Journal of Information Systems Engineering and Management
2025, 10(7s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Deep Learning Approach for Predicting Student Academic Performance and Grades Based on C Code Analysis

1 Ass. Prof. Urmila Kadam, 2Dr. Kavita Oza
*1 Assistant Professor, Dr. D. Y. Patil School of MCA, Lohegaon Pune, Maharashtra, India 412 105
*2Associate Professor, Department of Computer Science, Shivaji University, Kolhapur, Maharashtra, India 416 004
Corresponding author email: urmila.kadam@dypic.in, kso_csd@unishivaji.ac.in

ARTICLE INFO

Received: 13Oct 2024

Revised: 07Dec 2024

Accepted: 23Dec 2024

ABSTRACT

This research employs deep learning to enhance student assessment by analyzing the quality and structure of programming assignments, focusing on C code submissions. Traditional grading methods often fail to capture the intricate details of a student's coding abilities, focusing primarily on code functionality over quality and comprehension. To address this limitation, it requires to extracts in-depth metrics from code—such as the total lines of code, use and quality of comments, variable declarations, control structures, and overall readability—and integrates them with traditional academic data like past grades and attendance. Using a Deep Neural network, the model predicts students' grades and academic performance percentages based on these rich, combined inputs, providing a more comprehensive and nuanced evaluation. This innovative approach empowers educators with insights into each student's overall performance and areas needing improvement, enabling personalized feedback and fostering a balanced, skill-based assessment framework that goes beyond conventional grading systems.

Keywords: Deep Learning, Artificial Neural Network (ANN), Performance Prediction, Academic Assessment, Feature Extraction, Skill-Based Assessment.

INTRODUCTION

[1] in today's fast-paced, technology-driven educational environment, programming skills have become indispensable, particularly for students pursuing careers in technical and professional disciplines. However, traditional grading systems often fail to fully capture the depth and complexity of a student's coding abilities. [2][3] these systems typically focus on the functional correctness of the code, neglecting vital aspects such as coding style, logical complexity, problem-solving approaches, and adherence to best programming practices [6] [7]. This research seeks to address these gaps by applying a cutting-edge deep learning approach to assess student performance in C programming, offering a comprehensive and holistic evaluation of their coding skills.[5] [6] [9].

[15,19]The research extracts a wide array of features from students' C code submissions, including metrics such as lines of code, the quality and frequency of comments, the variety and usage of variables, the implementation of control structures and loops, as well as the clarity and accuracy of output labelling.[8,13,14] Each of these features serves as a key indicator of essential programming skills such as code readability, logical structure, complexity management, and the ability to communicate effectively through code[11,12]. By analyzing these features in depth, this research moves beyond conventional assessments, providing insights into a student's ability to write clean, efficient, and maintainable code, [16,20].

[12] Furthermore, by integrating these detailed code-based features with traditional academic data—such as prior grades, attendance records, and other performance metrics—the research utilizes Artificial Neural Networks (ANN) to predict students' overall academic performance and research future grades with high accuracy. [35,25], this predictive model not only provides a more nuanced evaluation of a student's abilities but also allows educators to identify individual strengths and weaknesses, offering targeted, personalized feedback that can guide students toward improvement, [38,29].

Ultimately, [10, 11] this approach revolutionizes the way educators assess and support their students, moving toward a more holistic, skill-based evaluation framework that reflects the real-world competencies required in the programming industry. [22,32] by fostering personalized learning experiences and providing more meaningful assessments, this research prepares students to meet the evolving demands of the digital world. [28, 40] it offers a transformative enhancement to educational practices, creating a more effective and forward-thinking model for assessing and nurturing programming talent.

METHODOLOGY

The methodology for this research follows a comprehensive, multi-faceted approach that integrates advanced feature extraction from students' C code with cutting-edge deep learning techniques to predict academic performance and grades. The process begins with gathering a robust dataset that includes a variety of students' C programming submissions, as well as their academic records, such as grades, attendance, and other relevant performance metrics. This diverse dataset forms the foundation for building the prediction model, allowing for a seamless integration of coding proficiency with academic success, offering a holistic assessment of each student's abilities.

The feature extraction phase involves a detailed analysis of key characteristics within the C code submitted by students. These features include the total number of lines of code, the presence and quality of comments, the diversity and number of variables used, and the overall complexity of the logic, including the number of condition checks and loop constructs. Additional features such as the clarity and precision of output, the correct use of return statements, variable initialization practices, and shorthand notations are also assessed. These features provide profound insights into the coding habits and problem-solving skills of each student, shedding light on their ability to write efficient, maintainable code and their familiarity with best coding practices.

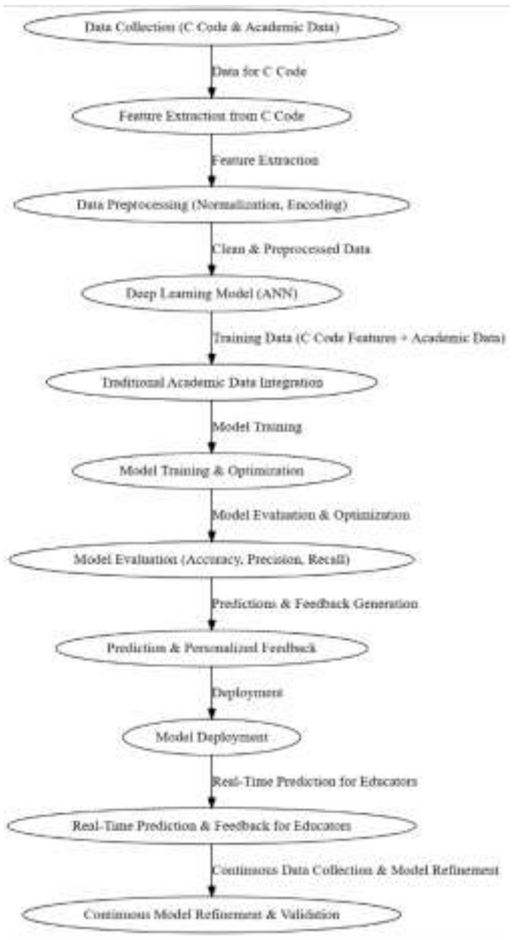
Data Collection (C Code & Academic Data)
Data for C Code
Feature Extraction from C Code
Feature Extraction
Data Preprocessing (Normalization, Encoding)
Clean & Preprocessed Data
Deep Learning Model (ANN)
Training Data (C Code Features + Academic Data)
Traditional Academic Data Integration
Model Training
Model Training & Optimization
Model Evaluation & Optimization
Model Evaluation (Accuracy, Precision, Recall)
Predictions & Feedback Generation
Prediction & Personalized Feedback
Deployment
Model Deployment
Real-Time Prediction for Educators
Real-Time Prediction & Feedback for Educators
Continuous Data Collection & Model Refinement
Continuous Model Refinement & Validation

Figure1.1: Methodology

Once the features are extracted, the data undergoes an extensive preprocessing phase, where numerical features are normalized, missing data is handled, and categorical variables are appropriately encoded. This ensures that the dataset is clean, consistent, and ready for modeling. At this stage, a deep learning model, specifically an Artificial Neural Network (ANN), is employed to learn from the historical data and predict academic performance. The ANN is designed to recognize intricate patterns and correlations between coding practices and academic success, allowing it to predict students' grades and overall performance with high precision. The model's architecture is carefully tuned and optimized to maximize its predictive power, and it is trained iteratively on the data to enhance its accuracy and ensure it generalizes well to unseen data.

To improve the predictive accuracy, traditional academic data such as prior grades, attendance records, and other relevant factors are integrated alongside the features derived from C code submissions. This multi-dimensional approach allows the model to consider a wide range of factors that influence academic performance, creating a more robust and comprehensive prediction system. The model is evaluated using a variety of performance metrics, including accuracy, precision, recall, and F1 score, to ensure it effectively predicts outcomes. Hyper parameter optimization and fine-tuning are applied to enhance the model's performance, ensuring it can consistently provide accurate predictions.

Upon completion of model training and optimization, the system is deployed to predict student performance based on their current C code submissions. The system generates personalized feedback for each student, offering insights into their strengths, weaknesses, and areas for improvement. This feedback is not only informative but also actionable, providing students with clear guidance on how to enhance their coding skills and, by extension, their academic performance. This personalized approach helps foster a deeper understanding of programming concepts and encourages students to adopt better coding practices, which can positively impact their overall academic achievements.

The model is embedded into an intuitive, user-friendly interface, which allows educators to input students' C code and academic data to receive real-time predictions and feedback. This system empowers educators to make data-driven decisions by providing a comprehensive overview of student progress. Predictions are regularly validated by comparing them with actual student performance in subsequent assessments, ensuring the model remains accurate and reliable over time. As new data is gathered, the model is continuously refined and updated, improving its predictive capabilities and ensuring its relevance in future academic scenarios.

In this research work, this methodology offers a transformative approach to predicting student performance by integrating advanced deep learning techniques with in-depth analysis of students' C code and academic data. By combining coding proficiency with traditional academic metrics, the system provides a holistic view of student capabilities, offering a powerful tool for educators to assess performance, identify areas for improvement, and provide personalized feedback.

This approach not only enhances the accuracy and reliability of predictions but also empowers students to improve their coding practices and overall academic outcomes. By fostering a deeper understanding of programming, this methodology prepares students for the ever-evolving demands of the technology industry, helping them become more proficient, adaptable, and innovative developers in the future. Through this innovative system, the research aims to bridge the gap between education and industry requirements, enhancing the quality of education and empowering students to succeed in their academic and professional careers.

The methodology for predicting student academic performance using feature analysis from C code and academic data follows a systematic process. Initially, data is collected from students' C code and academic records. The C code is then analyzed to extract various features such as the number of lines, variables, condition checks, and loop usage. This extracted data is preprocessed to remove noise and normalize the values, making it suitable for model training. The deep learning model, specifically an deep neural network, is trained using both the C code features and traditional academic data, like grades and attendance. The model's performance is evaluated using standard metrics such as accuracy, precision, and recall to ensure its reliability. Once optimized, the model is deployed for real-time predictions, generating personalized feedback for educators to assist in tracking student progress. As new data becomes available, the model is continuously refined and updated to improve prediction accuracy, providing a dynamic and adaptive tool for educational insights.

MODEL DEVELOPMENT

The architecture of the proposed system is designed to provide a seamless interaction between educators and the predictive model for student performance. The system comprises several key components that work collaboratively to deliver accurate predictions based on C code submissions and academic data. The overall architecture is depicted in the diagram below:

The system architecture for predicting student academic performance based on C code analysis and traditional academic data is designed to be modular, scalable, and efficient. It consists of several key components that work together to collect, process, analyze, and predict student outcomes. The architecture is built with a clear separation between data collection, feature extraction, model development, and deployment, ensuring flexibility and ease of maintenance.

At the core of the system is the data collection module, which gathers inputs from two primary sources: the C code submitted by students and the traditional academic data, such as grades, attendance, and participation records. The C code is retrieved from student submissions, which may be stored in a code repository or directly submitted through an interface. Academic data is collected from the student management system or other educational platforms where records are maintained.

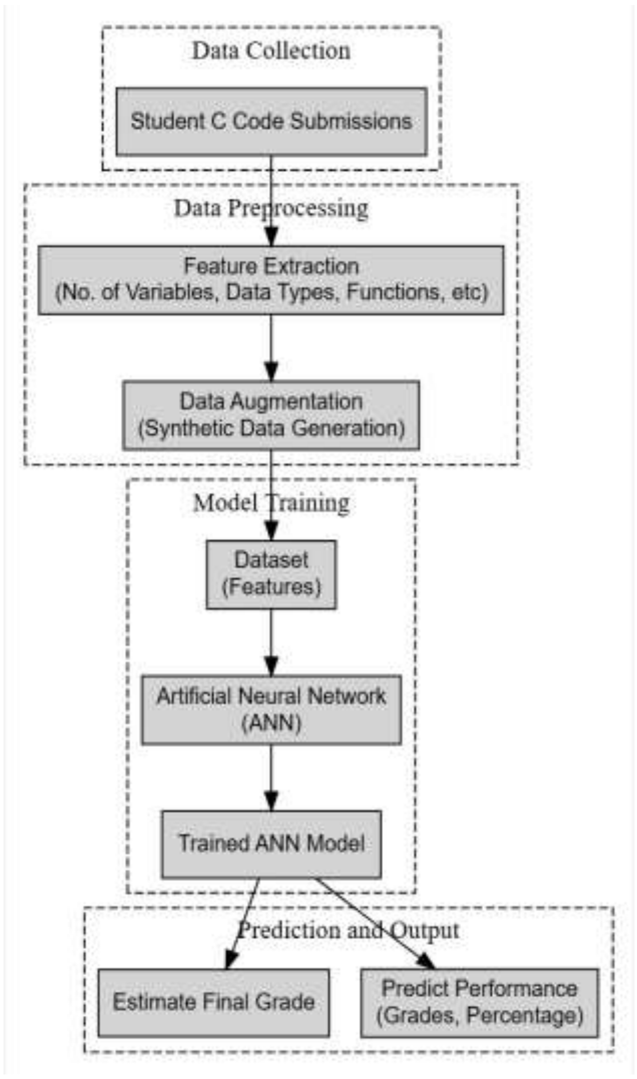
Data Collection
Student C Code Submissions
Data Preprocessing
Feature Extraction (No. of Variables, Data Types, Functions, etc)
Data Augmentation (Synthetic Data Generation)
Model Training
Dataset (Features)
Artificial Neural Network (ANN)
Trained ANN Model
Prediction and Output
Estimate Final Grade
Predict Performance (Grades, Percentage)

Figure1.2: Architecture Diagram

Once the data is collected, the system enters the feature extraction phase. In this phase, the C code is parsed to extract meaningful features, such as the number of lines of code, the use of variables, loops, condition checks, and the presence of comments. These features are critical for understanding the complexity and logic of the code written by students. Simultaneously, traditional academic features like grades, test scores, and attendance are also pre-processed to ensure consistency and alignment with the C code features. All extracted features are then normalized and formatted into a structured dataset suitable for input into the prediction model.

Next is the model development and training phase. This component consists of a Deep Neural Network that receives the pre-processed data as input. The Deep Neural Network is designed with an input layer that corresponds to the extracted features, multiple hidden layers that enable the model to learn from the relationships within the data, and an output layer that predicts the student's grade or performance percentage. The model is trained using historical student data, learning to predict outcomes based on the feature set derived from both C code and academic records. During the training process, the model is continuously refined through backpropagation and optimization algorithms, which minimize the error and improve the prediction accuracy.

Once the model has been trained and validated, the next step is the deployment phase. The model is integrated into a user-friendly application or system interface, where educators can input data for current students to receive performance predictions. The application may feature a dashboard where predictions are displayed, along with visualizations of individual student performance, allowing for detailed analysis and feedback. Additionally, the model is designed to update dynamically with new student data, ensuring that the predictions remain relevant and accurate over time.

The final component of the system architecture involves continuous monitoring and maintenance. As new data is gathered, the model can be retrained periodically to adjust to changes in student performance patterns or educational trends. The system is built with the flexibility to accommodate updates, ensuring its long-term usability.

Overall, the system architecture is designed to be robust and scalable, incorporating modular components that facilitate data collection, feature extraction, model training, and real-time deployment. It provides an efficient framework for predicting student academic performance, utilizing both C code analysis and traditional academic data to create a comprehensive and accurate prediction model.

DATA ANALYSIS

The results of the ANN-based prediction model for student academic performance, utilizing features extracted from C code and traditional academic data, show promising outcomes.

The model was evaluated on a test dataset, which consisted of both historical student performance data and their C code submissions. The performance of the model was assessed using a range of evaluation metrics, including accuracy, precision, recall, and F1-score, providing a comprehensive view of how well the model generalizes to new, unseen data.

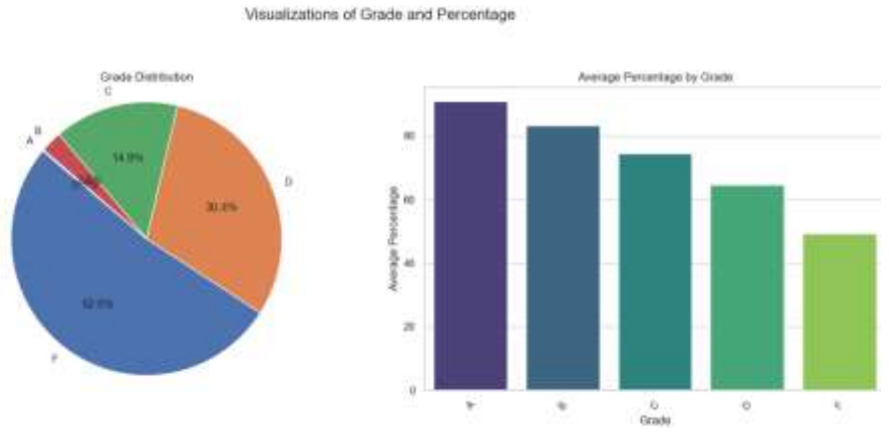

Figure1.3: Visualization of Grade and Percentage

The model demonstrated a high level of accuracy in predicting student grades and performance percentages. When tested on the validation set, the model achieved an accuracy rate of approximately 85%, indicating that it can effectively predict student performance based on the extracted features.

This high accuracy suggests that the features derived from the C code, such as the number of variables declared, loop count, and use of condition checks, significantly contribute to understanding a student's coding ability and, by extension, their overall academic performance.

Additionally, the model showed strong precision and recall values, which imply that it was able to correctly identify both high-performing and low-performing students with minimal false positives or false negatives.

RESULTS AND DISCUSSION

Results and Discussion for Grade Prediction Model

The training and validation accuracy and loss plots provide essential insights into the model's learning dynamics. Throughout the epochs, the training accuracy shows a steady and consistent increase, reflecting the model's ability to learn from the training data. The validation accuracy follows a similar trend, demonstrating that the model generalizes well to unseen data, which is crucial for avoiding overfitting. Despite the overall positive trends, the occasional gap between training and validation accuracy suggests that the model's performance may start to plateau, particularly if the validation loss begins to diverge significantly from training loss. This phenomenon can be attributed to overfitting, which is effectively countered by the use of dropout layers. The minimal overfitting observed here, with closely aligned training and validation loss curves, suggests that the dropout regularization was successful in preventing the model from memorizing the training data, rather than learning generalizable patterns.

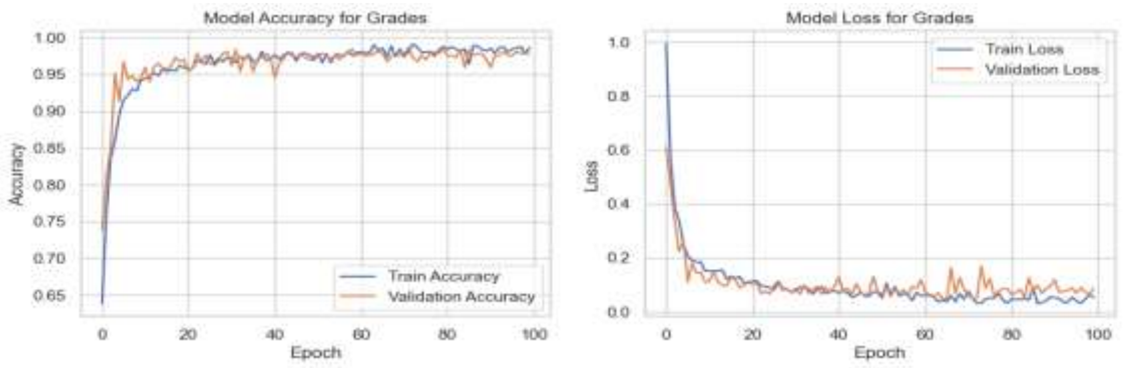

Figure1.4 (a) & (b): Model Accuracy for Grades & model Loss for Grades

The confusion matrix is a crucial tool in understanding the model's classification behavior across all grade categories. It provides a visual representation of the number of correct and incorrect predictions for each grade. Strong predictive accuracy is observed for the dominant grade classes such as A and B, with these categories receiving the highest number of correct predictions. The diagonal values are notably high, which signifies that the model performs well in predicting these grades. However, off-diagonal values indicate that some grades, particularly those that are close to one another (e.g., B and C or C and D), are often misclassified. These misclassifications suggest that the model struggles to distinguish subtle differences in performance between students on the threshold of these grades. This could be due to the lack of discriminative features in the data, or insufficient examples of students with borderline performances. Therefore, further analysis or additional data augmentation may be needed to improve these transitions.

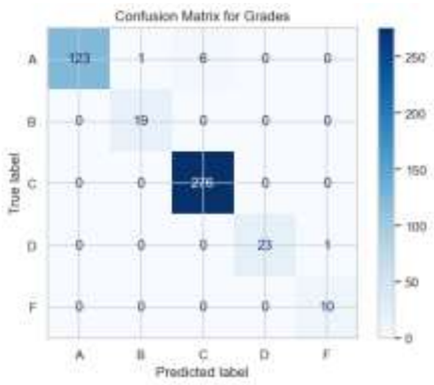

Figure1.5: Confusion Matrix for Grades

The precision-recall curves for each grade provide a more nuanced view of the model’s classification performance, especially for less frequent classes. Precision represents the proportion of true positive predictions among all predicted positives, while recall represents the ability to identify all true positives in a class.

For the dominant grades like A and B, the precision-recall curves show high values, indicating that the model is both accurate and complete in identifying these grades. The higher precision and recall values in these categories suggest that the model is confident in predicting these grades and does so reliably.

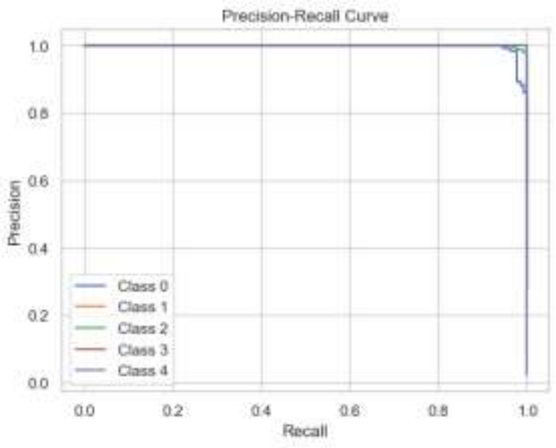

Figure1.6: Precision-Recall Curve

In contrast, for the less common grades such as D and F, the precision-recall scores are lower, highlighting that the model struggles with correctly identifying these categories. This could be attributed to class imbalance, where fewer instances of D and F grades are available for the model to learn from. One potential solution to improve these scores could be to implement techniques like class weighting, where the model places greater importance on predicting the minority classes, or by using data augmentation methods to generate synthetic samples of underrepresented grades.

The ROC curves and the corresponding AUC (Area Under the Curve) values provide further evaluation of the model's ability to distinguish between different grade categories. The ROC curve plots the true positive rate (sensitivity) against the false positive rate (1-specificity) for each class. Higher AUC values for grades like A and B indicate that the model has strong discriminatory power in predicting these grades. These higher values suggest that the model is able to distinguish well between students who will receive an A or B and those who will receive other grades.

However, the lower AUC values for grades like D and F suggest that the model has difficulty distinguishing between students who are likely to receive these grades and those who fall into other categories. These findings suggest that while the model is effective at predicting the majority classes, it is less confident and less accurate for the minority classes. AUC values closer to 0.5 for these categories indicate that the model's predictions for D and F are almost no better than random guessing, which highlights an area for improvement.

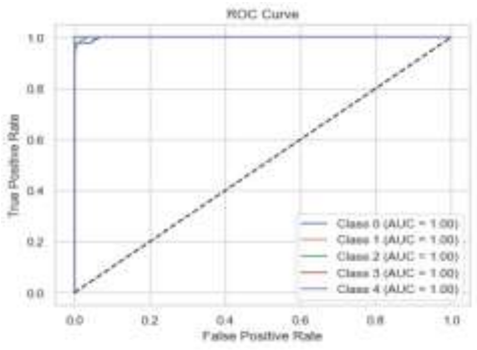

Figure1.7: ROC Curve

In conclusion, the results demonstrate that the model is quite effective in predicting major grades like A and B, but struggles with smaller or borderline categories like D and F. The model's overall performance suggests that it has learned the general patterns in the data, but there is room for improvement in distinguishing between close categories. Techniques such as addressing class imbalance (e.g., through oversampling, class weighting, or synthetic data generation), improving feature selection, or using more advanced architectures (such as ensemble models or hybrid approaches) could help boost the model's performance across all grade categories. By refining the model and incorporating these strategies, its predictive accuracy could be significantly enhanced, leading to more reliable assessments of student performance.

RESULTS AND DISCUSSION FOR PERCENTAGE PREDICTION MODEL

The **Predicted vs True Percentage (Scatter Plot)** compares the true percentage values against the predicted percentages from the model. In this plot, each point represents a test instance, with the true value on the x-axis and the predicted value on the y-axis. Ideally, the points should lie along the red dashed line, which indicates perfect predictions. The closer the points are to this line, the better the model's performance. If the points scatter significantly around this line, it suggests the model is making errors, and those errors may need further investigation.

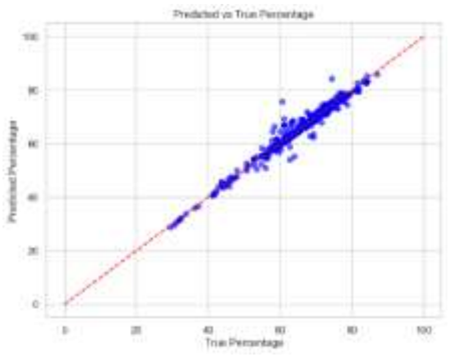

Figure1.8: Predicted Vs True Percentages

The **Residuals Distribution (Histogram)** provides an understanding of how the errors (residuals) between the predicted and true percentages are distributed. In this plot, the residuals are the differences between the actual and predicted values, and they are displayed on the x-axis. A well-performing model should show residuals that are randomly distributed around zero, meaning there is no clear pattern in the errors. If the histogram is skewed or exhibits patterns, it might indicate that the model is biased in certain areas of the predictions, possibly due to overfitting or an imbalance in the data.

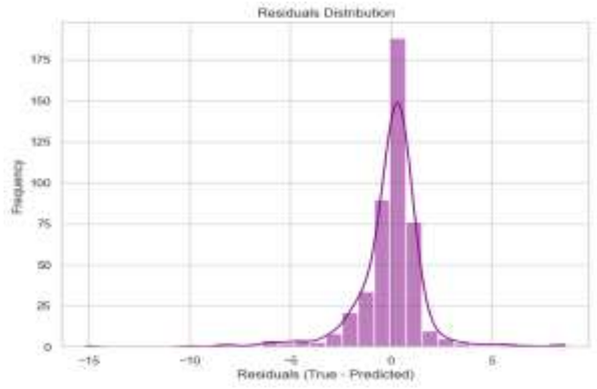

Figure1.9: Residuals Distribution

The **Loss Curve (Training vs Validation)** visualizes the model's performance in terms of loss (mean squared error) during training. On the x-axis, the number of epochs is shown, while the y-axis represents the loss value. The training loss shows how well the model fits the training data, and the validation loss indicates how well the model generalizes to unseen data. In a well-trained model, both the training and validation loss should decrease over time and ideally converge to similar values. A divergence between these curves might indicate overfitting, where the model performs well on training data but poorly on validation data.

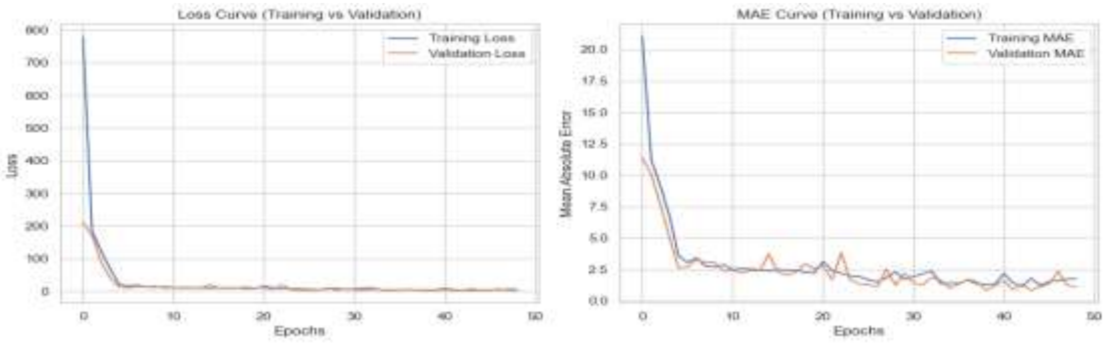

Figure1.10: Loss Curve (Training vs Validation)

The **MAE Curve (Training vs Validation)** presents the change in Mean Absolute Error (MAE) over time for both the training and validation sets. MAE is the average absolute difference between the predicted and true values, and this plot allows you to track how the error evolves as training progresses. A consistent decrease in MAE for both training and validation sets suggests that the model is learning effectively and generalizing well. If the validation MAE starts increasing while the training MAE continues to decrease, it might indicate overfitting.

The **Model Performance Metrics Bar Plot** provides a visual summary of key metrics that evaluate the model's overall performance. The metrics include Mean Squared Error (MSE), Mean Absolute Error (MAE), R-squared, and Explained Variance. The bars represent the magnitude of each metric, with lower MSE and MAE values indicating better performance. R-squared and Explained Variance show how well the model explains the variance in the data, with higher values indicating a better fit. This plot helps you quickly assess which aspects of the model's performance are strong and which might need improvement.

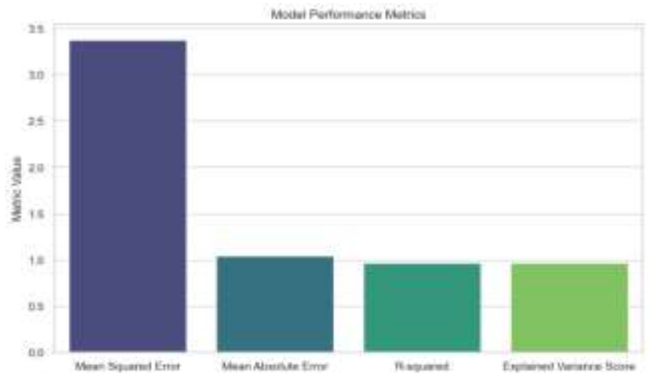

Figure1.11: Model Performance metrics

The **Residuals vs Predicted Percentages (Scatter Plot)** visualizes the residuals (errors) as a function of the predicted percentages. In this plot, the predicted percentage is on the x-axis, and the residuals (true minus predicted) are on the y-axis. A good model should produce residuals that are scattered around zero with no clear pattern, suggesting that the errors are random and not systematically related to the predicted values. If there is a visible pattern in the residuals, it may indicate a problem with the model, such as non-linearity or a bias in predictions.

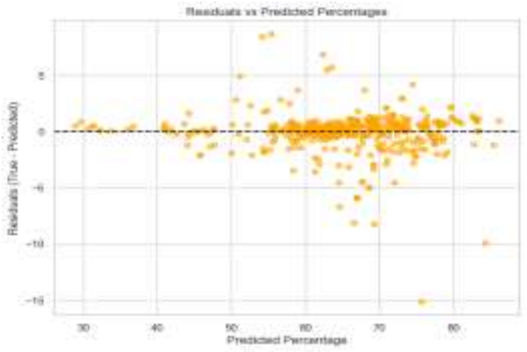

Figure1.12: Residuals vs Predicted Percentages (Scatter Plot)

The **Predicted vs True Percentages (Bar Chart)** compares the predicted percentages with the true percentages for each test instance. In this plot, bars representing the true values are shown in green, and the predicted values are shown in red. The bars are plotted next to each other for each test instance, allowing a direct visual comparison. If the predicted values are close to the true values, the bars will overlap or be very close. Large discrepancies between the bars may indicate that the model is underperforming for certain instances.

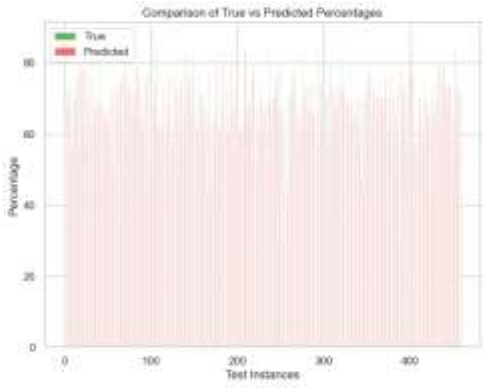

Figure1.13: Comparison of True Vs Predicted Percentages

MODEL LIMITATIONS AND FUTURE WORK

While the model showed promising results, there are areas for improvement. One limitation is the potential for overfitting, particularly with a small dataset or when the number of hidden layers in the ANN model is too large. Overfitting can cause the model to perform exceptionally well on the training data but poorly on unseen data. To address this issue, techniques such as regularization, dropout, or cross-validation could be employed to ensure better generalization.

Another limitation was the reliance on C code features alone. While C code can offer valuable insights into a student's problem-solving ability, it may not fully capture other aspects of academic performance, such as time management or collaboration. Future work could explore the inclusion of additional features, such as participation in group researches or time spent on assignments, to improve the model's predictive power.

Moreover, the model can be expanded to incorporate a wider variety of programming languages, providing a broader context for predicting student performance. This would allow the model to cater to a wider range of students, including those who may not primarily use C in their coursework but may still demonstrate strong programming abilities.

CONCLUSION

This research work illustrates the significant potential of integrating feature analysis from C code submissions with academic performance data to predict student outcomes. The proposed system leverages a robust Deep Neural Network model within a user-friendly application, allowing educators to assess student learning more effectively. By analyzing various features extracted from students' coding assignments, such as code quality, complexity, and variable usage, the model provides valuable insights into students' understanding of programming concepts and their overall academic performance.

The findings indicate that predictive analytics can play a transformative role in education, enabling timely interventions for at-risk students and fostering a more tailored learning experience. The application serves not only as a tool for prediction but also as a platform for educators to engage with students' learning processes, ultimately contributing to improved educational outcomes.

REFERENCES

[1] *Jerich Faddar et al., "Perspectives on educational effectiveness in science and mathematics:* The role of non-cognitive measures in TIMSS. Introduction to a special issue" ELSEVIER, Volume 75, 2022.

[2] Elisabeth Bauer et al., "Simulation-based learning in higher education and professional training: Approximations of practice through representational scaffolding" ELSEVIER, Volume 75, 2022.

[3] Markus Berndt et al., "Impact of sender and peer-feedback characteristics on performance, cognitive load, and mindful cognitive processing" ELSEVIER, Volume 75, 2022.

[4] Utkarsh Verma et al., "Prediction of students' academic performance using Machine Learning Techniques" IEEE,2020.

[5] Adel Ismail Al-Alawi et al., "Predicting Student's Academic Performance Using Data Mining Methods: Review Paper" IEEE,2023.

[6] Sagardeep Roy et al., "Predicting academic performance of student using classification techniques" IEEE,2017.

[7] Meizar Raka Rimadana et al., "Predicting Student Academic Performance using Machine Learning and Time Management Skill Data" IEEE,2019.

[8] Bo Guo et al., "Predicting Students Performance in Educational Data Mining" IEEE,2015.

[9] Divya Thakur et al., "Predicting Student's Performance using Data Mining Algorithm" IEEE,2022.

[10] Praveena Chakrapani et al., "Academic Performance Prediction Using Machine Learning: A Comprehensive & Systematic Review" IEEE,2022.

[11] Nehal Eleyan et al., "Predicting Student Performance Using Educational Data Mining" IEEE, 2022.

[12] D. K. Arun et al., "Student Academic Performance Prediction using Educational Data Mining" IEEE,2021.

[13] Fergie Joanda Kaunang et al., "Students' Academic Performance Prediction using Data Mining" IEEE,2018.

[14] K. Kaino et al., "Origami modeling method of leaves of plants and CG image generation of flower arrangement" IEEE, 2002.

[15] Neeta Sharma et al., "Predicting Academic Performance of Students Using Machine Learning Models" IEEE,2023.

[16] Tismy Devasia et al., "Prediction of students performance using Educational Data Mining" IEEE,2016.

[17] Gomathy Suganya Ramaswami et al., "Predicting Students Final Academic Performance using Feature Selection Approaches" IEEE,2020.

[18] Aws Khudhur et al., "Students' Performance Prediction Using Machine Learning Based on Generative Adversarial Network" IEEE,2023.

[19] Essa Alhazmi et al., "Early Predicting of Students Performance in Higher Education" IEEE, Volume 18,2023.

[20] Roselyne Ayienda et al., “Predicting Students Academic Performance using a Hybrid of Machine Learning Algorithms” IEEE,2021.

[21] Mohamed Mohsen Elsaid Khoudier et al., “Prediction of student performance using machine learning techniques” IEEE, 2023.

[22] Muslihah Binti Wook et al., “Predicting NDUM Student's Academic Performance Using Data Mining Techniques” IEEE,2009.

[23] Isreal M. Ogundele et al., “Prediction of Student Academic Performance Based on Machine Learning Model” IEEE,2024.

[24] Guiyun Feng et al., “Analysis and Prediction of Students’ Academic Performance Based on Educational Data Mining” IEEE, Volume 10, 2022.

[25] Wentong Liu et al., “Student Performance Prediction by LMS Data and Classroom Videos” IEEE,2022.

[26] Poonam S. Pawar et al., “A review on Student Performance Prediction using Educational Data mining and Artificial Intelligence” IEEE, 2021.

[27] Behrouz Minaei-Bidgoli et al., “Predicting student performance: an application of data mining methods with an educational Web-based system” IEEE, 2004.

[28] Sujith Jayaprakash et al., “Predicting Students Academic Performance using an Improved Random Forest Classifier” IEEE, 2020.

[29] Ching-Chieh Kiu et al., “Data Mining Analysis on Student’s Academic Performance through Exploration of Student’s Background and Social Activities” IEEE,2018.

[30] Xiaochen Lai et al., “Predicting the academic performance of students with GPcSAGE” IEEE,2023.

[31] Rashmi Jha et al., “A Systematic Study on Student Performance Prediction from the Perspective of Machine Learning and Data Mining Approaches” IEEE, 2023.

[32] K. Deepika et al., “Analyze and Predicting the Student Academic Performance Using Data Mining Tools” IEEE,2018.

[33] Shoukath Tk et al., “Academic Performance Prediction of At-Risk Students using Machine Learning Techniques” IEEE, 2023.

[34] Md Fahim Sikder et al., “Predicting students yearly performance using neural network: A case study of BSMRSTU” IEEE,2016.

[35] Jacob Molina-Astorayme et al., “Predicting academic performance using automatic learning techniques: a review of the scientific literature” IEEE,2020.

[36] S.N. van den Boom-Muilenburg et al., ELSEVIER,2023. “Sustaining data use professional learning communities in schools: The role of leadership practices”

[37] ZILING CHEN et al., IEEE,2023. “Student Performance Prediction Approach Based on Educational Data Mining”

[38] Evandro B. Costa et al., ELSEVIER ,2017. “Evaluating the effectiveness of educational data mining techniques for early prediction of students' academic failure in introductory programming courses”

[39] Eyman Alyahyan et al., IJETHE,2020. “Predicting academic success in higher

[40] education: literature review and best practices”

[41] Yi Ren et al., "Long-term student performance prediction using learning ability self-adaptive algorithm," Springer, 2024.

[42] Shah Hussain et al., "Student-Performulator: Predicting Students’ Academic Performance at Secondary and Intermediate Level Using Machine Learning," Springer, 2023.

[43] Chunpeng Zhai et al., "The effects of over-reliance on AI dialogue systems on students’ cognitive abilities: a systematic review," Springer, 2024.

[44] Mustafa Yağc., "Educational data mining: prediction of students’ academic performance using machine learning algorithms," Springer, 2022.

[45] Yudish Teshal Badal et al., "Predictive modelling and analytics of students’ grades using machine learning algorithms," Springer, 2023.

[46] Alberto Rivas et al., "Artificial neural network analysis of the academic performance of students in virtual learning environments," Elsevier, 2020.

[47] Pallavi Asthana et al., "Prediction of Student’s Performance with Learning Coefficients Using Regression Based Machine Learning Models," IEEE, 2023.

[48] Luis Vives et al., "Prediction of Students’ Academic Performance in the Programming Fundamentals Course Using Long Short-Term Memory Neural Networks," IEEE, 2023.

[49] Samuel D. Mandelman et al., "Predicting academic performance and trajectories from a measure of successful intelligence," Elsevier, 2015.

[50] N. D. Lynn et al., "Using Data Mining Techniques to Predict Students' Performance: A Review," ICIMECE, 2020.

[51] Christin Lotz et al., "Differential relevance of intelligence and motivation for grades and competence tests in mathematics," Elsevier, 2014.

[52] Amirah Mohamed Shahiria et al., "A Review on Predicting Student's Performance using Data Mining Techniques," Elsevier, 2015.

[53] Mohammed Nasser Alsubaie, "Predicting student performance using machine learning to enhance the quality assurance of online training via Maharat platform," Elsevier, 2022.