Impact of Optimizer on the MLP-Based Models for Student Performance Classification

Fairul Nazmie Osman, Mohd Azri Abdul Aziz*, Ihsan Mohd Yassin and Mohd Nasir Taib

Abstract— Effectively predicting student academic performance is a critical challenge in engineering education, where enhancing the performance and generalization of machine learning models can significantly aid early intervention strategies, which are crucial for engineering students as they help identify and support those at risk of falling behind, ensuring better academic outcomes and retention in the challenging field. This study investigates the impact of different optimization algorithms on Multi-Layer Perceptron (MLP) models for student performance forecasting, utilizing a dataset of 99 student samples. Recursive Feature Elimination (RFE) was employed to select the most salient features, thereby reducing model complexity. Five optimizers, AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum were evaluated to assess their influence on convergence speed, stability, and generalization. Performance was gauged by the number of epochs for convergence and key metrics including accuracy, precision, recall, and F1-score. AdamW and Nadam demonstrated superior overall performance, converging rapidly with stable results. AdamW achieved the highest F1-score (86.95%), while both AdamW and Nadam attained the highest testing accuracy (80.0%). Conversely, SGD with Momentum underperformed, exhibiting signs of underfitting with the lowest accuracy (55.0%) and F1-score (47.05%). By combining RFE with a careful selection of adaptive optimizers, this research underscores a robust methodology for developing MLP models capable of effectively analyzing educational data. These findings highlight the balance between learning efficiency and predictive reliability, supporting data-driven decision-making in education. Future research will focus on validating these findings on larger datasets and exploring the impact of optimizer choice on fairness metrics in educational predictions.

***Index Terms*— Academic performance forecasting, engineering education, Multi-Layer Perceptron (MLP), Recursive Feature Elimination (RPE).**

I. INTRODUCTION

Education is the cornerstone of societal progress, encompassing the acquisition of knowledge, skills, values, and beliefs essential for individual growth and collective development. It fosters critical thinking, problem-solving abilities, and adaptability, preparing individuals to contribute meaningfully to society and navigate an increasingly complex world. This process spans various levels, from primary schooling to higher education and lifelong learning, aiming to cultivate well-rounded individuals capable of innovation and informed decision-making.

Engineering education specifically focuses on applying scientific and mathematical principles to design, build, and maintain structures, machines, systems, and processes. It requires rigorous training in analytical skills, technical expertise, and practical application, emphasizing creativity and ethical considerations. Engineering education prepares individuals to tackle complex challenges in various fields, contributing to technological advancements and infrastructure development that underpin modern society.

Student forecasting plays a vital role in optimizing educational outcomes by predicting academic performance and identifying students at risk of falling behind. Utilizing machine learning models, data analytics, and various assessment metrics, forecasting enables educators to intervene early, provide personalized support, and enhance learning strategies. By predicting student outcomes, educational institutions can make data-driven decisions, tailor resources effectively, and improve overall academic retention and success, particularly in demanding fields like engineering education.

Machine learning (ML) has substantially advanced the capabilities of classification models, particularly in the domain of academic performance prediction. Artificial Neural Networks (ANN), and specifically Multilayer Perceptrons (MLP), are extensively utilized due to their proficiency in learning complex, non-linear patterns inherent in student data [1], [2]. However, achieving optimal model performance is contingent upon several factors, including meticulous feature selection, the choice of optimization algorithm, and appropriate evaluation metrics.

The efficacy of MLP classifiers is profoundly influenced by the optimization algorithm used during training, which dictates convergence speed, stability during training, and the model's ability to generalize to unseen data. While various optimizers exist, a comparative analysis of their performance, particularly newer adaptive optimizers like AdamW and Nadam, in conjunction with RFE for student academic performance prediction using MLP, has not been extensively explored, especially concerning convergence stability and generalization on smaller, focused datasets.

This study aims to fill this gap by systematically evaluating five prominent optimization algorithms: AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum. These optimizers were selected to represent different families of

This manuscript is submitted on 4th June 2025, revised on 3rd July 2025, accepted on 4th July 2025 and published on 31st October 2025.

Fairul Nazmie Osman, Mohd Azri Abdul Aziz, Ihsan Mohd Yassin, and Mohd Nasir Taib are from Faculty of Electrical Engineering, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia.

*Corresponding author
Email address: azriaziz@uitm.edu.my

optimization strategies, including those with adaptive learning rates (AdamW, AdaGrad, AmsGrad, Nadam) and momentum-based approaches (SGD with Momentum), some of which are newer and whose specific impact in this educational contextwarrants investigation. We analyze their impact on convergence speed and stability to determine the most effective optimization strategy for MLP models trained on RFE-selected features. The subsequent sections will review existing literature on classification models, feature selection, optimization techniques, and evaluation metrics, leading to the methodology employed for assessing optimizer performance and a discussion of the findings. Additionally, Recursive Feature Elimination (RFE) was employed to enhance model efficiency and predictive power by identifying and selecting the most relevant features, thereby improving accuracy while mitigating computational overhead [3], [4].

The remainder of this paper is organized as follows. Section I, presents an overview of related works, followed by the methodology (Section II). Next, results and discussions are presented in Section III. Finally, concluding remarks and future works are presented in Section IV.

A. Related Work

The development of effective predictive models for student academic performance hinges on a synergistic interplay of the classification algorithm, feature selection methodology, optimization strategy, and rigorous performance evaluation. MLPs have gained prominence in this area due to their robustness and capacity for complex data modeling [4], [5]. However, feature selection remains a critical precursor for enhancing model efficiency, with RFE recognized as an effective method for dimensionality reduction [4], [6]. Concurrently, the choice of optimization algorithm significantly shapes the learning dynamics of MLP models. Evaluating their impact systematically is therefore essential.

B. ANN and MLP for classifier

The Multilayer Perceptron, a specific architecture of Artificial Neural Networks, has emerged as a significant methodology in classification tasks due to its ability to model complex, non-linear relationships within data. MLPs consist of multiple layers of input, hidden, and output, facilitating better feature learning and generalization across varied applications, including classifier and prediction model [7], [8]. Multiple research projects have shown the effectiveness of MLPs in multidiscipline field, showcasing their adaptability and efficiency in processing non-linear datasets.

Balancing the trade-offs between model complexity, training efficiency, and accurate performance continues to be a pivotal area of investigation, aimed at harnessing the full potential of MLPs in real-world applications [9], [10], [11].

C. RFE Feature Selection & Optimizer Algorithm

Recursive Feature Elimination has emerged as a powerful feature selection technique in machine learning due to its ability to improve model performance by systematically removing the least important features, thus streamlining the modeling process. This methodology not only enhances computational efficiency but also reduces overfitting, which is crucial in high-dimensional datasets [12], [13]. RFE relies on a classifier to rank features based on their importance, allowing researchers to identify and retain the most significant variables while discarding irrelevant ones [14]. Such iterative feature selection has shown beneficial results across various applications, including medical imaging and bioinformatics, where clarity of data is paramount for accurate predictions [15].

In parallel with advanced feature selection methods like RFE, the choice of optimization algorithms is fundamental in enhancing the convergence of training processes for machine learning models. Optimizers like AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum each have unique advantages that cater to different aspects of optimization. For instance, AdamW and Nadam are known for their effective handling of sparse gradients and adaptive learning rates, which are beneficial in tasks involving large datasets with varied feature scales [16], [17]. Meanwhile, AdaGrad excels in adapting the learning rate to the parameters, ensuring that infrequently updated features receive larger updates, thus promoting better learning in scenarios with numerous features [18]. The AmsGrad variant addresses some limitations of the original Adam optimizer, providing more stable convergence properties in certain scenarios [19].

The combination of RFE for feature selection and robust optimizers is pivotal in achieving high-performance models. While RFE focuses on refining input variables, the optimizers facilitate efficient learning from the selected features, ensuring that the models are not only precise but also practical for real-world applications [19]. By leveraging these techniques, researchers can improve model interpretability and generalization, which are fundamental for tasks like disease diagnosis and environmental monitoring where understanding the underlying features is as critical as predictive accuracy[11].

D. Evaluation Metrics for Predictive Model performance evaluation

When evaluating predictive models, particularly those MLP architectures, performance metrics such as accuracy, precision, recall, and F1-score play a crucial role. These metrics provide essential insights, especially in scenarios involving a wide range of class distributions, which are common in real-world applications [8]. Accuracy, defined as the proportion of correctly classified instances out of the total instances, offers a general performance assessment; however, relying solely on accuracy can be misleading in cases of class imbalance, as it may fail to reflect the model's effectiveness in predicting minority class instances [10].

To gain a deeper understanding of model performance, precision and recall are often analyzed. Precision measures the proportion of true positives among all predicted positives, indicating the reliability of positive class predictions [10], [20]. In contrast, recall (sensitivity) assesses the model's ability to correctly identify all relevant instances, expressed as the ratio of true positives to the total number of actual positives. High recall is particularly important in applications where identifying all positive cases is prioritized, even at the expense of precision,

such as in medical diagnostics or fraud detection [21]. This emphasizes the need for a careful balance between precision and recall, contributing to improved model effectiveness by ensuring that both false positives and false negatives are carefully managed [9].

The F1-score serves as a comprehensive performance metric by combining precision and recall, making it especially useful for imbalanced classification problems [22]. It is particularly valuable in cases where the costs of false positives and false negatives differ, offering a more balanced evaluation compared to accuracy alone. For example, in various application domains such as medical diagnostics and fraud detection, researchers often emphasize the importance of the F1-score alongside accuracy and recall providing a holistic measure of classification performance [23]. The trade-offs between precision and recall become a focal point for maximizing model performance, often requiring that evaluations prioritize these metrics to achieve the best practical outcomes in sensitive applications [24].

E. Convergence Speed and Stability of Optimizers

The effectiveness of a MLP classifier often hinges on the choice of optimization algorithm, impacting both convergence speed and training stability [25]. Convergence speed, defined as the number of epochs required to achieve an optimal solution, and stability, pertaining to the smoothness of loss reduction, are crucial for efficient learning progression. Momentum-based optimizers like SGD with Momentum accelerate training [26], while adaptive optimizers such as AdamW, AdaGrad, AmsGrad, and Nadam dynamically adjust learning rates to enhance stability throughout the training process [27], [28]. Although faster convergence can reduce computational costs, it may also lead overfitting, particularly if not properly modelled [29]. Conversely, slow-converging optimizers like AdaGrad can struggle due to diminishing learning rates, often requiring more epochs for effective performance[30]. Consequently, evaluating optimizer performance should encompass an analysis of learning progression, model accuracy, precision, recall, and F1-score, ensuring generalization to unseen data [31]. In predictive modeling for education, where reliability is critical, selecting an optimizer that balances convergence speed and stability is essential, as this choice directly influences the overall success of MLP classifiers [32].

F. AI in Engineering Education: Tools, Methodologies, Outcomes, and Challenges

The integration of Artificial Intelligence (AI) into engineering education has emerged as a transformative force, offering innovative tools and methodologies to enhance student performance, personalize learning experiences, and prepare students for the demands of Industry 5.0. This section explores the role of AI in predicting student performance, the technical methodologies employed, the outcomes and effectiveness of these tools, and the challenges and ethical considerations associated with their implementation.

G. AI Tools for Student Performance Prediction

AI tools have been widely adopted in engineering education to predict student academic performance, identify at-risk students, and provide personalized learning pathways. These tools leverage machine learning algorithms, data mining, and natural language processing to analyze student data, such as learning processes, participation rates, and summative performance, to predict academic outcomes [1], [2]. For instance, evolutionary computation techniques have been used to develop prediction models that identify dominant variables influencing academic performance, such as knowledge acquisition and class participation, while downplaying the role of prerequisite knowledge [1]. Similarly, artificial neural networks (ANNs) with techniques like the Levenberg–Marquardt algorithm have been shown to outperform traditional machine learning methods in predicting student grades, achieving higher accuracy rates [33].

1) Technical Methodologies

The technical methodologies underpinning AI tools in engineering education are diverse and sophisticated. Machine learning algorithms, such as those used in adaptive learning systems, enable real-time feedback and early detection of learning difficulties [5]. Data analytics and intelligent tutoring systems are employed to tailor educational content and optimize curriculum design, ensuring that students receive personalized learning experiences [3], [34]. Additionally, AI-driven tools such as chatbots, virtual tutors, and interactive simulations are increasingly being integrated into engineering education to support second language learners (L2) by providing personalized feedback and interactive learning opportunities [4].

2) Outcomes and Effectiveness

The integration of AI tools into engineering education has yielded promising outcomes, enhancing student engagement, academic performance, and overall learning experiences. Studies have shown that AI-enhanced learning environments can improve student satisfaction and collaborative learning performances, particularly in online engineering courses [2]. Moreover, AI tools have been effective in boosting language skills, critical thinking, and problem-solving abilities among L2 learners, resulting in superior learning outcomes [4]. The use of AI in predicting student performance has also enabled early interventions, helping students at risk of dropping out to complete their studies and enhance their future competitiveness [6].

3) Challenges and Ethical Considerations

Despite the potential of AI in engineering education, several challenges and ethical considerations must be addressed. Data privacy and algorithmic bias are significant concerns, as AI systems rely on vast amounts of student data, which must be protected from breaches and misuse [35] . Additionally, the over-reliance on AI tools can lead to the assimilation of unverified knowledge, emphasizing the need for students to critically evaluate AI-generated content [36], [37]. Ethical frameworks are essential to ensure that AI integration in engineering education is transparent, fair, and aligned with

societal values [35].

H. Remarks

The integration of AI into engineering education offers significant opportunities for enhancing student performance, personalizing learning experiences, and preparing students for the demands of Industry 5.0. However, the implementation of AI tools must be accompanied by careful consideration of ethical issues, data privacy, and the need for human oversight to ensure that AI technologies are used judiciously and effectively. By addressing these challenges and leveraging the potential of AI, engineering education can continue to evolve and provide students with the skills and knowledge required to succeed in an increasingly complex and technology-driven world.

II. METHODOLOGY

This study evaluates the impact of different optimizers on MLP training, focusing on convergence speed, stability, and generalization. The methodology, depicted in Fig. 1, includes data collection, preprocessing, model training with various optimizers, rigorous evaluation, and comparative analysis.

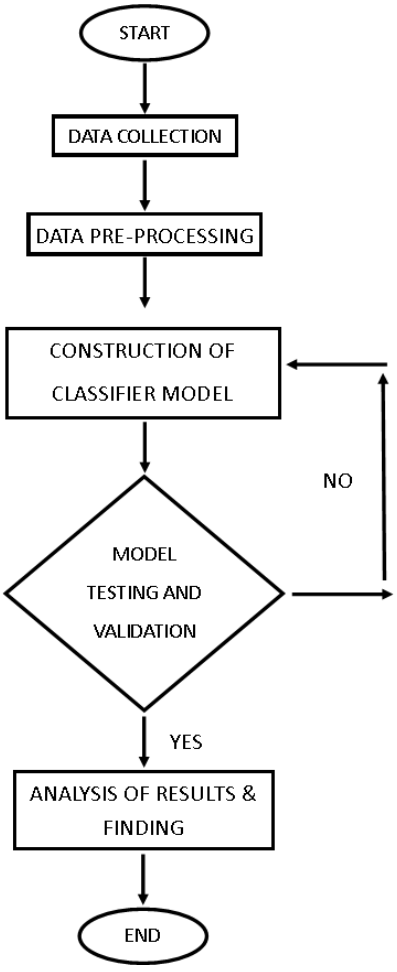

Fig 1. Research Methodology Flowchart

A. Data Set

The dataset utilized comprises 99 student samples, each characterized by 12 features reflecting academic engagement and learning behaviors, with a binary output representing academic performance. Features include access to course materials, note-taking habits, and tutorial participation (detailed in Table I).

Given the limited sample size (N=99), this study should be considered exploratory. The small dataset poses challenges for training data-intensive models like MLPs, potentially impacting the stability and generalizability of the findings. This limitation was a constraint of data availability from a specific institutional context.

TABLE I. 12 FEATURES WITH THE DESCRIPTION

Feature Number	Feature Description
Feature 1	Access to Lecture Slide and Additional Notes Before Lecture
Feature 2	Access to Lecture Slide and Additional Notes During Lecture
Feature 3	Access to Lecture Slide and Additional Notes After Lecture
Feature 4	Access to Lecture Slide and Additional Notes Outside Lecture
Feature 5	Note Length
Feature 6	Exercise Before Lecture
Feature 7	Exercise During Lecture
Feature 8	Exercise After Lecture
Feature 9	Exercise Outside Lecture
Feature 10	Tutorial Correct 3 and above
Feature 11	Tutorial Answer All Questions
Feature 12	Tutorial Wrong Before Correct

Recursive Feature Elimination (RFE) was employed for feature selection. RFE iteratively trains the model and removes the least important features, aiming to find an optimal subset that maximizes predictive accuracy while reducing overfitting risk. In this study, RFE reduced the initial 12 features to a subset of six: Feature1 (Access to Lecture Slide and Additional Notes Before Lecture), Feature3 (Access to Lecture Slide and Additional Notes After Lecture), Feature7 (Exercise During Lecture), Feature8 (Exercise After Lecture), Feature10 (Tutorial Correct 3 and above), and Feature12 (Tutorial Wrong Before Correct). These features intuitively represent proactive engagement (Feature1), post-lecture review (Feature3), active learning during and after lectures (Feature7, Feature8), and mastery in tutorials (Feature10, Feature12), suggesting their salience in predicting academic performance. This reduction enhances computational efficiency and aims to improve model

generalization by focusing on the most informative inputs.

B. MLP Model and Optimizer Configuration

A standard MLP architecture was used. The model was trained separately with five different optimizers: AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum, using their standard recommended hyperparameters unless otherwise specified. The maximum number of epochs was set to 1,000. Early stopping with a patience of 10 epochs (monitoring validation loss) was implemented to prevent overfitting and reduce unnecessary computation. The batch size was set to 32.

C. Confusion Matrix & MLP Performance Measurement.

Model performance was assessed using a confusion matrix (Table II) to derive accuracy, precision, recall, and F1-score.

TABLE II. A 2X2 CONFUSION MATRIX

Actual			
0	TN	FP	
1	FN	TP	
	0	1	Predicted

The confusion matrix as shown in Table II provides a detailed evaluation of classification performance, serving as a foundation for the next stage of experimental data analysis in MLP performance measurement. The confusion matrix metrics, including True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN), are presented in Table II. These values are essential in computing key performance indicators such as accuracy, precision, recall, and F1-score, which provide insight into the effectiveness of each optimization algorithm in distinguishing between different classes.

The accuracy of a classifier is determined by the proportion of correctly classified instances relative to the total instances, as in (1):

$$Accuracy = \frac{(TP + TN)}{TP + TN + FP + FN} \quad (1)$$

Precision measures the proportion of correctly predicted positive samples out of all predicted positive samples, reflecting the classifier's reliability in correctly identifying positive instances as in (2):

$$Precision = \frac{TP}{(TP + FP)} \quad (2)$$

Recall evaluates the model's ability to correctly identify actual positive instances, representing the proportion of positive samples that the model successfully identifies as in (3):

$$Recall = \frac{TP}{(TP + FN)} \quad (3)$$

The F1-score is a comprehensive evaluation metric that balances precision and recall, ensuring the model performs effectively across both positive and negative classifications as in (4):

$$F1 = \frac{2\,TP}{2TP + FP + FN} \quad (4)$$

Convergence speed was measured by the number of epochs required for each optimizer to reach a stable solution (based on early stopping criteria) across 20 independent runs. The performance metrics (Accuracy, Precision, Recall, F1-Score) reported in Tables III and IV were derived from a train-validation-test split (e.g. 70% train, 15% validation,15% test). For more robust estimates, especially with a small dataset, k-fold cross-validation (e.g., 5-fold or 10-fold) repeated multiple times would be advisable for future work or to strengthen these findings.

D. Epochs

The number of epochs plays a crucial role in training deep learning models, as it determines how many times the model iterates over the training data to update its parameters. In this study, the maximum number of epochs was set to 1000, with an early stopping mechanism employed to prevent unnecessary computations and mitigate overfitting. Early stopping was configured with a patience of 10 epochs, meaning training would halt if the validation loss did not improve over 10 consecutive epochs. This strategy ensures efficient training while avoiding excessive cycles that may result in diminishing performance returns. A batch size of 32 was selected, balancing computational efficiency with stable gradient updates to facilitate effective learning. The study evaluates how different optimization algorithms influence epoch convergence speed, which is a key metric for assessing optimizer efficiency and training stability.

To investigate the impact of optimization strategies on Multi-Layer Perceptron (MLP) training dynamics, this study analyzes the number of epochs required for convergence when using features selected by Recursive Feature Elimination (RFE). Optimizers that converge in fewer epochs are considered more computationally efficient, whereas those requiring a higher number of epochs may suggest slower learning or unstable updates. Each optimizer's performance was assessed over 20 independent runs to ensure robustness and account for variability across different training iterations. With the methodology established, including RFE-based feature selection, optimization strategy evaluation, and convergence analysis through epoch measurements are presented in the subsequent section presents results and discussion. It offers a comparative analysis of optimizer performance in terms of convergence speed, stability, and generalization, providing insights into selecting the most effective optimizer for predictive modelling tasks.

III. RESULTS AND DISCUSSION

This section presents the performance evaluation of the five optimizers, analyzing training/testing metrics, convergence speed, and stability. The discussion considers the bias-variance tradeoff, model generalization, and the impact of RFE in the

context of a small dataset.

A. Bias-Variance Tradeoff and Model Generalization

Table III and Table IV present the training and testing performance metrics, respectively, for five optimization algorithms: AdamW, AdaGrad, AmsGrad, Nadam, and SGD with Momentum. The accuracy drops observed across different optimizers indicate varying degrees of overfitting and underfitting tendencies. Specifically, AdamW, AmsGrad, and SGD with Momentum exhibited significant decreases in accuracy between training and testing which are 17.1%, 18.0%, and 16.2%, respectively, signaling potential overfitting. In contrast, AdaGrad and Nadam showed smaller reductions of only 6.9% and 11.3%, respectively, suggesting better generalization capabilities. Recall values also displayed considerable fluctuation: while AdamW maintained a perfect recall of 100% in both training and testing, SGD with Momentum suffered a notable decline from 50.0% during training to 44.44% in testing, highlighting high bias and an inability to capture the data distribution effectively.

A deeper analysis reveals that AdamW and AmsGrad exhibit high-variance behavior, characterized by strong training performance but notable deterioration during testing. AdamW achieved 97.1% accuracy, 94.1% precision, and 100% recall during training, but dropped to 80.0% accuracy, 76.9% precision, and maintained 100% recall during testing. Similarly, AmsGrad attained 91.3% accuracy and 96.4% precision in training, followed by a decrease to 73.3% accuracy and 75.0% precision in testing. These patterns are indicative of overfitting, where the models memorize the training data at the expense of broader generalization. Conversely, SGD with Momentum demonstrates underfitting symptoms, achieving low training accuracy (71.2%) and precision (82.4%), and deteriorating further to 55.0% accuracy and 50.0% precision during testing. Its recall also fell from 50.0% to 44.44%, signaling its limited ability to learn complex relationships within the data.

In contrast, AdaGrad and Nadam demonstrated more favorable bias-variance tradeoffs. AdaGrad achieved a moderate training accuracy of 86.9%, with only a minor reduction to 80.0% in testing, while maintaining an F1-Score of 84.2% across both datasets, indicating strong stability. Nadam sustained a high F1-Score of 89.9% in training and 85.7% in testing, with precision slightly dropping from 96.4% to 81.8%, and recall improving from 84.4% to 90.0%. These results suggest that optimizers employing adaptive learning rate strategies, such as AdaGrad and Nadam, are better equipped to manage model flexibility, thereby mitigating both excessive memorization and insufficient pattern learning. To further enhance the generalization performance, AdamW and AmsGrad would benefit from additional regularization techniques such as dropout, L2 regularization, or early stopping. For underfitting cases observed in SGD with Momentum, strategies such as dynamic learning rate adjustment, batch normalization, or increasing model complexity are recommended to improve learning capacity.

In summary, the comparative evaluation highlights that the choice of optimizer significantly influences model generalization behavior. AdamW and AmsGrad demonstrate tendencies toward high variance and overfitting, whereas SGD with Momentum exhibits high bias and underfitting. AdaGrad and Nadam achieve a better balance between bias and variance, with Nadam emerging as the most consistent and reliable performer across both training and testing scenarios. These findings reinforce the critical importance of aligning optimizer selection with the complexity of the dataset and model architecture. Effective generalization is not solely dependent on the optimizer itself but also requires complementary strategies such as regularization or architectural refinement, tailored to address either overfitting or underfitting challenges present in the learning process.

TABLE III. TRAINING PERFORMANCE METRICS OF VARIOUS OPTIMIZATION ALGORITHMS (CAPTION: TRAINING PERFORMANCE METRICS FOR MLP MODELS USING DIFFERENT OPTIMIZERS ON RFE-SELECTED FEATURES)

Optimizer	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
AdamW	97.1	94.1	100	96.7
AdaGrad	86.9	96.0	75.0	84.2
AmsGrad	91.3	96.4	84.4	89.9
Nadam	91.3	96.4	84.4	89.9
SDG with Momentum	71.2	82.4	50.0	62.2

TABLE IV. TESTING PERFORMANCE METRICS OF VARIOUS OPTIMIZATION ALGORITHMS (CAPTION: TESTING PERFORMANCE METRICS FOR MLP MODELS USING DIFFERENT OPTIMIZERS ON RFE-SELECTED FEATURES. VALUES REPRESENT PERFORMANCE ON THE UNSEEN TEST SET.)

Optimizer	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
AdamW	80.0	76.9	100	86.95
AdaGrad	80.0	88.9	80.0	84.2
AmsGrad	73.3	75.0	90.0	81.8
Nadam	80.0	81.8	90.0	85.7
SDG with Momentum	55.0	50.0	44.44	47.05

B. Optimizer Performance Comparison

The testing results highlight significant performance differences across the five optimizers in terms of accuracy, precision, recall, and F1-score. AdamW, AdaGrad, and Nadam achieve the highest accuracy (80.0%), while AmsGrad follows with 73.3%, and SGD with Momentum lags at 55.0%. Precision varies significantly, with AdaGrad achieving the highest (88.9%) and SGD with Momentum the lowest (50.0%), suggesting that AdaGrad maintains a strong ability to correctly classify positive instances while SGD with Momentum

struggles. Recall is highest for AdamW (100%), followed by AmsGrad and Nadam (90.0%), whereas SGD with Momentum exhibits the lowest recall (44.44%), indicating a failure to correctly capture positive cases. The F1-score follows a similar trend, with AdamW (86.95%), Nadam (85.7%), and AdaGrad (84.2%) performing the best, while SGD with Momentum remains the weakest (47.05%), reinforcing its inability to balance precision and recall effectively.

The results indicate that AdamW is highly recall-focused, capturing all positive instances (100%) but at the cost of lower precision (76.9%), leading to misclassifications. AdaGrad demonstrates the best balance between precision and recall (88.9% and 80.0%), making it the most stable in terms of classification reliability. AmsGrad performs decently but falls short of AdaGrad and Nadam due to its lower precision (75.0%) and accuracy (73.3%), suggesting slight overfitting or instability in decision-making. Nadam performs similarly to AmsGrad but with a higher F1-score (85.7%), making it a slightly better choice for consistent generalization. SGD with Momentum severely underperforms across all metrics, particularly in recall and F1-score, indicating that it struggles to learn effectively from the data and may require significant tuning or architectural improvements to be viable.

From these comparisons, AdamW is the best choice when prioritizing recall over precision, ensuring that all positive instances are identified, making it useful for scenarios where false negatives are costly. AdaGrad emerges as the most balanced optimizer, with the highest precision and stable recall, making it a reliable choice for applications requiring a trade-off between correct identification and avoiding misclassifications. Nadam is a strong competitor to AdaGrad, slightly favoring recall over precision, making it more suitable for cases where missing a positive instance is riskier than misclassifying a negative one. AmsGrad, while still viable, needs further tuning to enhance its precision for better performance. SGD with Momentum appears to be the least effective optimizer for this dataset, exhibiting weak classification performance overall, likely due to suboptimal learning dynamics, and should either be fine-tuned or replaced with a more adaptive optimizer.

C. Optimizer performance on Convergence and Stability using RFE as feature selection.

The number of epochs required for convergence varies significantly across the five selected optimizers highlighting differences in their efficiency and stability. SGD with Momentum consistently converges quickly, with epochs mostly in the 23 to 105 range, demonstrating its ability to efficiently navigate the loss landscape. However, minor variations across runs suggest sensitivity to initialization or hyperparameters. AdamW, another relatively fast optimizer, maintains a stable convergence pattern, typically requiring 11 to 158 epochs, with a few outliers where it converged faster than expected (Run 8: 11 epochs). Nadam and AmsGrad show higher variability, with Nadam fluctuating between 21 and 171 epochs and AmsGrad ranging from 11 to 145 epochs, indicating occasional instability in learning. AdaGrad, in stark contrast, is the slowest optimizer, requiring between 19 and 1000 epochs, frequently hitting the maximum limit, demonstrating its weakness in long-term convergence due to an aggressively decaying learning rate.

TABLE V. EPOCHS REQUIRED FOR CONVERGENCE ACROSS 20 RUNS FOR SELECTED OPTIMIZERS (CAPTION: EPOCHS TO CONVERGENCE (WITH EARLY STOPPING PATIENCE 10 ON VALIDATION LOSS, MAX 1000 EPOCHS) FOR EACH OPTIMIZER OVER 20 INDEPENDENT RUNS.)

Run	SDG with Momentom	Adagrad	AdamW	Nadam	AmsGrad
Run 1	66	534	100	39	11
Run 2	23	621	100	91	103
Run 3	32	1000	54	171	84
Run 4	79	386	69	122	75
Run 5	65	205	102	128	23
Run 6	45	488	158	83	100
Run 7	49	815	96	21	92
Run 8	57	1000	11	144	117
Run 9	62	1000	122	45	57
Run 10	105	410	148	77	93
Run 11	57	463	67	107	116
Run 12	26	368	158	127	80
Run 13	52	739	43	60	96
Run 14	65	282	99	117	132
Run 15	46	19	122	26	145
Run 16	69	335	71	105	76
Run 17	84	13	76	85	54
Run 18	84	1000	11	65	82
Run 19	75	966	11	98	11
Run 20	74	132	117	155	96

From these results, we can infer that momentum-based optimizers (SGD with Momentum and AdamW) perform consistently well in terms of convergence speed and stability. Momentum allows for faster movement across flat regions, reducing the time required for optimization. AdamW's weight decay component prevents excessive parameter updates, leading to smooth and controlled convergence. Nadam and AmsGrad, while adaptive, display moderate instability, possibly due to their reliance on higher-order moment estimates, which may amplify noisy gradient updates. The extreme slowness of AdaGrad confirms its well-known issue of diminishing learning rates, which can prevent the optimizer from making meaningful updates in later training stages. The fact that AdaGrad often requires 1000 epochs suggests it may be unsuitable for models trained on this dataset without modifications such as learning rate restarts.

For practical applications, AdamW and SGD with Momentum are the most reliable optimizers, as they balance fast convergence and stability without extreme fluctuations in required epochs. Nadam and AmsGrad are viable but require fine-tuning, particularly with respect to learning rates and batch sizes, to prevent oscillations that may slow or destabilize learning. AdaGrad, in its current form, is not ideal for this dataset, as it converges too slowly, making it computationally expensive and inefficient for training deep models. RFE feature selection appears to favor optimizers that balance momentum and adaptive learning rates, as seen in the strong performance of AdamW and SGD with Momentum, while penalizing optimizers that rely too heavily on learning rate decay, such as AdaGrad. Future research should explore how different learning rate schedules affect these optimizers, as techniques like cyclical learning rates or warm restarts might mitigate some of the instability seen in Nadam and AmsGrad while improving the efficiency of AdaGrad.

D. Limitations of the Study

The primary limitation of this study is the small dataset size (N=99). MLP models are typically data-hungry, and a small sample size can lead to:

i. Overfitting: Models may learn idiosyncrasies of the training data that do not generalize. This was observed with AdamW and AmsGrad.
ii. Unstable Performance Metrics: Results might be sensitive to the specific train/validation/test split. While 20 runs were used for epoch convergence, the main performance metrics in Tables III and IV were based on a single train/validation/test split. Employing k-fold cross-validation would provide more robust estimates.
iii. Limited Generalizability: Findings may not directly translate to larger, more diverse student populations.

Additionally, this study used default or standard hyperparameter settings for the optimizers. Extensive hyperparameter tuning for each optimizer might yield different relative performances. The choice of specific MLP architecture also influences results.

These limitations mean the findings should be interpreted as preliminary insights into optimizer behavior on small, RFE-processed educational datasets.

IV. CONCLUSION

This study investigated the impact of five optimization algorithms (AdamW, Nadam, AdaGrad, AmsGrad, and SGD with Momentum) on the performance of MLP-based student classification models, using RFE for feature selection on a dataset of 99 students. Key findings indicate that AdamW and Nadam offered the best balance of rapid convergence and strong predictive performance on unseen data, with AdamW achieving the highest F1-score (86.95%) and Nadam showing robust generalization (80.0% accuracy, 85.7% F1-score). AdaGrad also demonstrated stable, albeit slower, performance. Conversely, SGD with Momentum underperformed significantly, suggesting underfitting, while AdamW and AmsGrad showed signs of overfitting, a risk heightened by the small dataset.

The study highlights that even with dimensionality reduction via RFE, optimizer choice significantly influences a model's learning efficiency, stability, and generalization. For small datasets like the one used, adaptive optimizers such as Nadam and AdaGrad appear to offer better generalization than more aggressive optimizers if overfitting is a concern. However, the primary limitation remains the dataset size, which tempers the conclusiveness of these findings. Institutions with similar data characteristics might consider Nadam or AdamW for initial predictive models but must be mindful of potential overfitting and the critical need for validation on larger datasets.

Future research should prioritize validating these findings on larger, more diverse student populations. Exploring techniques like k-fold cross-validation for more robust performance estimation, systematic hyperparameter optimization for each optimizer, and investigating ensemble methods that could leverage the distinct strengths of different optimizers are crucial next steps to enhance predictive accuracy and reliability in educational applications. Further work could also explore the impact of these optimizers on fairness and bias in student outcome predictions.

ACKNOWLEDGMENT

I would like to express my deepest gratitude to the Faculty of Electrical Engineering,Universiti Teknologi MARA Shah Alam and IPSIS, the Postgraduate Office of UiTM, for their unwavering support and guidance in every aspect. Their assistance has been invaluable in bringing this publication to be materialized.

REFERENCES

[1] P. Jiao, F. Ouyang, Q. Zhang, and A. H. Alavi, "Artificial intelligence-enabled prediction model of student academic performance in online engineering education," Artif Intell Rev, vol. 55, no. 8, pp. 6321–6344, Dec. 2022, doi: 10.1007/s10462-022-10155-y.
[2] F. Ouyang, M. Wu, L. Zheng, L. Zhang, and P. Jiao, "Integration of artificial intelligence performance prediction and learning analytics to improve student learning in online engineering course," International

Journal of Educational Technology in Higher Education, vol. 20, no. 1, p. 4, Jan. 2023, doi: 10.1186/s41239-022-00372-4.

[3] M. M. A. Saeed, R. A. Saeed, Z. E. Ahmed, A. S. A. Gaid, and R. A. Mokhtar, "AI Technologies in Engineering Education," 2024, pp. 61–87. doi: 10.4018/979-8-3693-2728-9.ch003.

[4] P. Saradha et al., "Integration of AI tools in learning pedagogy for L2 Learners in Engineering Education: Impacts, Challenges and Possibilities," in 2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT), IEEE, Jun. 2024, pp. 1–6. doi: 10.1109/ICCCNT61001.2024.10726115.

[5] E. M. Slomp, D. Ropelato, C. Bonatti, and M. D. da Silva, "Adaptive Learning in Engineering Courses: How Artificial Intelligence (AI) Can Improve Academic Outcomes," in 2024 IEEE World Engineering Education Conference (EDUNINE), IEEE, Mar. 2024, pp. 1–6. doi: 10.1109/EDUNINE60625.2024.10500580.

[6] S.-C. Tsai, C.-H. Chen, Y.-T. Shiao, J.-S. Ciou, and T.-N. Wu, "Precision education with statistical learning and deep learning: a case study in Taiwan," International Journal of Educational Technology in Higher Education, vol. 17, no. 1, p. 12, Dec. 2020, doi: 10.1186/s41239-020-00186-2.

[7] A. Ahmad et al., "A Novel Application of MLP Networks in Classifying L Band Eco-Friendly Microwave Absorbers," in 2024 IEEE 14th International Conference on Control System, Computing and Engineering (ICCSCE), IEEE, Aug. 2024, pp. 294–298. doi: 10.1109/ICCSCE61582.2024.10696895.

[8] F. N. Osman, M. A. A. Aziz, and M. N. Taib, "Enhancing Students' Academic Performance Classifier using ADASYN and MLP," in 2024 IEEE 22nd Student Conference on Research and Development (SCOReD), 2024, pp. 221–226. doi: 10.1109/SCOReD64708.2024.10872712.

[9] N. V. Chawla and D. A. Davis, "Bringing Big Data to Personalized Healthcare: A Patient-Centered Framework," J Gen Intern Med, vol. 28, no. S3, pp. 660–665, Sep. 2013, doi: 10.1007/s11606-013-2455-8.

[10] M. A. A. Aziz, N. Ismail, I. M. Yassin, A. Zabidi, and M. S. A. M. Ali, "Agarwood oil quality classification using cascade-forward neural network," in 2015 IEEE 6th Control and System Graduate Research Colloquium (ICSGRC), 2015, pp. 112–115. doi: 10.1109/ICSGRC.2015.7412475.

[11] A. M. Abdu, M. M. Mokji, and U. U. Sheikh, "Machine Learning for Plant Disease Detection: An Investigative Comparison Between Support Vector Machine and Deep Learning," Iaes International Journal of Artificial Intelligence (Ij-Ai), vol. 9, no. 4, p. 670, 2020, doi: 10.11591/ijai.v9.i4.pp670-683.

[12] E. T. A. Albert, N. H. Bille, B. J. Martin, and N. M. E. Leonard, "Integrating genetic markers and adiabatic quantum machine learning to improve disease resistance-based marker assisted plant selection," Journal of Scientific Agriculture, pp. 63–76, Sep. 2023, doi: 10.25081/jsa.2023.v7.8556.

[13] G. Karami, M. Giuseppe Orlando, A. Delli Pizzi, M. Caulo, and C. Del Gratta, "Predicting Overall Survival Time in Glioblastoma Patients Using Gradient Boosting Machines Algorithm and Recursive Feature Elimination Technique," Cancers (Basel), vol. 13, no. 19, p. 4976, Oct. 2021, doi: 10.3390/cancers13194976.

[14] X. Zhang et al., "Recursive SVM feature selection and sample classification for mass-spectrometry and microarray data," BMC Bioinformatics, vol. 7, no. 1, p. 197, Dec. 2006, doi: 10.1186/1471-2105-7-197.

[15] E. M. Senan et al., "Diagnosis of Chronic Kidney Disease Using Effective Classification Algorithms and Recursive Feature Elimination Techniques," J Healthc Eng, vol. 2021, pp. 1–10, Jun. 2021, doi: 10.1155/2021/1004767.

[16] W. Nazih, A. O. Aseeri, O. Y. Atallah, and S. El-Sappagh, "Vision Transformer Model for Predicting the Severity of Diabetic Retinopathy in Fundus Photography-Based Retina Images," IEEE Access, vol. 11, pp. 117546–117561, 2023, doi: 10.1109/ACCESS.2023.3326528.

[17] A. Javed, I. Rashid, S. Tahir, S. Saeed, A. M. Almuhaideb, and K. Alissa, "AdamW+: Machine Learning Framework to Detect Domain Generation Algorithms for Malware," IEEE Access, vol. 12, pp. 79138–79150, 2024, doi: 10.1109/ACCESS.2024.3407546.

[18] R. Anil, V. Gupta, T. Koren, and Y. Singer, "Memory-Efficient Adaptive Optimization," 2019, doi: 10.48550/arxiv.1901.11150.

[19] J. Sanchez-Palma and J. L. Ordóñez-Ávila, "A PID Control Algorithm With Adaptive Tuning Using Continuous Artificial Hydrocarbon Networks for a Two-Tank System," Ieee Access, vol. 10, pp. 114694–114710, 2022, doi: 10.1109/access.2022.3217209.

[20] L. R. Zwerwer et al., "Identifying the Need for Infection-Related Consultations in Intensive Care Patients Using Machine Learning Models," Sci Rep, vol. 14, no. 1, 2024, doi: 10.1038/s41598-024-52741-w.

[21] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in Proceedings of the 23rd international conference on Machine learning - ICML '06, New York, New York, USA: ACM Press, 2006, pp. 233–240. doi: 10.1145/1143844.1143874.

[22] G. Ryan, P. Katarina, and D. Suhartono, "MBTI Personality Prediction Using Machine Learning and SMOTE for Balancing Data Based on Statement Sentences," Information, vol. 14, no. 4, p. 217, 2023, doi: 10.3390/info14040217.

[23] M. M. Ahsan, S. A. Luna, and Z. Siddique, "Machine-Learning-Based Disease Diagnosis: A Comprehensive Review," Healthcare, vol. 10, no. 3, p. 541, Mar. 2022, doi: 10.3390/healthcare10030541.

[24] L. Hu et al., "Development and Validation of a Deep Learning Model to Reduce the Interference of Rectal Artifacts in MRI-based Prostate Cancer Diagnosis," Radiol Artif Intell, vol. 6, no. 2, Mar. 2024, doi: 10.1148/ryai.230362.

[25] A. Al Bataineh, D. Kaur, and S. M. J. Jalali, "Multi-Layer Perceptron Training Optimization Using Nature Inspired Computing," Ieee Access, vol. 10, pp. 36963–36977, 2022, doi: 10.1109/access.2022.3164669.

[26] S. Sutarman, M. A. I. Hutagalung, O. Darnius, and M. Y. P. El Sya'ban, "Stochastic Gradient Descents Optimizer and Its Variants: Performance of the Optimizers for Multinomial Logistic Models on Large Data Sets by Simulation," Mathematical Modelling and Engineering Problems, vol. 11, no. 10, pp. 2823–2832, 2024, doi: 10.18280/mmep.111025.

[27] Y. Liu et al., "High-Precision Tomato Disease Detection Using NanoSegmenter Based on Transformer and Lightweighting," Plants, vol. 12, no. 13, p. 2559, 2023, doi: 10.3390/plants12132559.

[28] J.-H. Kim, H. Kang, J. Yang, H. Jung, S. Lee, and J. Lee, "Multitask Deep Learning for Human Activity, Speed, and Body Weight Estimation Using Commercial Smart Insoles," IEEE Internet Things J, vol. 10, no. 18, pp. 16121–16133, 2023, doi: 10.1109/jiot.2023.3267335.

[29] K. Sriporn, C.-F. Tsai, C.-E. Tsai, and P. Wang, "Analyzing Malaria Disease Using Effective Deep Learning Approach," Diagnostics, vol. 10, no. 10, p. 744, Sep. 2020, doi: 10.3390/diagnostics10100744.

[30] A. I. Mohammed and A. AK. Tahir, "A New Optimizer for Image Classification using Wide ResNet (WRN)," Academic Journal of Nawroz University, vol. 9, no. 4, p. 1, Sep. 2020, doi: 10.25007/ajnu.v9n4a858.

[31] M. Wang, W. Fu, X. He, S. Hao, and X. Wu, "A Survey on Large-Scale Machine Learning," IEEE Trans Knowl Data Eng, vol. 34, no. 6, pp. 2574–2594, 2022, doi: 10.1109/TKDE.2020.3015777.

[32] W. Aydi and F. S. Alduais, "Estimating Weibull Parameters Using Least Squares and Multilayer Perceptron vs. Bayes Estimation," Computers, Materials & Continua, vol. 71, no. 2, pp. 4033–4050, 2022, doi: 10.32604/cmc.2022.023119.

[33] M. Ilić, G. Keković, V. Mikić, K. Mangaroska, L. Kopanja, and B. Vesin, "Predicting Student Performance in a Programming Tutoring System Using AI and Filtering Techniques," IEEE Transactions on Learning Technologies, vol. 17, pp. 1891–1905, 2024, doi: 10.1109/TLT.2024.3431473.

[34] G. D. Furman, "Enhancing Engineering Education: The Role of Artificial Intelligence in Personalizing Learning and Outcomes," in 2024 4th International Conference on Big Data Engineering and Education (BDEE), IEEE, Aug. 2024, pp. 61–65. doi: 10.1109/BDEE63226.2024.00018.

[35] O. Al-Omari, A. Alyousef, S. Fati, F. Shannaq, and A. Omari, "Governance and Ethical Frameworks for AI Integration in Higher Education: Enhancing Personalized Learning and Legal Compliance," Journal of Ecohumanism, vol. 3, no. 8, Jan. 2025, doi: 10.62754/joe.v3i8.5781.

[36] K. Baltaci, M. Herrmann, and A. Turkmen, "Integrating Artificial Intelligence into Electrical Engineering Education: A Paradigm Shift in Teaching and Learning," in 2024 ASEE Annual Conference & Exposition Proceedings, ASEE Conferences. Doi: 10.18260/1-2–47644.

[37] J. Sun, C.-F. Kwong, and G. Buticchi, "The Potential of AI in Electrical and Electronic Engineering Education: A Review," in 2024 IEEE 11th International Conference on E-Learning in Industrial Electronics (ICELIE), IEEE, Nov. 2024, pp. 1–6. doi: 10.1109/ICELIE62250.2024.10814858.