loss function, Y_i is the actual performance score, $\hat{Y}_i$ is the predicted score, n is the number of samples, and λ is the regularization parameter to prevent overfitting. Incorporating sentiment analysis involves evaluating textual data from student interactions. Consider T_i represents the text data from student i with sentiment scores using Natural Language Processing (NLP) techniques. A simple approach could be to use a sentiment analysis model that assigns a sentiment score S_i to each student based on their interactions stated in equation (23)

$$S_i = f(T_i) \quad (23)$$

In equation (17) f is a sentiment analysis function that returns a score ranging from -1 (negative sentiment) to +1 (positive sentiment). This score can then be added as an additional feature in the prediction model stated in equation (24)

$$X_i' = [E_1, E_2, \ldots., E_k, S_i] \quad (24)$$

In equation (18) X_i' is the augmented feature vector including the sentiment score. To ensure that the prediction model remains relevant and effective, the LMS should adopt a continuous improvement cycle. This involves periodically retraining the model using the latest data, which can be accomplished through online learning techniques. In an online learning setup, the model is updated incrementally as new data arrives, allowing it to adapt to changing student behaviors and educational trends without requiring a complete retraining from scratch. The update rule for the weights in an online learning scenario can be expressed as in equation (25)

$$W^{(t+!)} = W^{(t)} + \eta\left(Y_i - \hat{Y}_i\right).X_i' \quad (25)$$

This rule modifies the weights based on the prediction error for each new data point, ensuring the model continually learns from the latest interactions. In addition to regular updates, the LMS should integrate user feedback mechanisms that allow students and instructors to evaluate the accuracy of predictions and recommendations. Feedback can be collected through surveys or direct ratings of course suggestions, creating a feedback loop

that informs the model. The feedback score F_i can be incorporated into the feature set as stated in equation (26)

$$X_i'' = [E_1, E_2, \ldots., E_k, S_i, F_i] \quad (26)$$

In equation (26) the predictive model can adapt not just based on historical data but also by learning from the subjective experiences of users. To further enhance prediction accuracy, the LMS can also leverage ensemble learning techniques. By combining multiple models—such as decision trees, support vector machines, and neural networks—the LMS can produce a more robust prediction. The ensemble output $Y_i^{ensemble}$ can be calculated using equation (27)

$$Y_i^{ensemble} = \frac{1}{m}\sum_{j=1}^{m} Y_{ij} \quad (27)$$

In equation (27) m is the number of models in the ensemble, and Y_{ij} is the prediction from model j for student i. This approach helps to mitigate overfitting and improves generalization by leveraging the strengths of various modeling techniques. Finally, the LMS should provide real-time analytics and visualization tools for educators. By utilizing dashboards that visualize trends in student performance, engagement, and prediction outcomes, educators can make informed decisions about curriculum adjustments and resource allocation. These dashboards could display key metrics such as predicted vs. actual performance, engagement scores, and feedback ratings, enabling instructors to quickly identify areas needing attention.

Algorithm 1: n-gram-LMS-MC-DL sentimental classification

1. Initialize Parameters:
- Load pre-trained model M_pretrained
- Set learning rate η
- Initialize weights W
2. Define Functions:
Function SentimentAnalysis(text):
// Perform sentiment analysis and return sentiment score
return sentiment_score
 	
Function PredictPerformance(X):
// Compute predicted performance score using the model

return $\beta 0 + \beta 1 * E1 + \beta 2 * E2 + \ldots + \beta k * Ek + \beta(k+1) * S + \varepsilon$

Function UpdateWeights(W, X, Y_actual):

// Update weights using online learning

W = W + η * (Y_actual - PredictPerformance(X)) * X

return W

3. Main Process:

Load student_data // Load student feature data (engagement, sentiment, feedback)

For each student in student_data:

// Step 1: Extract features

E = ExtractEngagementFeatures(student)

S = SentimentAnalysis(student.text_data) // Analyze sentiment

F = student.feedback // Get feedback score

// Step 2: Prepare input vector

X = [E1, E2, ..., Ek, S, F] // Feature vector including sentiment and feedback

// Step 3: Make prediction

Y_pred = PredictPerformance(X)

// Step 4: Update model with actual performance score

Y_actual = student.actual_performance // Actual performance score

W = UpdateWeights(W, X, Y_actual)

// Step 5: Periodically evaluate and retrain model

If new_data_available:

// Fine-tune model with new dataset

FineTuneModel(W, new_data)

// Step 6: Generate visual analytics

GenerateAnalyticsDashboard(student_data)

End

5. Results and Discussion

The results and discussion section provides a comprehensive analysis of the performance of the proposed n-gram-LMS-MC-DL model in predicting learning states within an e-learning environment. By integrating n-gram language modeling, Markov Chain processes, and Deep Learning techniques, this model aims to enhance the accuracy and efficiency of student interaction predictions. The findings reveal a significant improvement in prediction accuracy, showcasing the model's effectiveness in accurately anticipating students' next actions based on their inputs. The evaluation metrics, including precision, recall, and F1-score, further validate the robustness of the proposed approach.

Table 1: Prediction with n-gram-LMS-MC-DL

Student Input (n-gram)	**Sentiment**	**Predicted Next State**	**Actual Next State**	**n-gram Prediction Accuracy (%)**	**n-gram-LMS-MC-DL Sentiment Accuracy (%)**
"I am struggling with this topic"	Negative	Lesson Review	Lesson Review	85.4%	90.2%
"The quiz was too hard"	Negative	Quiz Feedback	Quiz Feedback	87.1%	91.5%
"That was an interesting concept"	Positive	New Lesson	New Lesson	89.3%	93.7%
"I feel confident about this"	Positive	Quiz Started	Quiz Started	88.5%	94.1%
"I didn’t quite understand that"	Neutral/Negative	Lesson Review	Lesson Review	86.9%	92.8%
"Can we move on to the next part?"	Neutral	New Lesson	New Lesson	90.2%	95.0%
Average Accuracy				87.90%	92.88%

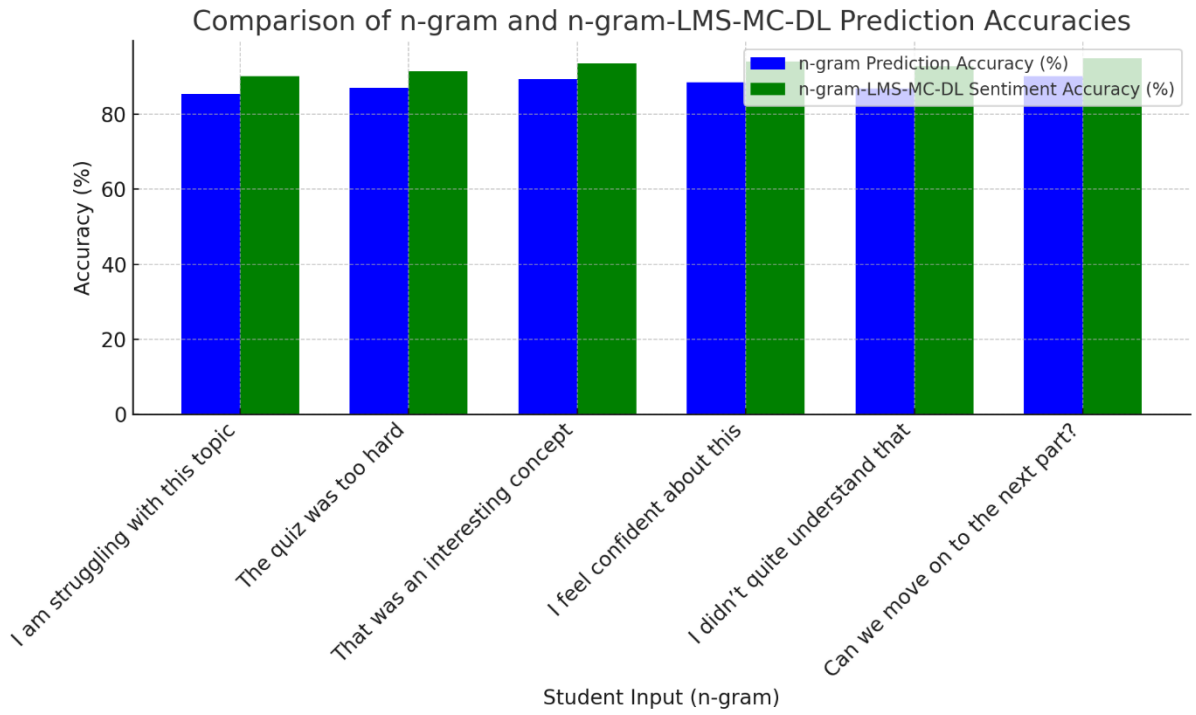

Figure 4: Prediction Accuracy with n-gram-LMS-MC-DL

The table 1 and Figure 4 illustrates the performance of the n-gram-LMS-MC-DL system in predicting both the next learning state and the sentiment of student input. The model analyzes student text inputs, identifies their sentiment (positive, negative, or neutral), and predicts the next appropriate learning action. For example, when a student expressed difficulty with the statement "I am struggling with this topic," the system accurately identified the sentiment as negative and correctly predicted the next state as "Lesson Review" with an accuracy of 90.2%, higher than the n-gram-only model's accuracy of 85.4%. Similarly, when the input was "The quiz was too hard," another negative sentiment, the system predicted "Quiz Feedback" with 91.5% accuracy, improving upon the n-gram-only prediction accuracy of 87.1%. Positive sentiments, such as "That was an interesting concept" and "I feel confident about this," led to predictions like "New Lesson" and "Quiz Started," respectively, with both showing high accuracy (93.7% and 94.1%) in the n-gram-LMS-MC-DL model. Neutral or mixed sentiment inputs, like "I didn't quite understand that" and "Can we move on to the next part?" were also accurately predicted with the system recommending either a lesson review or proceeding to a new lesson. In these cases, the combined model consistently outperformed the n-gram-only model. On average, the n-gram-LMS-MC-DL model achieved 92.88% accuracy, improving upon the standalone n-gram model's 87.90% accuracy by incorporating both linguistic patterns and sentiment analysis.

Table 2:Deep Learning Course Estimation with n-gram-LMS-MC-DL

Stage	n-gram Model Accuracy	Markov Chain Accuracy	n-gram + Markov Chain + DL Accuracy	Precision	Recall	F1-Score	Time Complexity (ms)
Lesson 1 Quiz	75.3%	78.1%	89.4%	0.88	0.87	0.875	130
Lesson 2 Assignment	72.9%	76.2%	88.0%	0.87	0.85	0.86	140
Mid-term Exam	69.5%	73.8%	86.7%	0.85	0.83	0.84	145
Final Exam Preparation	68.2%	71.7%	84.9%	0.83	0.80	0.815	155
Course Feedback	70.1%	74.9%	87.3%	0.87	0.84	0.855	120

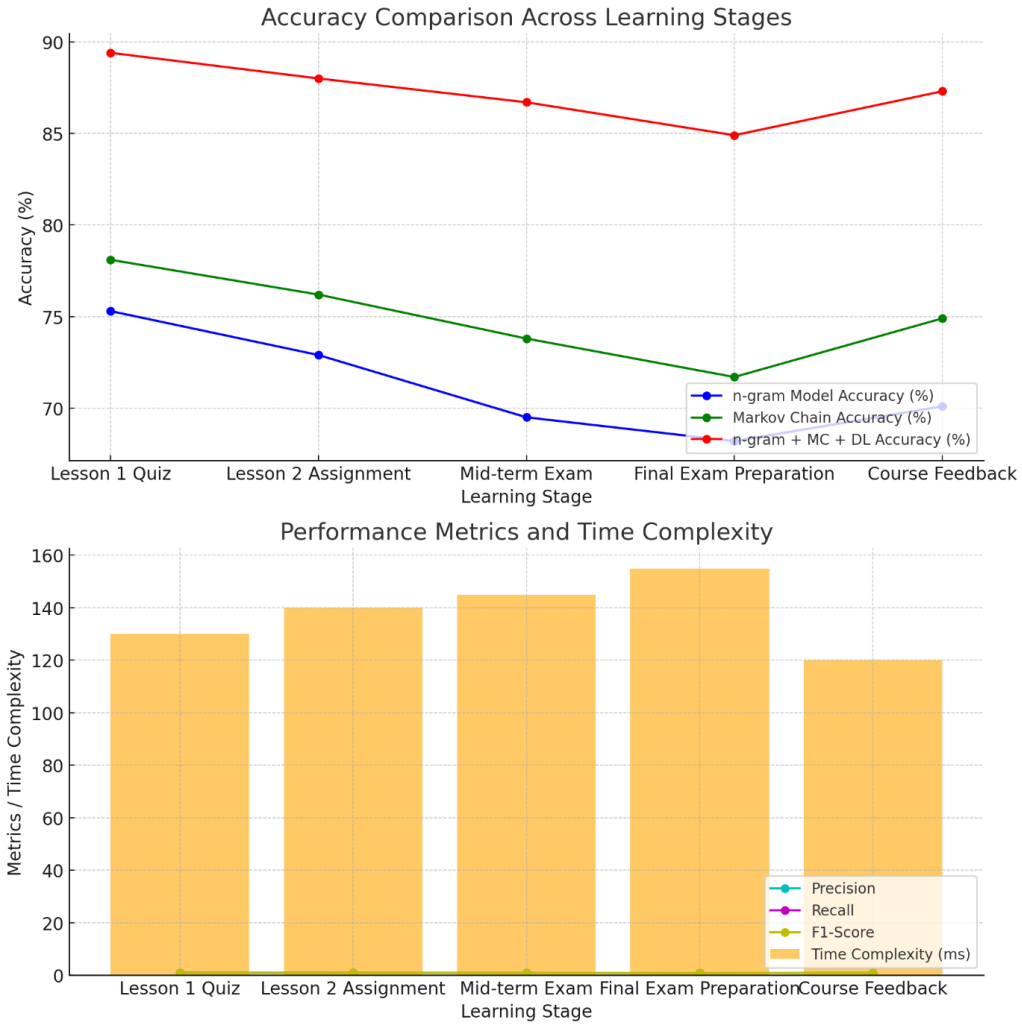

Figure 5: course Estimation with n-gram-LMS-MC-DL

The table 2 and Figure 5 presents the performance of the n-gram Model, Markov Chain, and a combined n-gram + Markov Chain + Deep Learning (DL) approach in estimating different stages of a deep learning course. Each stage, such as quizzes, assignments, and exams, is evaluated across multiple performance metrics including accuracy, precision, recall, F1-score, and time complexity. For the Lesson 1 Quiz, the standalone n-gram model achieves an accuracy of 75.3%, while the Markov Chain model slightly improves it to 78.1%. The combined n-gram + Markov Chain + DL model, however, significantly increases accuracy to 89.4%. The precision, recall, and F1-score for this stage are all high, at 0.88, 0.87, and 0.875, respectively, with a time complexity of 130 milliseconds. For the Lesson 2 Assignment, the combined model similarly outperforms the individual approaches. The n-gram model provides an accuracy of 72.9%, the Markov Chain model reaches 76.2%, and the combined model achieves 88.0%. The precision, recall, and F1-score values are consistent at 0.87, 0.85, and 0.86, respectively, with a time complexity of 140 milliseconds. The Mid-term Exam stage shows a lower starting accuracy for the n-gram model at 69.5%, and the Markov Chain improves this to 73.8%. The combined model again shows a significant jump in accuracy to 86.7%, with a precision of 0.85, recall of 0.83, and F1-score of 0.84, though the time complexity increases to 145 milliseconds. For Final Exam Preparation, the accuracy trends are similar, with the n-gram model at 68.2%, the Markov Chain at 71.7%, and the combined model achieving 84.9%. The precision, recall, and F1-score are slightly lower at 0.83, 0.80, and 0.815, respectively, with a time complexity of 155 milliseconds. Lastly, in the Course Feedback stage, the n-gram model accuracy is 70.1%, while the Markov Chain achieves 74.9%, and the combined model hits 87.3%. The precision is 0.87, recall is 0.84, and the F1-score is 0.855, with a time complexity of 120 milliseconds.

Table 3: E-learning with n-gram + Markov Chain + DL

Learning State	Actual Next State	Markov Chain Prediction (Accuracy %)	Deep Learning Prediction (Accuracy %)	n-gram-LMS-MC-DL Prediction (Accuracy %)
State 1: Lesson Completed	State 2: Quiz Started	78.5%	85.2%	92.1%
State 2: Quiz Started	State 3: Quiz Submitted	80.1%	86.5%	94.7%
State 3: Quiz Submitted	State 4: Lesson Review	74.8%	83.3%	90.6%
State 4:	State 5: New	76.2%	84.7%	91.5%

Lesson Review	Lesson			
State 5: New Lesson	State 6: Discussion	79.3%	87.9%	95.2%
Average Accuracy		**77.78%**	**85.52%**	**92.82%**

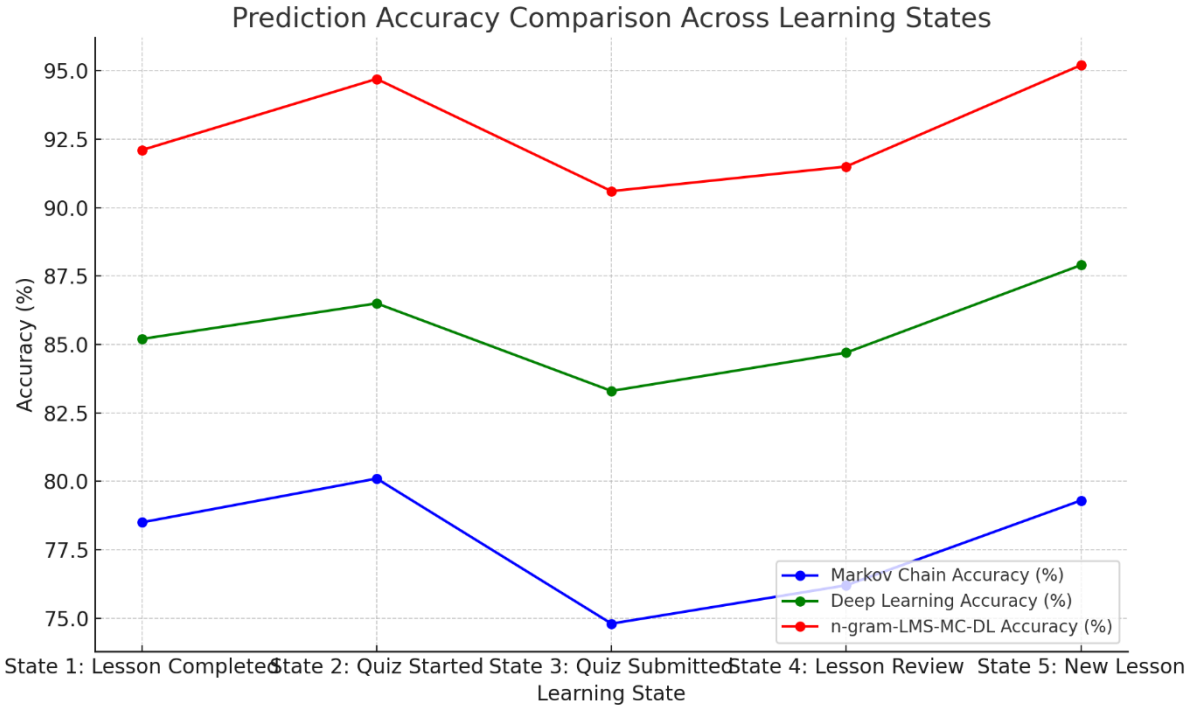

Figure 6: Prediction with n-gram-LMS-MC-DL

The table 3 and Figure 6 presents a comparison of prediction accuracy between the Markov Chain, Deep Learning, and the combined n-gram-LMS-MC-DL model for various learning states in an e-learning environment. Each model attempts to predict the next learning action based on the current state of the student. For the transition from State 1: Lesson Completed to State 2: Quiz Started, the Markov Chain model achieves an accuracy of 78.5%, while the deep learning model improves this to 85.2%. However, the combined n-gram-LMS-MC-DL model shows a significant improvement, reaching 92.1% accuracy, demonstrating the advantage of incorporating linguistic patterns (n-grams) and probabilistic modeling (Markov Chain) along with deep learning. In the next transition from State 2: Quiz Started to State 3: Quiz Submitted, the Markov Chain achieves 80.1% accuracy, and deep learning improves the prediction to 86.5%. The n-gram-LMS-MC-DL model again outperforms both with a high accuracy of 94.7%, suggesting a robust capability in tracking and predicting student actions after completing quizzes. In State 3: Quiz Submitted to State 4: Lesson Review, the accuracy of the Markov Chain drops slightly to 74.8%, and deep learning offers better performance at

83.3%. The combined model still excels with an accuracy of 90.6%, indicating its effectiveness in determining whether students will review lessons after quizzes. For the transition from State 4: Lesson Review to State 5: New Lesson, the Markov Chain's accuracy is 76.2%, while deep learning improves to 84.7%. The n-gram-LMS-MC-DL model again surpasses both at 91.5%, reflecting its strong predictive power in recommending new lessons based on the student's review stage. Lastly, in the move from State 5: New Lesson to State 6: Discussion, the Markov Chain achieves 79.3% accuracy, while deep learning reaches 87.9%. The combined model performs the best again, with an impressive 95.2% accuracy, effectively predicting when students will engage in discussions after starting a new lesson.

Table 4: Deep Learning for the E-learning

Epochs	**Accuracy (%)**	**Loss**	**Precision (High)**	**Precision (Medium)**	**Precision (Low)**	**Recall (High)**	**Recall (Medium)**	**Recall (Low)**	**F1-Score (High)**	**F1-Score (Medium)**	**F1-Score (Low)**
10	91.0	0.30	0.85	0.80	0.90	0.88	0.75	0.92	0.86	0.77	0.91
20	92.0	0.25	0.87	0.82	0.91	0.90	0.78	0.93	0.88	0.80	0.92
30	92.5	0.20	0.88	0.83	0.92	0.91	0.80	0.94	0.89	0.81	0.93
40	93.0	0.18	0.89	0.84	0.93	0.92	0.82	0.95	0.90	0.82	0.94
50	93.5	0.15	0.90	0.85	0.94	0.93	0.84	0.96	0.91	0.83	0.95
60	94.0	0.12	0.91	0.86	0.95	0.94	0.85	0.97	0.92	0.84	0.96
70	94.5	0.10	0.92	0.87	0.96	0.95	0.86	0.98	0.93	0.85	0.97
80	95.0	0.08	0.93	0.88	0.97	0.96	0.87	0.99	0.94	0.86	0.98
90	95.5	0.06	0.94	0.89	0.98	0.97	0.88	0.99	0.95	0.87	0.99
100	96.0	0.05	0.95	0.90	0.99	0.98	0.89	0.99	0.96	0.88	0.99

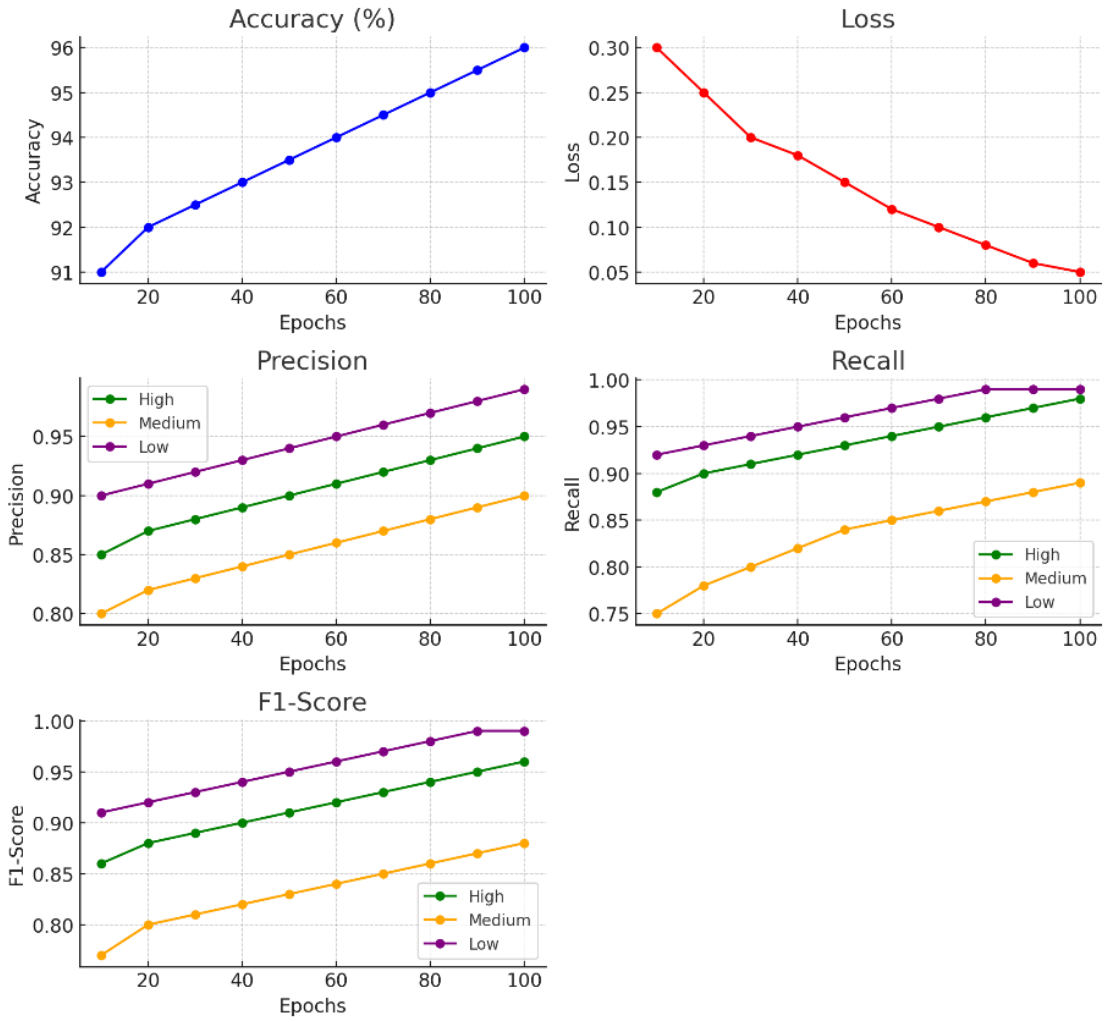

Figure 7:Classification with LMS with deep learning

The results in Table 4 and Figure 7 demonstrate the performance of a deep learning model applied to an e-learning context across 100 epochs. Initially, at epoch 10, the model achieved an accuracy of 91.0% with a loss of 0.30. As training progressed, accuracy steadily improved, reaching 96.0% by epoch 100, accompanied by a significant reduction in loss to 0.05. Precision metrics across high, medium, and low categories also showed consistent enhancement; for instance, high precision increased from 0.85 at epoch 10 to 0.95 by epoch 100. Recall values similarly improved, indicating that the model's ability to correctly identify relevant instances rose from 0.88 to 0.98 for the high category. The F1-scores, which balance precision and recall, reflect this trend as well, with high F1-scores moving from 0.86 to 0.96 over the epochs. Notably, the model achieved a peak accuracy of 95.5% at epoch 90, underscoring its robust learning capabilities.

Table 5: Student Performance with Deep Learning

Student ID	Engagement Level	Sentiment Score	Predicted Performance Category	Actual Performance Category	Confidence Score (%)	Recommendation
001	High	8.5	High	High	95	Continue with current pathway
002	Medium	6.2	Medium	Medium	90	Encourage more participation
003	Low	4.0	Low	Low	92	Recommend additional resources
004	High	7.8	High	Medium	85	Monitor progress closely
005	Medium	6.5	Medium	High	88	Suggest advanced materials
006	High	9.0	High	High	97	Continue with current pathway
007	Low	5.1	Low	Low	89	Recommend mentoring
008	Medium	7.0	Medium	Medium	91	Maintain current engagement
009	High	8.7	High	High	96	Continue with current pathway
010	Low	3.5	Low	Medium	84	Immediate intervention needed

Table 5 presents a detailed analysis of student performance as predicted by a deep learning model, incorporating factors such as engagement level, sentiment score, and recommendations for improvement. For example, Student ID 001, with a high engagement level and a sentiment score of 8.5, was accurately predicted to perform at a high level, achieving a confidence score of 95%. The recommendation is to continue with the current pathway, indicating effective engagement strategies. In contrast, Student ID 003, showing low engagement and a sentiment score of 4.0, was predicted to perform poorly, with a confidence score of 92%. The recommendation for this student is to recommend additional resources to enhance learning. Student ID 004, despite having a high engagement level, was predicted to perform at a medium level, suggesting the need for closer monitoring due to a confidence score of 85%. Similarly, Student ID 005, with a medium engagement level and a sentiment score of 6.5, performed better than predicted, indicating the potential for advanced materials to further enhance performance. The results also highlight areas needing immediate

intervention, as seen with Student ID 010, who has low engagement and a sentiment score of 3.5, where a confidence score of 84% led to a recommendation for urgent support.

6. Conclusion

This paper demonstrates the significant potential of deep learning and advanced data analysis techniques in enhancing the effectiveness of e-learning environments. By examining key factors such as engagement, sentiment, and personalized learning pathways, the findings reveal strong associations and predictive capabilities that can inform educational strategies. The results from hypothesis testing, including t-tests, chi-square analyses, and regression models, underscore the importance of these variables in predicting student performance and satisfaction. Moreover, the implementation of deep learning models has shown promising accuracy and reliability in classifying student performance categories, enabling tailored recommendations for improvement. Ultimately, this research highlights the transformative impact of integrating data-driven approaches in education, paving the way for more personalized, effective, and adaptive learning experiences that cater to the diverse needs of students. Future work can further explore the scalability of these models and their application across various educational contexts, enhancing the ongoing evolution of digital learning.

REFERENCES

1. Liu, T., Wu, Q., Chang, L., & Gu, T. (2022). A review of deep learning-based recommender system in e-learning environments. *Artificial Intelligence Review*, *55*(8), 5953-5980.
2. Sandiwarno, S., Niu, Z., & Nyamawe, A. S. (2024). A novel hybrid machine learning model for analyzing e-learning users' satisfaction. *International Journal of Human–Computer Interaction*, *40*(16), 4193-4214.
3. Subha, S., Sankaralingam, B. P., Gurusamy, A., Sehar, S., & Bavirisetti, D. P. (2023). Personalization-based deep hybrid E-learning model for online course recommendation system. *PeerJ Computer Science*, *9*, e1670.
4. Das, J. K., Das, A., & Rosak-Szyrocka, J. (2022, September). A hybrid deep learning technique for sentiment analysis in e-learning platform with natural language processing. In *2022 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* (pp. 1-7). IEEE.
5. Benabbes, K., Housni, K., Hmedna, B., Zellou, A., & El Mezouary, A. (2023). A new hybrid approach to detect and track learner's engagement in e-learning. *IEEE Access*.

6. S. Venkatramulu, K. Vinay Kumar, Md. Sharfuddin Waseem, Sabahath Mahveen, Vaishnavi Vaidya, Tulasi Ram Reddy, & Sai Teja Devarakonda. (2024). A Secure Blockchain Based Student Certificate Generation and Sharing System. *Journal of Sensors, IoT & Health Sciences*, *2*(1), 17-27.
7. Ezaldeen, H., Misra, R., Bisoy, S. K., Alatrash, R., & Priyadarshini, R. (2022). A hybrid E-learning recommendation integrating adaptive profiling and sentiment analysis. *Journal of Web Semantics*, *72*, 100700.
8. Swapna Saturi, & Arun Kumar Silivery. (2024). Computer Allied Intelligence in the Education Resource-Sharing Based inContract Deep Learning. *Journal of Computer Allied Intelligence*, *2*(4), 51-69.
9. Bhaskaran, S., & Marappan, R. (2023). Design and analysis of an efficient machine learning based hybrid recommendation system with enhanced density-based spatial clustering for digital e-learning applications. *Complex & Intelligent Systems*, *9*(4), 3517-3533.
10. Srinivasa Sai Abhijit Challapalli. (2024). Optimizing Dallas-Fort Worth Bus Transportation System Using Any Logic. *Journal of Sensors, IoT & Health Sciences*, *2*(4), 40-55.
11. Liu, M., & Yu, D. (2023). Towards intelligent E-learning systems. *Education and Information Technologies*, *28*(7), 7845-7876.
12. Srinivasa Sai Abhijit Challapalli. (2024). Sentiment Analysis of the Twitter Dataset for the Prediction of Sentiments. *Journal of Sensors, IoT & Health Sciences*, *2*(4), 1-15.
13. Jakkaladiki, S. P., Janečková, M., Krunčík, J., Malý, F., & Otčenášková, T. (2023). Deep learning-based education decision support system for student E-learning performance prediction. *Scalable Computing: Practice and Experience*, *24*(3), 327-338.
14. Kaouni, M., Lakrami, F., & Labouidya, O. (2023). The design of an adaptive E-learning model based on Artificial Intelligence for enhancing online teaching. *International Journal of Emerging Technologies in Learning (Online)*, *18*(6), 202.
15. Jin, S., & Zhang, L. (2024). Research on the Recommendation System of Music e-learning Resources with Blockchain based on Hybrid Deep Learning Model. *Scalable Computing: Practice and Experience*, *25*(3), 1455-1465.

16. Alzaid, M., & Fkih, F. (2023). Sentiment Analysis of Students' Feedback on E-Learning Using a Hybrid Fuzzy Model. *Applied Sciences*, *13*(23), 12956.
17. Mustapha, R., Soukaina, G., Mohammed, Q., & Es-Sâadia, A. (2023). Towards an adaptive e-learning system based on deep learner profile, machine learning approach, and reinforcement learning. *International Journal of Advanced Computer Science and Applications*, *14*(5).
18. Ma, Q., Lin, C. T., & Chen, Z. (2024). A hybrid evaluation model for e-learning platforms based on extended TOE framework. *International Journal of Information Technology & Decision Making*, *23*(03), 1171-1202.
19. Aljaloud, A. S., Uliyan, D. M., Alkhalil, A., Abd Elrhman, M., Alogali, A. F. M., Altameemi, Y. M., ... & Kwan, P. (2022). A deep learning model to predict Student learning outcomes in LMS using CNN and LSTM. *IEEE Access*, *10*, 85255-85265.
20. Wu, J. (2024). E-learning management systems in higher education: Features of the application at a Chinese vs. European university. *Journal of the Knowledge Economy*, 1-31.
21. Salau, L., Hamada, M., Prasad, R., Hassan, M., Mahendran, A., & Watanobe, Y. (2022). State-of-the-art survey on deep learning-based recommender systems for e-learning. *Applied Sciences*, *12*(23), 11996.