BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **25**, No 2
Sofia • 2025 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2025-0016

Refining Graduation Classification Accuracy with Synergistic Deep Learning Models

Nguyen Thi Kim Son[1,2], *Nguyen Huu Quynh*[3], *Bui Tuan Minh*[4]

[1]*Hanoi University of Industry, Hanoi, Vietnam*
[2] *Graduate University of Science and Technology, Vietnam Academy of Science and Technology, Hanoi, Vietnam*
[3]*CMC University, Hanoi, Vietnam*
[4]*Thuyloi University, Hanoi, Vietnam*
E-mails: sonntk@haui.edu.vn (*correspondig author*) *nhquynh@cmc-u.edu.vn 2051063681@e.tlu.edu.vn*

Abstract**:** *Learning Analytics plays an important role in monitoring and improving educational outcomes, but is often challenged by limited dataset sizes, resulting from privacy regulations and curriculum changes. This paper proposes the LATCGAd* (*Learning Analysis by Transformer with Conditional Generative Adversarial Framework Network and Adaptive Layer Normalization model*)*, a deep learning framework that combines the Transformer architecture and Conditional Generative Adversarial Network* (*CGAN*) *to overcome the above problems. The CGAN component generates synthetic data samples, which balance and expands the dataset size, while the Transformer leverages this rich dataset to improve prediction performance. The integration of Adaptive Layer Normalization* (*AdaLN*) *in the Transformer also helps stabilize the learning process and minimize overfitting. Experiments on datasets from Hanoi Metropolitan University and Hanoi National University show that the LATCGAd model achieves an accuracy of up to 96.97%, outperforming traditional models such as Decision Tree, SVM and Transformer alone. This result confirms the effectiveness of LATCGAd in educational predictive analysis and its potential for widespread application in the field of learning analytics.*

Keywords**:** *Deep learning, Transformer, CGAN, Graduation classification, Learning analytics.*

1. Introduction

Learning Analytics (LA) involves applying established methods and models to address fundamental issues affecting educational processes and the development strategies of educational institutions [1]. A key application of LA is monitoring and predicting student learning outcomes, while identifying potential issues early for timely intervention [2]. This not only enhances learning performance but also

provides valuable insights for teachers and administrators to make well-informed and effective decisions.

Educational datasets are often tiny, primarily due to the lack of compatibility between educational data systems, making it difficult to combine data from different sources difficult. The differences in curricula across educational institutions further complicate data integration. Additionally, the need to protect privacy and comply with ethical obligations in learning analytics limits data collection and usage. Moreover, the inability to inherit data from previous studies due to privacy concerns reduces the scale and availability of data, leading to the issue of small datasets.

In recent years, machine learning methods such as Decision Trees, Support Vector Machines (SVM), Deep Neural Networks (DNN), Graph Attention Networks (GAT), and Transformers have been widely used in LA to predict student outcomes and performance [3]. However, these models have many limitations when working with small and imbalanced datasets, a common challenge in the educational context due to confidentiality constraints and frequent curriculum changes. Decision Trees, while easy to understand, often suffer from overfitting, especially to small datasets, which reduces their generalizability. SVM models can handle high-dimensional data efficiently but struggle with imbalanced data, leading to biased predictions. DNN models are powerful but require large amounts of labeled data to perform well, which can easily lead to overfitting when data is scarce [4]. GATs are effective at capturing graph-based relationships but are not applicable to traditional tabular datasets [5]. Even Transformers, with their powerful string processing capabilities, require large datasets to realize their full potential, leading to suboptimal performance when dealing with small or imbalanced datasets [6].

To address these challenges, more advanced models are needed that can handle small and imbalanced datasets while still maintaining high prediction accuracy. This paper proposes a novel hybrid model, LATCGAd, which synergistically combines a Transformer and a Conditional Generative Adversarial Network (CGAN). In this hybrid model, each component serves a distinct function to enhance performance, particularly in the context of small datasets. Initially, the CGAN addresses the issue of data scarcity by generating synthetic samples for underrepresented labels, thereby expanding and balancing the dataset. Once the dataset has been augmented, the Transformer Encoder leverages this more diverse data to learn intricate relationships between input features, enhancing its representational capacity. To further optimize performance and ensure stability when working with small data, the model incorporates Adaptive Layer Normalization (AdaLN) into each Transformer Encoder layer. AdaLN dynamically adjusts the normalization parameters based on the specific characteristics of the input data, thereby mitigating bias and variance across the layers. This adjustment promotes more efficient convergence and improves the overall performance of the model, ensuring that the Transformer learns with greater stability and resilience, while also reducing the risk of overfitting—a common challenge when working with small datasets.

Our work offers significant contributions that enhance the accuracy and efficiency of educational predictive analysis:

- Data Balancing with CGAN. LATCGAd effectively combats class imbalance by generating realistic synthetic samples for minority classes. This innovation significantly boosts the detection accuracy for underrepresented categories, specifically focusing on classes such as excellent or medium.
- Enhanced Multi-Class Classification. By employing a stacked Transformer Encoder, LATCGAd demonstrates exceptional performance in multi-class classification, making it particularly valuable for predicting early graduation classification.
- Improved Accuracy and Robust Performance on Small Datasets. Our model employs Adaptive Layer Normalization (AdaLN) within each Transformer Encoder layer, which allows for dynamic adjustments of normalization parameters based on input data characteristics. This approach minimizes bias and variance across network layers, leading to quicker convergence and enhanced overall performance.

These contributions collectively present a powerful, efficient, and precise solution for multi-class educational predictive analysis, effectively addressing the critical challenges existing prediction systems face in diverse, real-world, multi-platform environments.

This paper is structured as follows: Section II provides a brief overview of related research. Section III describes the proposed model and methodology in detail. The effectiveness of the proposed approach will be thoroughly discussed in Section IV, with experiments conducted on datasets from the databases of the University of Education, Vietnam National University (VNU), and Hanoi Metropolitan University, Hanoi, Vietnam. Finally, Section V concludes the paper and discusses future research directions.

2. Related works

In recent years, the use of artificial intelligence in educational research has emerged as a growing trend that has garnered significant attention from scholars. AI is approached as the use of machine learning and deep learning models to analyze educational data more accurately. Several studies have applied data mining methods in education, such as Decision Trees, KNN, Bayes, Association Rules, and Logistic Regression, to predict learning outcomes and student capabilities, achieving promising results [3]. P o p c h e v and O r o z o v a [7] used five algorithms (Decision Tree, Random Forest, Logistic Regression, Naive Bayes, Support Vector Machines, and Neural Network) to classify students based on assessments of their knowledge and skills. K u s t i t s k a y a, K y t m a n o v and N o s k o v [8] explored various ML techniques, including Bayesian Networks, k-Nearest Neighbors (k-NN), and Linear Discriminant Analysis (LDA), for early detection of at-risk students in blended learning settings. In their study, I q b a l et al. [9] applied machine learning techniques such as CF, SVD, NMF, and RBM to analyze the learning data of Electrical Engineering students at ITU. They provided suggestions to improve scores and identified students needing special support during their studies. A l s h a n q i t i and N a m o u n [10] predicted academic performance by combining three techniques: correlation filtering, fuzzy rules, and Lasso linear regression. Other studies [11, 12]

applied machine learning techniques to analyze student behavior and predict students at risk of failing, which drew interest from the academic community. S o n et al. [13] used logistic regression and discriminant feature selection to predict student graduation outcomes based on admission data and academic performance in the first and second years of university. S o n, Q u y n h and M i n h [14] proposed the Learning Analysis by Graph Convolutional Network and Transformer approach for predicting students' graduation ranks. This method combines a Graph Convolutional Network (GCN) to augment the training set with labeled samples, along with a Transformer to predict graduation ranks.

To improve prediction model accuracy and address small dataset limitations, C a r d o n a and C u d n e y [15] employed SVM combined with the Smote technique to predict students likely to drop out. M u k h t a r, A m i e n and D e w i [16] used the Decision Tree method combined with Smote to predict whether students would graduate on time. Y a q i n, R a h a r d i and A b d u l l o h [17] compared KNN, Support Vector Machine, and ANN combined with SMOTE to determine the best algorithm for predicting student on-time graduation.

In 2018, B e n d a n g n u k s u n g [4] introduced a Deep Neural Network (DNN) model, which outperformed other machine learning algorithms in predicting the learning outcomes of students in the same course. P o u d y a l, M o h a m m a d i-A r a g h and B a l l [18] developed a hybrid 2D CNN model to predict student learning outcomes and compared it with traditional machine learning models, showing that the hybrid 2D CNN model outperformed them. K u s u m a w a r d a n i and A l f a r o z i [19] proposed a Transformer Encoder model to predict the order of student learning activities based on their log data in online courses. The study demonstrated that this model outperformed regression models such as LSTM in both accuracy and F1-score, particularly in predicting students at risk of dropping out. The research of Y a n g et al. [20] proposed a Social-path Embedding-based Transformer Neural Network to predict student graduation outcomes. This model not only considered academic performance but also explored students' social relationships, an important factor in deciding career paths after graduation. The results showed that this model outperformed traditional methods in predicting student graduation. However, the application of deep learning in educational data science is still in its infancy, with many studies only emerging in recent years.

3. The proposed method

3.1. Model

In this section, we introduce our proposed model, which combines a Transformer with CGAN under AdaLN. The purpose of this model is to improve the accuracy of predicting student graduation classifications, especially in cases where the dataset size is small. The LATCGAd refers to the integration of three key components to enhance performance in learning data analysis:

- **Conditional Generative Adversarial Network (CGAN).** CGAN is used to generate synthetic samples for underrepresented labels, helping to expand and

balance the dataset. This addresses the issue of data scarcity in underrepresented groups.

- **Transformer encoder.** Once the dataset is expanded, the Transformer Encoder learns from this more diverse data, optimizing its ability to capture complex relationships among input features.
- **Adaptive Layer Normalization (AdaLN).** To improve accuracy and ensure robust performance when working with small datasets, the model integrates AdaLN into each Transformer Encoder layer. AdaLN automatically adjusts the normalization parameters based on the input data's characteristics, reducing bias and variance across the network layers. This promotes faster convergence and enhances overall model performance.

The combination of these three components helps the model learn more stably and reduces overfitting, which is a common issue when working with small datasets.

Fig. 1 illustrates the LATCGAd model, where the combination of the CGAN and Transformer Encoder provides an effective solution for accurately predicting graduation classification on small and imbalanced datasets. In this model, CGAN expands and balances the dataset by generating synthetic samples for specific labels, addressing the issue of data scarcity in underrepresented groups. Once the dataset is expanded and balanced, the Transformer Encoder learns from this diverse dataset, optimizing its ability to capture complex relationships among input features. Notably, to improve accuracy and ensure robust performance on small datasets, the model integrates AdaLN into each Transformer Encoder layer. Adaptive Layer Normalization automatically adjusts normalization parameters based on the characteristics of the input data. It reduces bias and variance across network layers, enhancing convergence and overall model performance. As a result, the Transformer learns more stably and mitigates overfitting, a common challenge when working with small datasets. The tight integration of CGAN, Adaptive Layer Normalization, and Transformer not only improves accuracy but also enhances precision, recall, and F1-score, enabling the model to make more reliable and comprehensive predictions.

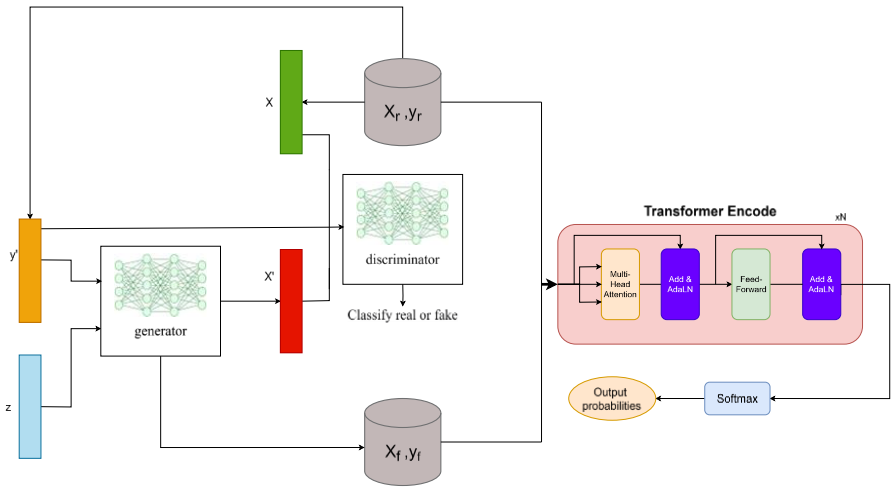


Fig. 1. The proposed model LATCGAd

The operation of the proposed model is depicted in Fig. 1. Real data samples X_r (which contain student-related information, such as survey data and academic scores from the first two years of study) are collected along with the labels y' (representing the actual academic ranking after students graduate, serving as the ground truth labels for training and evaluating the model. This label is used in both the CGAN and Transformer components: in CGAN, it acts as a condition during data generation to ensure that the synthetic data aligns with specific academic ranking categories; in the Transformer model, it serves as the target variable for the classification task. To generate additional synthetic data and expand the training dataset, we propose integrating CGAN into the model. Using the original data, we train the CGAN model, where the generator takes in noise vectors z (z is a random input that allows the Generator to create diverse synthetic data instead of repeating a single sample for each label) and labels y' to create synthetic data. The discriminator then distinguishes between real and synthetic data using the labels y' (y' is the label that helps the Generator create synthetic data with the correct label for each class. Meanwhile, the Discriminator uses y' to verify whether the generated data matches the assigned label, allowing the generator to improve and produce data that closely resembles the real data (see details in Fig. 2). After training the CGAN, we will use the generator to generate additional new data X_f (synthetic student-related data generated by CGAN, including simulated survey responses and academic scores from the first two years, corresponding to each y_f) and y_f. We chose CGAN to allow the generation of data based on specific classification labels.

Next, the newly generated datasets X_f and y_f are combined with the original data X_r and y_r to form a larger dataset, which is then used to train the Transformer model for predicting student graduation classifications.

In this paper, we utilize a generator with three hidden layers and a discriminator with four hidden layers. We apply the Adam optimizer, learning rate, and Beta_1.

For the Transformer model, we only use the Transformer Encoder. The final output will be a latent feature vector, which will then be passed through a fully connected layer for classification prediction. We use parameters such as multi-head attention, feed-forward layers, the number of Transformer encoder layers, the Adam optimizer, learning rate, and weight decay.

3.2. Conditional Generative Adversarial Network (CGAN)

Our proposed model utilizes the Conditional Generative Adversarial Network (CGAN). The Conditional Generative Adversarial Network (CGAN) is a variant of the original Generative Adversarial Network (GAN) introduced by Goodfellow in 2014. The CGAN algorithm was developed by researchers Mehdi Mirza and Simon Osindero, as detailed in their paper titled “Conditional Generative Adversarial Nets” [21]. This section provides an overview of the CGAN model, which serves as a key component of our approach, illustrated in Fig. 2.

The Conditional Generative Adversarial Network (CGAN) consists of two main components: (1) the Generator Neural Network, and (2) the Discriminator Neural Network.

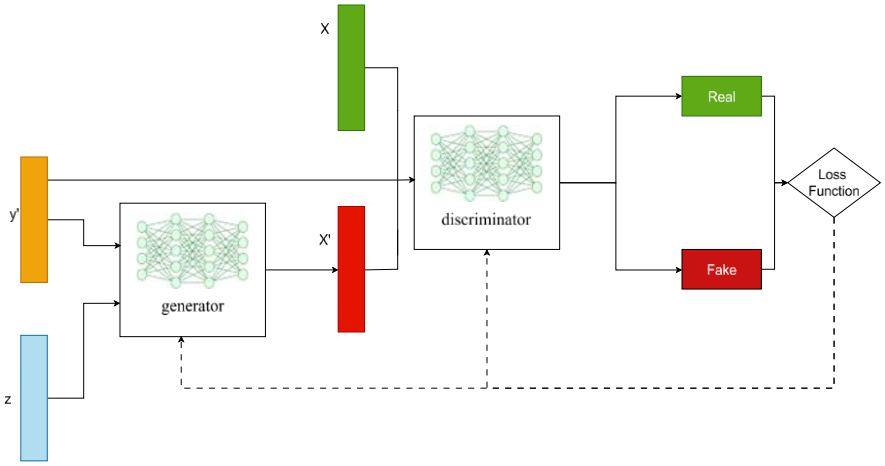


Fig. 2. CGAN model

Since CGAN is a conditional generative model, both the Generator and Discriminator networks are trained simultaneously, with both receiving the label of the data as input, ensuring they generate and evaluate data that aligns with specific labels. The CGAN operates as follows: the Generator network takes as input a noise vector z and a condition label y', generating new data according to the condition provided by y. The real samples (x, y') and the newly generated samples (x', y') are then passed to the Discriminator network, which distinguishes between real and fake samples. This process is akin to a **min-max game** between two players, with the loss function calculated as described below.

(1) $$\min_G \max_D V(D,G) = E_{x\sim p_{\text{data}}(x)}[\log D(x \mid y)] + E_{z\sim p_z(z)}[\log(1 - D(G(z \mid y) \mid y))].$$

For real data input x and label y, $\log D(x \mid y)$ is the probability that the Discriminator believes the data x (with label y) is real. The Discriminator's goal is to maximize this probability as much as possible. Meanwhile, $G(z \mid y)$ generates fake data using the Generator model from the noise matrix z, based on label y. $\log(1 - D(G(z \mid y) \mid y))$ is the probability that the Discriminator believes the newly generated data (with label y) is fake. The Discriminator aims to maximize this probability, while the Generator's goal is to make $D(G(z \mid y) \mid y)$ as close to 1 as possible, meaning it has successfully fooled the Discriminator.

3.3. Transformer model

Our proposed model also incorporates the **Transformer** to predict student classifications. This section provides an overview of the Transformer model, which serves as a core component in our approach, illustrated in Fig. 3 [6].

The Transformer model consists of two main components: the Encoder and the Decoder. Both parts consist of multiple layers, each containing two key components: Multi-Head Self-Attention and Feed-Forward Neural Networks.

- The Encoder is responsible for transforming the input into a latent feature vector representation that the model can understand and use for various tasks.
- The Decoder is responsible for generating outputs based on the latent feature vector representation from the Encoder. The Decoder's goal is to compute the

probability of the latent feature vectors and determine the output. The result could be a label for classification models or a target sequence based on the source sequence encoded by the Encoder.

- Self-attention is a special case of the attention mechanism, where the Query, Key, and Value all come from the same data sequence. This mechanism helps the model learn relationships between words in the same sequence without having to process them sequentially, as in RNN models. Multi-Head Attention extends the self-attention mechanism by using multiple "attention heads" to compute attention in parallel. Instead of using a single attention mechanism, the model splits the hidden representation into multiple parts and computes attention on each part separately. The results are then combined and further processed.
- Feed-Forward is a crucial component applied after each Multi-Head Attention step in both the Encoder and Decoder. The Feed-Forward adds nonlinearity and creates more complex representations, improving performance in tasks involving sequential data processing, such as machine translation, text classification, or other natural language processing tasks. Each Feed-Forward in the Transformer is a feed-forward neural network with two fully connected layers and a nonlinear activation function in between.

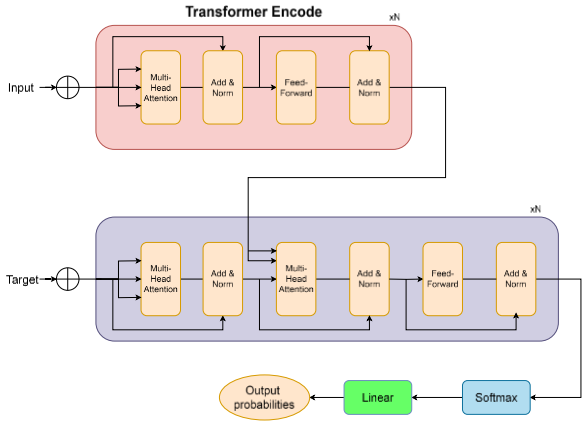


Fig. 3. Transformer model

Adaptive Layer Normalization (AdaLN) is an extension of Layer Normalization that adapts normalization parameters based on input data characteristics. It is particularly useful in deep learning models, like Transformers, where input data varies during training. AdaLN enhances model performance by dynamically adjusting parameters to reduce bias and variance across layers. In Layer Normalization, normalization is applied by calculating the mean μ and standard deviation σ of the inputs:

(2) $$\hat{x_i} = \frac{x_i - \mu}{\sigma}.$$

After normalization, learned parameters γ and β (scale and shift) are applied:

(3) $$y_i = \gamma \hat{x_i} + \beta.$$

AdaLN adapts these parameters for each input dynamically:

(4) $$y_i = \gamma(x) \times \hat{x_i} + \beta(x),$$

where $\gamma(x)$ and $\beta(x)$ are computed based on the input x for each layer. These parameters are learned through a sub-network, allowing for adjustment according to the data's characteristics.

In Transformer models, AdaLN improves stability and efficiency, especially when working with small or heterogeneous datasets, ensuring effective learning across layers.

4. Experiments

4.1. Datasets

In this paper, we use three datasets consisting of students majoring in primary education from Hanoi Metropolitan University (HNMU), math education from Hanoi Metropolitan University, and literature education from the University of Education, Vietnam National University, Hanoi (VNU). We collected these datasets from the universities. The preprocessing steps included handling missing data by removing attributes with over 60% missing values, and if there are fewer, we will fill the missing values with the mean. We then checked for random distributions of scores within columns, analyzed correlations with the "rating" column, and removed insignificant columns.

The first dataset includes students majoring in primary education who studied at HNMU from 2014 to 2021. This dataset, labeled by HNMU1, was collected and processed over 18 months, from March 2020 to September 2021. After cleaning, the dataset contains 932 records and 21 variables (comprising 3 survey data and 20 grade data (scores for the first two years of students)) for training, with actual labels (11 Medium classification labels,430 Good classification labels, 468 Very Good classification labels, and 23 Excellent classification labels). The dataset is imbalanced in the Medium and Excellent classes, with only 11 and 23 samples. The detailed distribution of the HNMU1 dataset is shown in Fig. 4.

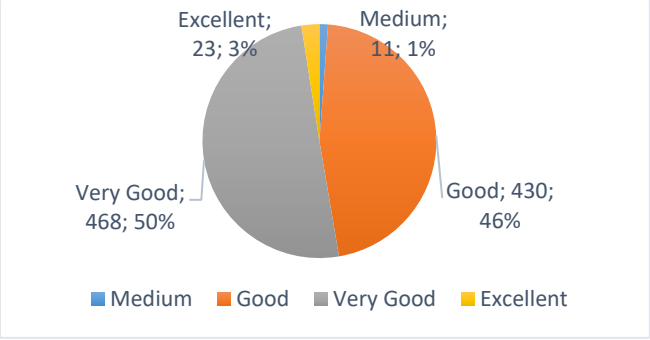


Fig. 4. The structure of the dataset HNMU1

The second dataset consists of students majoring in math education at HNMU from 2014 to 2023. This dataset, labeled by HNMU2, was collected and processed over two years, from 2022 to 2023, with 551 records and 62 variables (comprising 34 survey data and 28 grade data (scores for the first two years of students)) for training, with actual labels (19 Medium classification labels, 337 Good classification

labels, 191 Very Good classification labels, and four Excellent classification labels). The dataset is imbalanced in the Medium and Excellent classes, with only 19 and 4 samples. The structure of the dataset HNMU2 is shown in Fig. 5.

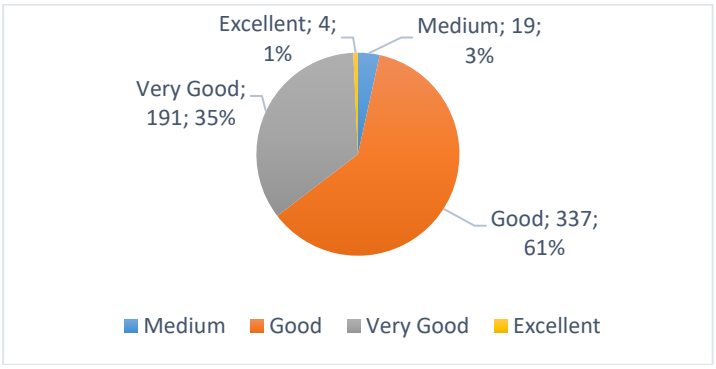


Fig. 5. The structure of the dataset HNMU2

The third dataset, labeled by VNU dataset, contains students majoring in literature education at VNU from 2014 to 2023. Collected and processed over three years, from 2021 to 2023, it contains 271 records and 72 variables (comprising 48 survey data and 24 grade data (scores for the first two years of students)) for training, with corresponding labels assigned to each student, including Good, Very Good, and Excellent, with actual labels (46 Good classification labels, 187 Very Good classification labels, and 38 Excellent classification labels). The VNU dataset is less imbalanced compared to HNMU1 and HNMU2; however, the Good and Excellent classes are still imbalanced relative to the most prevalent class, Very Good (Fig. 6).

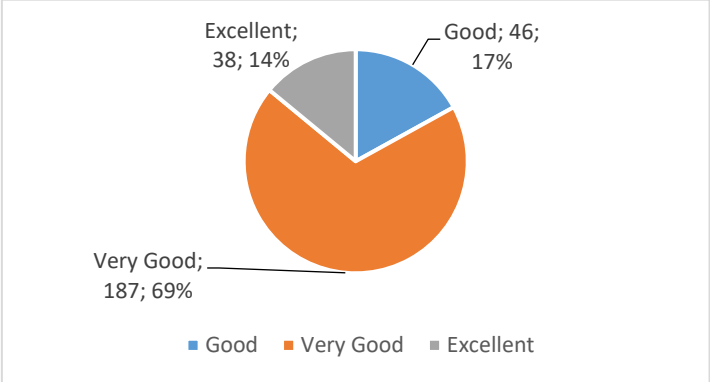


Fig. 6. The structure of the dataset VNU

4.2. Evaluation metrics

The evaluation metrics used include: Accuracy (Acc), Macro Averaged Precision (P), Macro Averaged Recall (R), and Macro F1-score (F1), calculated using the following formulas:

(5) $$\text{Acc} = \frac{\text{Correct predictions}}{\text{All predictions}},$$

(6) $$P = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FP}_i},$$

(7) $$R = \frac{1}{N}\sum_{i=1}^{N}\frac{\mathrm{TP}_i}{\mathrm{TP}_i+\mathrm{FN}_i},$$

(8) $$\mathrm{F1} = \frac{2\times P\times R}{P+R},$$

where N is the number of classes, TP_i (True Positive of class i), FP_i (False Positive of class i), and FN_i (False Negative of class i) are key metrics in classification tasks, and all predictions are the total number of data samples.

4.3. Experimental evaluation

To demonstrate the effectiveness of our proposed model, we will test it on these three datasets: HNMU1, HNMU2, and VNU. We will use survey data and student scores from their first two years to predict student classification. By using real-world data, we aim to show that our proposed model is effective in practical applications. The dataset is divided into train, validation, and test sets, with 60% of the data used for training, 10% for validation, and 30% for testing.

4.3.1. Experimental setup

We will compare the proposed model against three different deep learning algorithms (DNN, GAT, and Transformer), and traditional machine learning methods (which are known to perform well with small datasets): Decision Tree, SVM, and Logistic Regression.

Machine learning models:

- **Decision tree.** Uses entropy for information calculation, with a minimum of 1 sample per leaf and 4 samples for splits.
- **SVM.** Utilizes the RBF kernel, penalty coefficient = 1, and gamma = "scale" to automatically calculate the gamma value based on the characteristics of the data.
- **Logistic regression.** Ridge regularization, inverse regularization coefficient = 1, and lbfgs optimization (Limited-memory Broyden-Fletcher-Goldfarb-Shanno)

Deep learning models. The parameters are optimized through the cross-validation and grid search process to determine the optimal combination for each dataset. The main criteria for parameter selection include minimizing validation loss, achieving high accuracy, and improving class balance.

DNN model. A 4-layer DNN with ReLU and softmax activation functions. The first layer has 512 neurons, the second 256, the third 64, and the output layer matches the number of classes in each dataset. The model uses the Adam optimizer with lr = 0.005 and dropout = 0.6.

CGAN model. This CGAN model is structured with two main components: the Generator and the Discriminator. The Generator in the CGAN network consists of three layers to generate new data from the latent space. Specifically, the first layer of the Generator has 256 neurons, the second layer has 512 neurons, and the third layer has 1024 neurons. The output of the Generator is 21 for HNMU1, 62 for HNMU2, and 72 for VNU. The activation function for HNMU1 and VNU is LeakyReLU with a coefficient of 0.2, and for HNMU2, it is ReLU. The Adam optimizer is used with a learning rate (lr) of 0.0002 and Beta_1 of 0.5. The loss function is Binary Cross Entropy Loss, which helps the network learn nonlinear features effectively and avoid

neuron death. The Discriminator also consists of 4 layers to evaluate the authenticity of the data generated by the Generator. Specifically, the first layer of the Discriminator has 1024 neurons, the second layer has 512 neurons, the third layer has 256 neurons, and the fourth layer has 64 neurons. The activation function for HNMU1 and VNU is LeakyReLU with a coefficient of 0.2, and for HNMU2, it is ReLU. The Adam optimizer is used with lr = 0.0002 and beta_1 = 0.5. The loss function is Binary Cross-Entropy Loss. The output of the Critic is a single value representing the Discriminator's score for the input sample.

- CGAN generates for HNMU1 an additional 32 samples per one class.
- CGAN generates for HNMU2 an additional 25 samples per one class.
- CGAN generates for VNU an additional 12 samples per one class.

Table 1. Number of samples before and after creation with CGAN on the HNMU dataset

Label	Before generating	After generating
Medium	11	43
Good	430	462
Very Good	468	500
Excellent	23	55
Total	**932**	**1060**

Table 2. Number of samples before and after creation with CGAN on the HNMU2 dataset

Label	Before generating	After generating
Medium	19	51
Good	337	369
Very Good	191	223
Excellent	4	36
Total	**551**	**679**

Table 3. Number of samples before and after creation with CGAN on the VNU dataset

Label	Before generating	After generating
Good	46	58
Very Good	187	199
Excellent	38	50
Total	**271**	**307**

The parameters of the CGAN model (such as the number of layers, learning rate, and activation functions) are tailored for each dataset to ensure that the synthetic data closely resembles the real data. For HNMU1 and VNU, the LeakyReLU activation function is used in both the generator and discriminator, whereas ReLU is more effective for HNMU2. The number of synthetic data samples for each class is also adjusted differently for each dataset to ensure class balance without introducing excessive noise.

GAT Model. We use a 2-layer model with the HNMU1, HNMU2, and VNU datasets.

- For the HNMU1 dataset, the first layer has eight attention heads and calculates the number of features as six (output of each attention head), resulting in 48 features. The activation function is ELU. The second layer has 1 attention head since it is used for classification, and the activation function is softmax. Models use the Adam optimizer with lr = 0.005 and dropout = 0.6.

- For the HNMU2 dataset, the first layer has eight attention heads and calculates the number of features as eight (output of each attention head), resulting in 64 features. The activation function is ELU. The second layer has one attention head since it is used for classification, and the activation function is softmax. Models use the Adam optimizer with lr = 0.005 and dropout = 0.6.
- For the VNU dataset, the first layer has eight attention heads and calculates the number of features as six (output of each attention head), resulting in 48 features. The activation function is ELU. The second layer has one attention head since it is used for classification, and the activation function is softmax. Models use the Adam optimizer with lr = 0.005 and dropout = 0.6.

Transformer Model.

- For HNMU1, the Transformer uses two multi-heads. The feed-forward layer in each encoder layer has 64 units. The number of Transformer encoder layers is one, with dropout = 0.6. This is followed by a fully connected network with an output of four (corresponding to the number of classes in the HNMU1 dataset). Models use the Adam optimizer with lr = 0.005 and weight decay = 0.0005.
- For HNMU2, the Transformer uses seven multi-heads. The feed-forward layer in each encoder layer has 64 units. The number of Transformer encoder layers is two, with dropout = 0.5. This is followed by a fully connected network with an output of four (corresponding to the number of classes in the HNMU2 dataset). Models use the Adam optimizer with lr = 0.005 and weight decay = 0.0005.
- For VNU, the Transformer uses two multi-heads. The feed-forward layer in each encoder layer has 128 units. The number of Transformer encoder layers is one, with dropout = 0.6. This is followed by a fully connected network with an output of three (corresponding to the number of classes in the VNU dataset). Models use the Adam optimizer with lr = 0.005 and weight decay = 0.0005.

Table 4. Generator model parameter table on the HNMU1, HNMU2, and VNU datasets

Dataset	First layer	Second layer	Third layer	Activation function	Output layer	Output activation function
HNMU1	256	512	1024	LeakyReLU(0.2)	21	Tanh
HNMU2	256	512	1024	ReLU	62	Tanh
VNU	256	512	1024	LeakyReLU(0.2)	72	Tanh

Table 5. Training parameter table for the Generator model on the HNMU1, HNMU2, and VNU datasets

Dataset	Optimizer	Learning Rate	Beta_1	Loss function
HNMU1	Adam	0.0002	0.5	Binary Cross-Entropy Loss
HNMU2	Adam	0.0002	0.5	Binary Cross-Entropy Loss
VNU	Adam	0.0002	0.5	Binary Cross-Entropy Loss

Table 6. Discriminator model parameter table on the HNMU1, HNMU2, and VNU datasets

Dataset	First layer	Second layer	Third layer	Fourth Layer	Activation function	Third layer	Output activation function
HNMU1	1024	512	256	64	LeakyReLU(0.2)	1	Sigmoid
HNMU2	1024	512	256	64	ReLU	1	Sigmoid
VNU	256	512	256	64	LeakyReLU(0.2)	1	Sigmoid

Table 7. Training parameter table for the Discriminator model on the HNMU1, HNMU2, and VNU datasets

Dataset	Optimizer	Learning rate	Beta_1	Loss function
HNMU1	Adam	0.0002	0.5	Binary Cross-Entropy Loss
HNMU2	Adam	0.0002	0.5	Binary Cross-Entropy Loss
VNU	Adam	0.0002	0.5	Binary Cross-Entropy Loss

Table 8. Transformer model parameter table on the HNMU1, HNMU2, and VNU datasets

Dataset	Multi-head	Feed-forward layer	Number of transformer encodes	Fully connected layer	Activation function
HNMU1	2	64	1	4	Softmax
HNMU2	7	64	2	4	Softmax
VNU	2	128	1	3	Softmax

Table 9. Training parameter table for the Transformer model on the HNMU1, HNMU2, and VNU datasets

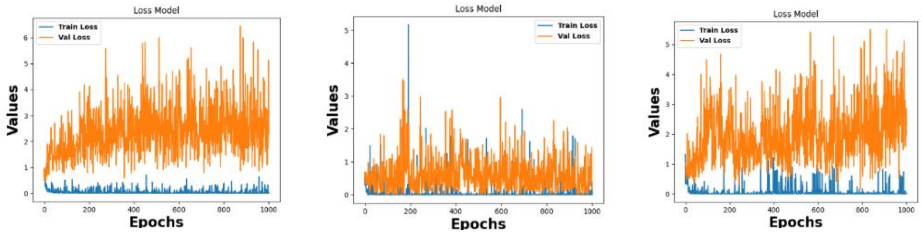
Dataset	Optimizer	Learning rate	Weight decay	Loss function	DropOut
HNMU1	Adam	0.005	0.0005	Cross-Entropy Loss	0.6
HNMU2	Adam	0.005	0.0005	Cross-Entropy Loss	0.5
VNU	Adam	0.005	0.0005	Cross-Entropy Loss	0.6

The number of attention heads is selected based on the complexity of the feature space. For HNMU1 and VNU, two multi-heads are appropriate due to the smaller number of variables in the datasets. In contrast, HNMU2 requires seven multi-heads to learn complex patterns from a larger feature space. Feed-Forward Layer: The number of units in the feed-forward layers is determined by the complexity and size of the dataset. For HNMU1 and HNMU2, the feed-forward layers have 64 units, whereas VNU requires 128 units due to the higher number of variables in the dataset. Learning Rate and Optimizer: The learning rate is set to 0.005 with the Adam optimizer after experimenting with different values and observing the convergence speed and accuracy.

During the model training process, several important parameters were utilized to optimize and adjust the model's learning capability, including Beta_1, lr, and Dropout. Beta_1 is a parameter in the Adam optimization algorithm that determines the exponential decay rate of the first moment estimate, helping the model update gradients more stably and efficiently. In the paper, a Beta_1 value of 0.5 was chosen to balance convergence speed and learning stability. The lr controls the adjustment speed of the model's weights after each gradient update. This value was set differently for each model: 0.0002 for CGAN and 0.005 for Transformer, ensuring optimal convergence speed while preventing oscillations or slow convergence. Additionally, Dropout was used as a regularization technique to mitigate overfitting by randomly dropping some neurons during training. In the paper, the Dropout value was set at either 0.5 or 0.6, depending on the model and dataset, to enhance generalization and model robustness when applied to real-world data.

4.3.2. Model training

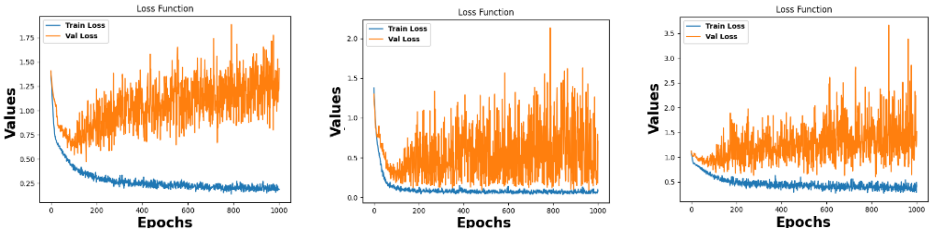
We trained the DNN model with the three datasets HNMU1, HNMU2, and VNU, training the model for 1000 epochs. The images of the loss values for HNMU1, HNMU2, and VNU are shown respectively in Fig. 7. The principle for selecting the best model is that the model with the lowest average of train loss and validation loss will be chosen. Based on this principle, for the HNMU1 dataset, the model selected at epoch 357 has a train loss of 0.0363 and a validation loss of 0.5976. For the HNMU2 dataset, the model selected at epoch 934 has a train loss of 0.0002 and a validation loss of 0.0016. For the VNU dataset, the model selected at epoch 791 as a train loss of 0.0351 and a validation loss of 0.1696. Although the experimental results already allow for selecting the best model, the charts provide deeper insights into the training process and enhance the transparency and reliability of the experiment.



(a) (b) (c)

Fig. 7. Training the DNN model: on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)

We trained the GAT model with the three datasets HNMU1, HNMU2, and VNU, training the model for 1000 epochs. The images of the loss values for HNMU1, HNMU2, and VNU are shown respectively in Fig. 8. The principle for selecting the best model is the model with the lowest average of train loss and validation loss. Based on this principle, for the HNMU1 dataset, the model selected at epoch 154 has a train loss of 0.3432 and a validation loss of 0.4730. For the HNMU2 dataset, the model selected at epoch 891 has a train loss of 0.0433 and a validation loss of 0.0976. For the VNU dataset, the model selected at epoch 557 has a train loss of 0.3635 and a validation loss of 0.6443.

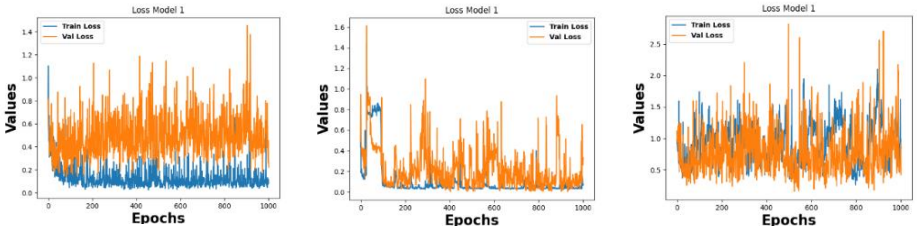


(a) (b) (c)

Fig. 8. Training the GAT model: on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)

We trained the Transformer model for 1000 epochs for the three datasets HNMU1, HNMU2, and VNU. The training graphs of the models for the three datasets

HNMU1, HNMU2, and VNU are shown respectively in Fig. 9. The principle for selecting the best model is to take the average of the training loss and validation loss, and the epoch with the smallest value will be chosen. Based on this principle, for the model in Fig. 9a, the model selected at epoch 132 has a train loss of 0.0849 and a validation loss of 0.1451. For the model in Fig. 9b, the model selected at epoch 275 has a train loss of 0.0405 and a validation loss of 0.0481. For the model in Fig. 9c, the model selected at epoch 44 has a train loss of 0.2905 and a validation loss of 0.2547.



(a) (b) (c)

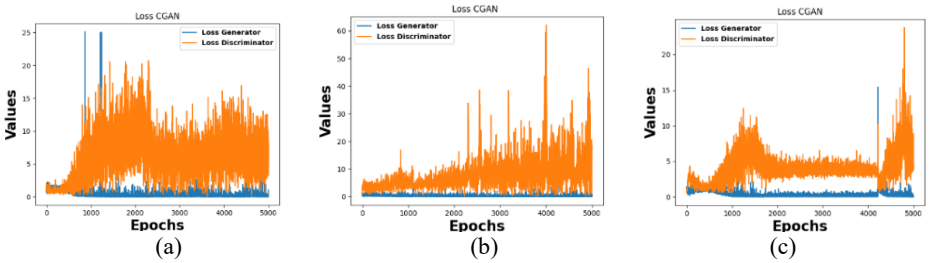
Fig. 9. Training the Transformer model: on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)

For LATCGAd model, the number of epochs for training the CGAN model for the three datasets HNMU1, HNMU2, and VNU is 5000 epochs. The training graphs of the models are shown respectively in Fig. 10. The principle for model selection is that the CGAN model with the smallest FID value (FID is a method for assessing the difference between generated data and real data) will be chosen.

$$\text{FID} = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}, \tag{9}$$

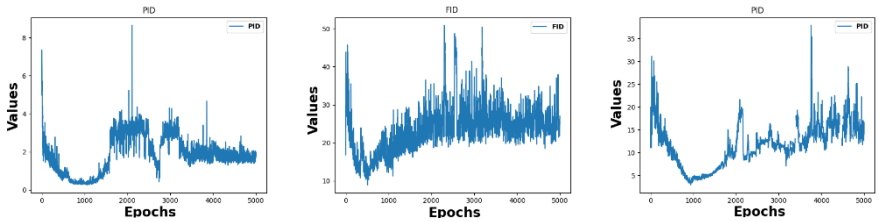
where μ_r, μ_g are the average vector of features of real-world data and generated data. Σ_r, Σ_g are the variance matrix of real-world data and generated data. Tr is the trace of the matrix, i.e., the sum of the elements on the main diagonal of the matrix. ||.|| is the Euclidean distance between two vectors.

Based on this principle, for the model in Fig. 11a, the model was selected at epoch 1073 because it has an FID of 0.2561. For the model in Fig. 11b, the model was selected at epoch 511 because it has an FID of 8.900. For the model in Fig. 11c, the model was selected at epoch 929 because it has an FID of 2.5141.



(a) (b) (c)

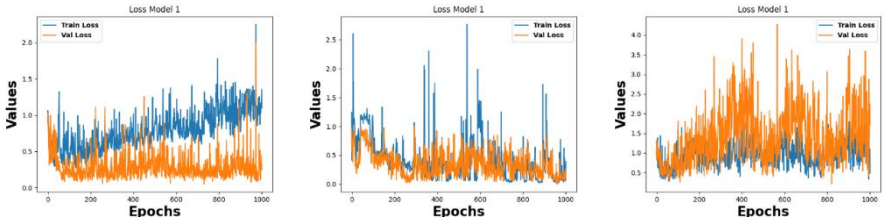
Fig. 10. Training the CGAN model (in the LATCGAd model): on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)



(a) (b) (c)

Fig. 11. FID values: on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)

We will then train the Transformer model for 1000 epochs for the three datasets HNMU1, HNMU2, and VNU. The training graphs for the models for the three datasets HNMU1, HNMU2, and VNU are shown respectively in Fig. 12. The principle for selecting the best model is to take the average of the training loss and validation loss, and the epoch with the smallest value will be chosen. Based on this principle, for the model in Fig. 12a, the model selected at epoch 71 has a train loss of 0.2677 and a validation loss of 0.1237. For the model in Fig. 12b, the model selected at epoch 962 has a train loss of 0.0361 and a validation loss of 0.0018. For the model in Fig. 12c, the model selected at epoch 61 has a train loss of 0.3878 and a validation loss of 0.2793.



(a) (b) (c)

Fig. 12. Training the Transformer model (in the LATCGAd model): on the HNMU1 dataset (a); on the HNMU2 dataset (b); on the VNU dataset (c)

4.3.3. Experimental results

Experimental results on the three datasets (HNMU1, HNMU2, and VNU) show that the LATCGAd model outperforms traditional models (Decision Tree, SVM, Logistic Regression) and deep learning models (DNN, GAT, and standard Transformer).

On the HNMU1 dataset, LATCGAd achieves an accuracy of 95.56%, significantly higher than Decision Tree (89.64%), SVM (84.64%), Logistic Regression (92.86%), DNN (93.57%), and GAT (82.14%) (Table 10). In addition to accuracy, the model also improves Precision (72.50%), Recall (74.78%), and F1-score (73.61%), demonstrating its ability to reduce errors and correctly classify almost all true positive samples, outperforming all compared models.

On the HNMU2 dataset, LATCGAd maintains the highest performance with an accuracy of 96.97%, surpassing standard Transformer (95.15%), Decision Tree (89.70%), SVM (82.42%), Logistic Regression (74.55%), DNN (86.06%), and GAT (90.91%). The model also maintains a well-balanced Precision (73.26%) and Recall (74.09%). However, Decision Tree achieves a higher Precision (94.65%), likely due to its conservative approach in making predictions, which also increases the risk of

overfitting. Although Decision Tree has the highest Recall (79.26%), LATCGAd remains more stable and generalizes better, thanks to synthetic data from CGAN (Table 11).

Table 10. Prediction Results on the HNMU1 Dataset

Method	Accuracy	Precision	Recall	F1-score
Decision tree	89.64	39.77	48.09	42.90
SVM	84.64	35.95	45.31	38.77
Logistic Regression	92.86	43.13	49.05	45.71
DNN	93.57	69.07	74.15	71.35
GAT	82.14	34.81	44.57	37.46
Transformer	93.57	44.71	47.96	46.26
LATCGAd	95.56	72.50	74.78	73.61

Table 11. Prediction Results on the HNMU2 Dataset

Method	Accuracy	Precision	Recall	F1-score
Decision tree	89.70	94.65	79.26	82.48
SVM	82.42	66.76	51.63	54.15
Logistic Regression	74.55	59.08	64.47	58.77
DNN	86.06	68.28	69.35	68.12
GAT	90.91	57.96	55.20	56.07
Transformer	95.15	72.26	73.30	72.72
LATCGAd	96.97	73.26	74.09	73.66

On the VNU dataset, LATCGAd achieves an accuracy of 87.65%, outperforming Decision Tree (83.95%), SVM (83.95%), Logistic Regression (71.60%), DNN (74.07%), GAT (81.48%), and standard Transformer (86.42%). A key advantage is that Precision increases to 95.56%, significantly surpassing the other models, indicating its high reliability in predicting positive cases and minimizing false positives (Table 12). However, Recall is 58.73%, slightly lower than DNN (66.91%) and Transformer (61.41%). This trade-off is justified by its optimized Precision, making it suitable for applications requiring high confidence in identifying critical cases. The F1-score of LATCGAd on the VNU dataset reaches 67.62%, surpassing most machine learning models, though slightly lower than standard Transformer (67.89%).

Table 12. Prediction Results on the VNU Dataset

Method	Accuracy	Precision	Recall	F1-score
Decision tree	83.95	67.59	55.08	59.06
SVM	83.95	52.19	50.83	51.05
Logistic Regression	71.60	64.91	64.83	59.44
DNN	74.07	58.19	66.91	60.71
GAT	81.48	62.63	61.49	61.68
Transformer	86.42	80.83	61.41	67.89
LATCGAd	87.65	95.56	58.73	67.62

A key factor influencing the experimental results is the differences in characteristics among the three datasets: HNMU1, HNMU2, and VNU. Each dataset varies in size, number of features, and class imbalance levels, which directly impact the models' performance.

Specifically, the HNMU1 dataset has the largest sample size (932 samples) but a limited number of features (21 features, with only three survey-based features). As a result, the model primarily relies on academic scores from the first two years. While this enables the model to quickly identify learning trends, it also increases the risk of missing additional insights from non-academic factors. By balancing the data with CGAN, LATCGAd significantly improves accuracy and F1-score compared to other models.

The HNMU2 dataset is smaller (551 samples) but contains 62 features (including 34 survey-based features), providing a more comprehensive view of students. This allows LATCGAd to capture multidimensional relationships between academic and non-academic data. As a result, HNMU2 achieves the highest accuracy (96.97%), while maintaining a balance between Precision and Recall, demonstrating that rich and diverse data plays a crucial role in enhancing model performance.

In contrast, the VNU dataset has the smallest sample size (271 samples) but includes 72 features. Despite the dataset's high feature richness, its small size makes the model more susceptible to overfitting. The high Precision (95.56%) indicates that LATCGAd is highly effective in reducing false positives, but the low Recall (58.73%) suggests that some true positive samples were missed, likely due to the limited training data. The differences in size and composition across these datasets highlight the necessity of data balancing and augmentation using CGAN, particularly when working with small or highly imbalanced datasets. Additionally, this underscores the importance of feature selection and analysis in optimizing deep learning model performance.

In summary, LATCGAd demonstrates superior accuracy and overall performance across all three datasets, particularly in the context of small and imbalanced data, thanks to the combination of data generation from CGAN and model optimization through Adaptive Layer Normalization.

As previously discussed, educational datasets are often limited by confidentiality constraints and data collection challenges, resulting in small sample sizes that are not publicly available. While the proposed model has shown superior performance when evaluated on three independent datasets from two universities, the comparison results remain somewhat unsatisfactory due to the lack of inheritance from prior published research. This limitation is an important consideration for future educational research, particularly in the context of data sharing and the integration of existing findings.

5. Conclusion

The LATCGAd model proposed in this paper introduces a novel approach to addressing challenges in educational predictive analytics, particularly when working with small and imbalanced datasets. By combining CGAN's ability to generate synthetic data with Transformer's strength in capturing complex relationships, the proposed model significantly enhances prediction accuracy and generalization capability. The integration of Adaptive Layer Normalization (AdaLN) further

improves stability and mitigates overfitting, enabling the model to perform consistently across diverse educational datasets.

Experimental results on three real-world datasets (HNMU1, HNMU2, and VNU) demonstrate the superiority of LATCGAd over traditional machine learning models and other deep learning approaches. Notably, the model achieves 96.97% accuracy on the HNMU2 dataset, showcasing its ability to leverage synthetic data for enhanced predictive performance. These findings confirm that LATCGAd is a reliable framework for improving learning analytics and providing practical solutions for educational institutions in forecasting and enhancing student outcomes.

Future research could further refine the model by incorporating additional data sources, optimizing the CGAN architecture, or extending the LATCGAd framework to domains beyond education. The combination of data augmentation and deep learning continues to unlock significant potential in solving complex predictive tasks across various fields.

References

1. Dutt, A., M. A. Ismail, T. Herawan. A Systematic Review on Educational Data Mining. – IEEE Access, Vol. **5**, 2017, pp. 15991-16005.
2. Bienkowski, M., M. Feng, B. Means. Enhancing Teaching and Learning through Educational Data Mining and Learning Analytics. – U.S. Department of Education, Office of Educational Technology, 2012.
3. Xu, X., J. Wang, H. Peng, R. Wu. Prediction of Academic Performance Associated with Internet Usage Behaviors Using Machine Learning Algorithms. – Computers in Human Behavior, Vol. **98**, 2019, pp. 166-173.
4. Bendangnuksung, D. P. Students' Performance Prediction Using Deep Neural Network. – International Journal of Applied Engineering Research, 2018, pp. 1171-1176.
5. Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio. Graph Attention Networks. – In: Proc. of International Conference on Learning Representations (ICLR'18), 2018. DOI: 10.48550/arXiv.1710.10903.
6. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin. Attention is All You Need. – Advances in Neural Information Processing Systems (NeurIPS), 2017, pp. 5998-6008.
7. Popchev, I. P., D. A. Orozova. Towards a Multistep Method for Assessment in e-Learning of Emerging Technologies. – Cybernetics and Information Technologies, Vol. **20**, 2020, No 3, pp. 116-129.
8. Kustitskaya, T. A., A. A. Kytmanov, M. V. Noskov. Early Student-at-Risk Detection by Current Learning Performance and Learning Behavior Indicators. – Cybernetics and Information Technologies**,** Vol. **22**, 2022, No 1, pp. 117-130.
9. Iqbal, Z., J. Qadir, A. N. Mian, F. Kamiran. Machine Learning Based Student Grade Prediction: A Case Study. – arXiv preprint, 2021.
10. Alshanqiti, A., A. Namoun. Predicting Student Performance and Its Influential Factors Using Hybrid Regression and Multi-Label Classification. – IEEE Access, Vol. **8**, 2020, pp. 203827-203844. DOI: 10.1109/access.2020.3036572.
11. Waheed, H., S. Hassan, N. R. Aljohani, J. Hardman, R. Nawaz. Predicting Academic Performance of Students from VLE Big Data Using Deep Learning Models. – Computers in Human Behavior, Vol. **104**, 2020, pp 106-189. DOI: 10.1016/j.chb.2019.106189.
12. Wardle, C. Challenges of Content Moderation: Define "Harmful Content" (online). Retrieved 1 September 2023.
https://tinyurl.com/ys5wc3t6

13. S o n, N. T. K., N. V. B i e n, N. H. Q u y n h, C. C. T h o. Machine Learning Based Admission Data Processing for Early Forecasting Students' Learning Outcomes. – International Journal of Data Warehousing and Mining, 2022, pp. 1-15. DOI: 10.4018/IJDWM.313585.
14. S o n, N. T. K., N. H. Q u y n h, B. T. M i n h. Early Prediction Students' Graduation Rank Using LAGT: Enhancing Accuracy with GCN and Transformer on Small Datasets. – J. Comput. Sci. Cybern., Vol. **40**, 2024, No 4, pp. 299-314. DOI: 10.15625/1813-9663/21095.
15. C a r d o n a, T. A., E. A. C u d n e y. Predicting Student Retention Using Support Vector Machines. – Procedia Manufacturing, 2019, pp. 1827-1833. DOI: 10.1016/j.promfg.2020.01.256.
16. M u k h t a r, H., J. A. A m i e n, F. D e w i. Prediction of Student Graduation Using Decision Tree. – CELSciTech, Vol. **5**, 2021, pp. 7-18.
17. Y a q i n, A., M. R a h a r d i, F. F. A b d u l l o h. Accuracy Enhancement of Prediction Method Using SMOTE for Early Prediction Student's Graduation in XYZ University. – International Journal of Advanced Computer Science and Applications, Vol. **13**, 2022, No 6, pp. 418-424. DOI: 10.14569/IJACSA.2022.0130652.
18. P o u d y a l, S., M. J. M o h a m m a d i-A r a g h, J. B a l l. Prediction of Student Academic Performance Using a Hybrid 2D CNN Model. – Electronics, Vol. **11**, 2022, No 7, pp. 1-21. DOI: 10.3390/electronics11071005.
19. K u s u m a w a r d a n i, S. S., S. A. A l f a r o z i. Transformer Encoder Model for Sequential Prediction of Student Performance Based on Their Log Activities. – IEEE Access, Vol. **11**, 2023, pp. 18960-18971. DOI: 10.1109/ACCESS.2023.3246122.
20. Y a n g, G., Y. O u y a n g, Z. Y e, R. G a o, Y. Z e n g. Social-Path Embedding-Based Transformer for Graduation Development Prediction. – Applied Intelligence, Vol. **52**, 2022, pp 14119-14136. DOI: 10.1007/s10489-022-03268-y.
21. M i r z a, M., S. O s i n d e r o. Conditional Generative Adversarial Nets. – arXiv preprint, 2014.

Received: 11.11.2024, First Revision: 02.02.2025, Second Revision: 28.02.2025, Accepted: 07.03.2025