Hindawi
Complexity
Volume 2021, Article ID 5519769, 21 pages
https://doi.org/10.1155/2021/5519769

Research Article

Leveraging the Power of Deep Learning Technique for Creating an Intelligent, Context-Aware, and Adaptive M-Learning Model

Muhammad Adnan,[1] **Duaa H. AlSaeed**,[2] **Heyam H. Al-Baity**,[2] **and Abdur Rehman**[1]

[1]*Institute of Computing, Kohat University of Science and Technology, Kohat 26000, Pakistan*
[2]*Information Technology Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia*

Correspondence should be addressed to Muhammad Adnan; adnankust@gmail.com

Received 16 February 2021; Revised 14 March 2021; Accepted 12 April 2021; Published 14 July 2021

Academic Editor: Furqan Aziz

Machine learning (ML) and deep learning (DL) algorithms work well where future estimations and predictions are required. Particularly, in educational institutions, ML and DL algorithms can help instructors in predicting the learning performance of learners. Furthermore, the prediction of the learning performance of learners can assist instructors and intelligent learning systems (ILSs) in taking preemptive measures (i.e., early engagement or early intervention measures) so that the learning performance of weak learners could be increased thus reducing learners' failures and dropout rates. In this study, we propose an intelligent learning system (ILS) powered by the mobile learning (M-learning) model that predicts learners' performance and classify them into various performance groups. Subsequently, adaptive feedback and support are provided to those learners who struggle in their studies. Four M-learning models were created for different learners considering their learning features (study behavior) and their weight values. The M-learning model was based on the artificial neural network (ANN) algorithm with the aim to predict learners' performance and classify them into five performance groups, whereas the random forest (RF) algorithm was used to determine each feature's importance in the creation of the M-learning model. In the last stage of this study, we performed an early intervention/engagement experiment on those learners who showed weak performance in their study. End-user computing satisfaction (EUCS) model questionnaire was adopted to measure the attitude of learners towards using an ILS. As compared to traditional machine learning algorithms, ANN achieved the highest prediction accuracy for all four learning models, i.e., model 1 = 90.77%, model 2 = 87.69%, model 3 = 83.85%, and model 4 = 80.00%. Moreover, the five most important features that significantly affect the students' final performance were MP3 = 0.34, MP1 = 0.26, MP2 = 0.24, NTAQ = 0.05, and AST = 0.018.

1. Introduction

Despite the numerous advantages of mobile learning (M-learning), the practicality of M-learning had been limited due to not knowing about the exact behavior or learning features of M-learners and how these learning features have different meanings and values for different M-learners [1]. The use of mobile phones for E-learning, U-learning, and M-learning creates additional challenges for the delivery of the adaptive learning content [2]. First, the limited space on a mobile phone screen does not allow the learning content to be presented in the same way as it is presented on web and desktop applications. Moreover, due to low and inconsistent internet speed, mobile learners feel frustration when accessing the learning content during the learning process. Nonetheless, M-learning has several advantages over traditional and web-based learning systems [3]. M-learning-based education has already reached many learners as it

provides the possibility of an adaptation process that is difficult to achieve in traditional classroom settings. Mobile learners (M-learners) using mobile phone technologies can access and study the learning content at any time and from anywhere. In M-learning settings, M-learners have the freedom to choose the learning content they want to learn, unlike traditional classroom settings where learners are bound to the instructor's provided learning content and instructions.

Adaptive learning (AL) is an educational method that uses computer algorithms to orchestrate the interaction with the learner and delivers tailored learning content according to the needs and performance of each learner [4]. In AL systems, learning content is presented according to the unique needs of each learner as indicated by learner responses, interactions, and feedback. Though the primary target of AL systems is education, they can be used in a variety of areas, i.e., professional training, business education, simulation, etc. AL systems have been designed primarily for desktop applications and web applications and recently have been the prime target for mobile learning systems [5].

Providing tailored learning content is one of the challenges that an adaptive learning system faces. To provide needed support and to assist learners at the optimal time, it is necessary to determine the exact learning behavior, the strength, and weaknesses of different learners. The learning features determine the learning behavior of various learners which when collected and processed carefully can help adaptive learning systems to guide learners properly. Machine learning (ML) and deep learning (DL) algorithms can model the learning behavior of learners when the corresponding learning features are properly provided. ML/DL algorithms, when trained adequately, can determine the features that have a significant impact on the performance of learners. Moreover, ML/DL multiclass classification algorithms can classify different learners into performance groups according to their previous interaction and performance.

Intelligent learning system (ILS) integrated with machine learning (ML) techniques can be used to create an M-learning model that properly reflects the learning behavior of individual learners. A carefully developed M-learning model based on proper learning features can predict and identify learners' strengths, weaknesses, preferences, and performance. ML techniques that have been used to develop the M-learning model include *K*-nearest neighbors (KNNs) [6], linear discriminant analysis (LDA) [7], multiclass logistic regression (softmax regression) [8], support vector machine (SVM) [9], decision trees (DTs) [10], random forest (RF) [11], and artificial neural networks (ANNs) [12]. Fundamental steps in developing an M-learning model include defining proper learning features (time, location, preferences, performance, etc.), generating a dataset of learning features, features' analysis, preprocessing, transformation into the appropriate form, choosing proper ML algorithm, training, testing, and reevaluating the M-learning model [13]. When developed and matured, the M-learning model can be used as a part of ILS for intelligent guidance and adaptive support. During the learning process, the M-learning model can recommend additional learning content if they need it or let the learner move ahead if the current topic is already covered. An adaptive M-learning system performs adaptation based on the M-learning model and updates the M-learning model when new facts are driven from M-learners' learning features. Information gathered from the interaction mechanism of M-learning systems is often imprecise, and its interpretation is typically uncertain. Machine learning algorithms are commonly used for learner modeling because of their ability to represent complex feature relationships, feature weights, and combined features' effect on the learning behavior of individual learners in the learning environment [14].

Models based on DL algorithms can focus on the right features by themselves, requiring little intervention from human programmers. ML and DL algorithms have the single most important goal of AI research: allowing computers to understand and model our real world well enough to exhibit something identical to what we human beings call intelligence. Some of the popular DL algorithms are artificial neural network (ANN) [15], convolutional neural network (CNN) [16], recurrent neural network (RNN) [17], self-organizing maps (SOMs) [18], deep Boltzmann machine (DBM) [19], and deep autoencoder [20].

The contributions of this study are as follows:

(1) Creating an M-learning dataset: the dataset includes learning features' instances/records generated during the learning process and logged on an online Firebase cloud. The dataset is used as an input to the ANN for the development of the M-learning model.

(2) M-learners' behavior modeling: we used the M-learners' dataset to model learning in the M-learning environment, i.e., creating the M-learning model. In modeling M-learners, the important step is to determine the effect of independent learning features on dependent features, i.e., M-learners' performance (final grades).

(3) Using the artificial neural network (ANN) for learners' classification: ANN (with hidden layers) is a type of deep neural network or deep learning algorithm that is used for M-learners' classification, important features' identification (learning features), and feature weight determination.

(4) Deployment of the M-learning model on mobile devices: in this step, we deployed the M-learning model on mobile devices of M-learners to determine how accurately the learning performance of M-learners is predicted and how the M-learning model helps M-learners in the learning process and making the right decision.

(5) Performing early intervention/engagement experiment: the objective of this experiment was to determine whether early intervention by ILSs motivates M-learners to improve their performance. The result of this experiment revealed that early intervention could be a very effective technique in encouraging weak learners to improve their learning behavior.

2. Related Work

Recently, Kotsiantis presented a novel case study, where he gathered demographics and learning data from written assignments for the ML regression method to estimate learners' future performance [21]. In this experimental study, Kotsiantis revealed that the education domain offers many interesting applications to ML. A learner model was developed based on the mapping between predictors' features and target feature. Linear regression ML technique was compared to other popular and benchmark ML techniques such as model trees (MT, the counterpart of decision trees (DTs)) [22], artificial neural networks (ANNs) [23], locally weighted linear regression (LWLR) [24], and support vector machine (SVM) [25]. The comparison results revealed that model trees (MTs) give accurate results for the construction of a software support learning tool.

ML techniques used for educational purposes can be divided into classification/regression algorithms, sequential pattern analysis, association rules, clustering, and web information mining [26]. Campbell proposed and developed a learner model using linear regression that demonstrated that learner SAT scores are insignificantly predictive of learner success [27]. With the inclusion of the LMS login feature, the predictive power of the model was tripled. The author demonstrated that the login feature is directly proportional to the level of effort. The more log the learner performed, the more the effort he/she is undertaking. It was revealed that learners with a low or average number of SAT scores could achieve success in the final grade through above-average levels of effort.

One of the popular and simple ML techniques used in educational settings is association rule mining [28]. Like other ML techniques, association rule mining tries to find hidden patterns in the data. Association rule mining finds patterns by analyzing features that occur together and highly correlated features. Rules in association rule mining are not extracted from users' behavior or preference but rather finding relationships between elements in every distinct transaction.

Numerous research studies in the past have been carried out that used unsupervised clustering algorithms to group learners according to similar learning features. Anaya and Boticario used a clustering technique called expectation maximization (EM) that groups similar important features without knowing in advance about the structure of the data [29]. In clustering algorithms, the features are grouped together according to their similarities by applying the Euclidean metric. Learners were grouped by clustering techniques according to their online collaboration in distance learning settings. The results indicated that, for successful collaborative learning, regular and frequent evaluation of team collaboration is necessary. Lin et al. applied two-stage clustering (self-organizing maps and *K*-means) on training data of automobile driving schools [30]. The result of clustering served as a reference point for future training courses. The authors also developed an educational training prediction model using a backpropagation neural network. The prediction model was used to create a knowledge management system that assists the learner in automobile training courses.

Deep learning (DL) is the most trending and discussed research area in the field of AI. DL is a subfield in ML that uses hidden layers in neural network models to represent data both at a granular and abstract level. DL algorithms have small processing units inside hidden layers called neurons or perceptions that can apply linear and nonlinear transformations to the input data. Different DL models and architectures have been proposed, developed, and successfully applied in both supervised and unsupervised problems in the fields of image processing, natural language processing, handwriting recognition, self-driving cars, and computer vision. DL algorithms form a hierarchy of multiple hidden layers where high layers drive data from lower layers. Table 1 lists the popular DL architecture, comparisons with baseline algorithms, evaluation method, and performance score.

3. Proposed Mechanism

Mainly, we have divided our research work into four phases. The first phase is about mobile learning system (Learnit) development, testing, and deployment. The development of the mobile learning system was necessary for feature generation and gathering. Phase two describes how learners used Learnit, the types of learning features, and how learning features were gathered from it. Phase three (discussed in Section 4) gives a complete description of how different machine and deep learning algorithms were applied to learning features for analysis and examination. Phase four (also discussed in Section 4) detailed results generated from the analysis and how results can be used in the adaptive M-learning process. Figure 1 shows the steps carried out during M-learning model development, testing, integration, features' collection, deployment, early intervention, and conducting end-user computing satisfaction (EUCS) experiment.

3.1. Mobile Learning System (Learnit). Figure 2 shows the interaction of learners with different activities of the Learnit app, whereas Figure 3 shows learning, problem posting, and quiz-taking activities. For this study, three courses (computer basics, C++ programming, and JAVA programming) were offered to learners. While using Learnit, a learner can select any course he/she is interested in. Each course is further divided into three learning modules. Each learning module consisted of three learning topics. After the completion of each module, a quiz activity was offered to learners to evaluate their learning. A final quiz/checkpoint was

Table 1: DL algorithm/architecture used in education, baseline algorithms, and evaluation method. The column values indicate whether the DL algorithm outperformed baseline algorithms (greater), underperformed (lesser), or obtained the same results (equal).

Reference	DL model architecture	Baseline ML model architecture	Evaluation method	Performance
Guo et al. [31]	Student performance prediction network (SPPN) based on the sparse autoencoder	Naïve Bayes, MLP, SVM	Accuracy	Greater
Akram et al. [32]	LSTM	SVM, RF, majority class	Accuracy, precision, recall, F_1	Greater
Fok et al. [33]	Deep neural networks	DT, association rules	Accuracy	Greater
Abhinav et al. [34]	MLP	*K*-NN, SVD	RMSE, MAE	Greater
Amazona and Hernandez [35]	ANN	DT, Naïve Bayes	Accuracy	Greater
Alvarado et al. [36]	WE	N-grams	Precision, recall, *F*-measure	Equal
Kim et al. [37]	GritNet, bidirectional long short-term memory (BLSTM)	Standard logistic regression	Performance	Greater
Fei and Yeung et al. [22]	RNN, LSTM	SVM, LogReg, IOHMM	Accuracy	Greater
Samuel-Soma et al. [38]	Ensemble techniques	NB, DT, *K*-NN, disc, PWC	Accuracy, performance	Greater
Alam et al. [39]	Artificial neural network	Random forest, Chi2	Classification, accuracy	Equal
Khajah et al. [40]	LSTM	BKT	Accuracy	Equal
Ma et al. [41]	DNN	DT, SVM	Prediction, accuracy	Lesser
Lalwani and Agrawal et al. [42]	LSTM	PFA, BKT	Accuracy	Equal
El Fouki and Aknin [43]	DNN	PCA	Prediction	—
Abidi et al. [44]	DL	Generalized linear model (GLM), logistic regression (LR), decision tree (DT), random forest (RF), and gradient boosted trees (XGBoost)	Learners' confusion prediction	Greater for RF, GLM, XGBoost, and DL
Hadullo et al. [45]	MLP	—	Performance prediction factors	—
Wang et al. [46]	CNN, RNN	SVM, LogReg, DT, AdaBoost, GTB, RF, GNB	Accuracy, precision, recall, *F*-measure	Equal
Ndukwe et al. [47]	DNN	—	Classification	—
Whitehill et al. [48]	FNN	—	Accuracy	—
Chai et al. [49]	ANN	—	Accuracy	Greater
Yeung and Yeung [50]	LSTM	—	Accuracy	—
Sun et al. [51]	CNN	ANN	Prediction, feature reconstruction	Greater
Tanuar et al. [52]	ANN	Linear model, DT	Accuracy	Greater
Yang et al. [53]	LSTM	—	MSE	—
Dyuti Islam [54]	MLP	RF, SVM	Accuracy, precision, recall, *F*-measure	Greater
Wang et al. [55]	Convolutional GRU	RF, SVM, BPNN, RNN, LSTM	Accuracy, F_1-score	Greater
Saa et al. [56]	NN	RF	Prediction accuracy	Lesser
Adam et al. [57]	LSTM	—	—	—
Sharma et al. [58]	CNN, AlexNet, VGG16, LSTM	SVM, HMM	Accuracy	Greater

administered after the completion of three courses to evaluate the overall learning outcomes of the learners. Learners were provided with two types of learning content (video and PDF text) for every learning topic.

3.2. Features' Extraction. As mentioned earlier, the learners' features were extracted and gathered from learners while they used the Learnit app. Table 2 shows the preprocessed features along with their category and description.

875 learners from the Institute of Computing, Department of Physics, and Institute of Numerical Science participated in using the Learnit app. After data cleansing and security, 850 learners' data were finally selected to be included in the dataset and to be further used as an input to the learner model. These features were further preprocessed and converted into a form that the learner model can accept as an input. The features' preprocessing steps include features' encoding, features' scaling, and splitting features into training and testing sets.

4. ANN-Based M-Learning Model (Modeling Learner Using ANN)

To develop the M-learning model, ANN with two hidden layers was used as shown in Figure 4. For the ANN-based learning model, the most important element is historical feature instances. The more the feature instances are provided to the learner model, the better it will learn and will generate accurate results. In this study, the ANN-based learner model should accurately categorize learners into 5 (A, B, C, D, and F) different groups based on their learning performance.

4.1. Training ANN-Based M-Learning Model. Training the ANN-based M-learning model includes the following steps:

(1) Feeding the M-learning model with feature instances, i.e., feed-forward technique or forward propagation where data flow in one direction, from the input to the output layer.

(2) Selecting the activation function for hidden layer neuron activation. In our experiment, we have selected the ReLU activation function.

(3) Using the activation function at the output layer. As our problem is categorical, therefore, we have used the softmax function.

(4) Backpropagation: to reduce the difference between the actual result and generated result, i.e., to reduce the loss.

Forward propagation is the core process during the model training and learning phase. It is the forward propagation along with the backpropagation technique, where a model learns the weights and bias values for neural network synapses. Our input features' data are in the form of $n * 13$, where 13 is the number of input features and n is feature instances/records. The default configuration of the ANN accepts one feature instance at a time, but multiple feature instances can be entered into the ANN by a technique called matrix multiplication. Using matrix multiplication, multiple feature instances can enter the ANN at a time, speeding up the training and learning process as a result. To perform matrix multiplication, we defined two matrices, X and $W1$, where X represents the input features' data and $W1$ is the weight of the synapse between the input layer and hidden layer 1. Matrix X is in the form $N * M$, where M is the input features and N is the feature instances. Similarly, the $W1$ matrix represents the weights of the synapses between the input layer and hidden layer 1, having $13 * 11$ dimensions, where 13 is the input neurons and 11 represents the values of the weights of the synapses attached to each input neuron (11 synapses per neuron). We used the Python Dense class to initially initialize weights on synapses. Later, synapses' weights get updated according to the error value backpropagated from the output layer. Mathematically, matrices X and $W1$ can be represented as

$$X = \begin{bmatrix} X1_1 & X2_1 & X3_1 & \dots & X13_1 \\ X1_2 & X2_2 & X3_2 & \dots & X13_2 \\ X1_3 & X2_3 & X3_3 & \dots & X13_3 \\ \dots & \dots & \dots & \dots & \dots \\ X1_n & X2_n & X3_n & \dots & X13_n \end{bmatrix},$$
$$W_1 = \begin{bmatrix} W1_1 & W1_2 & W1_3 & \dots & W1_{11} \\ W2_1 & W2_2 & W2_3 & \dots & W2_{11} \\ W3_1 & W3_2 & W3_3 & \dots & W3_{11} \\ \dots & \dots & \dots & \dots & \dots \\ W13_1 & W13_2 & W13_3 & \dots & W13_{11} \end{bmatrix}. \quad (1)$$

Matrix X, when multiplied by matrix $W1$, yields matrix a_1 having $n * 11$ dimensions as illustrated in equation (2). For simplicity, the resultant matrix multiplication process is represented in equation (3).

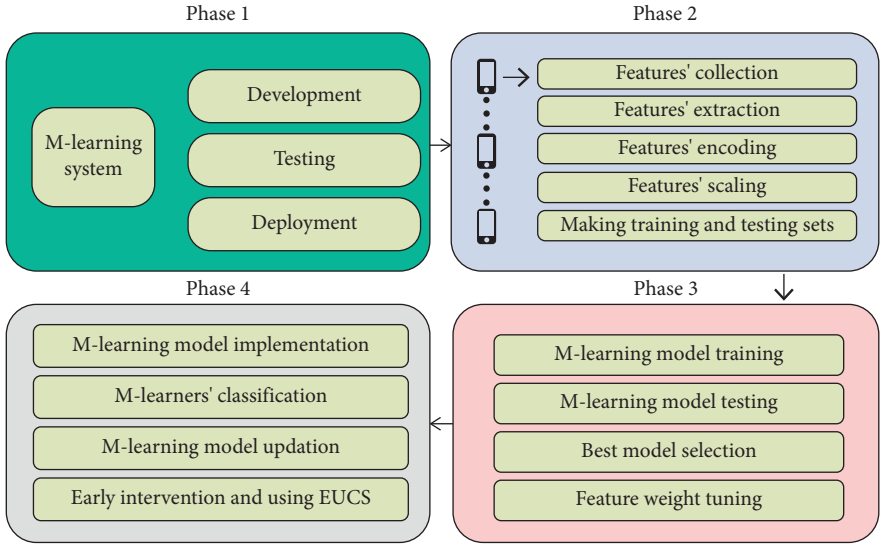

Figure 1: Workflow steps during M-learning model development and deployment.

$$a_1 = \begin{bmatrix} \{X1_1 * W1_1 + \cdots + X13_1 * W13_1\} & \{X1_1 * W1_2 + \cdots + X13_1 * W13_2\} & \ldots & \{X1_1 * W1_{11} + \cdots + X13_1 * W13_{11}\} \\ \{X1_2 * W1_1 + \cdots + X13_2 * W13_1\} & \{X1_2 * W1_2 + \cdots + X13_2 * W13_2\} & \ldots & \{X1_2 * W1_{11} + \cdots + X13_2 * W13_{11}\} \\ \{X1_3 * W1_1 + \cdots + X13_3 * W13_1\} & \{X1_3 * W1_2 + \cdots + X13_3 * W13_2\} & \ldots & \{X1_3 * W1_{11} + \cdots + X13_3 * W13_{11}\} \\ \ldots & \ldots & \ldots & \ldots \\ \{X1_n * W1_1 + \cdots + X13_n * W13_1\} & \{X1_n * W1_2 + \cdots + X13_n * W13_2\} & \ldots & \{X1_n * W1_{11} + \cdots + X13_n * W13_{11}\} \end{bmatrix}, \tag{2}$$

$$a_1 = XW_1. \tag{3}$$

Two processes are performed during ANN forward propagation when input data are transferred from the input layer to hidden layer 1. First, at each neuron, at hidden layer 1, the input feature value is multiplied by synapses' weights. This multiplication task is done at each neuron in hidden layer 1 for all features' data and synapses connected to the corresponding neuron. The multiplication results are then added together, and the activation function is applied to the result. The processing at each neuron in hidden layer 1 can be summarized by the following equation. θ represents the activation function performed after the sum of the product operation.

$$\theta\left(\sum_{k=1}^{11} X_i W_i\right). \tag{4}$$

Secondly, the activation function is performed on the result obtained from the sum of the product operation. We used the rectified linear unit (ReLU) activation function at each hidden layer neuron because it is computationally less expensive, efficient, and can easily propagate errors during the backpropagation process. ReLU activation function is represented in the following equation:

$$z_1 = f(a_1), \tag{5}$$

where $f(a_1)$ is

$$f(a) = \begin{cases} 0, & \text{for } a_1 < 0, \\ a_1, & \text{for } a_1 > = 0. \end{cases} \tag{6}$$

The ReLU activation function is applied independently to each entry in matrix a_1, and a new matrix is generated called z_1. The size of matrix z_1 is the same as that of matrix a_1. All the neurons at hidden layer 1 perform the ReLU activation function on their input matrix and generate modified matrices ready for feed-forwarding to hidden layer 2. The matrix data are propagated forward from hidden layer 1 to hidden layer 2 in the same way as it was propagated from the input layer to hidden layer 1. Similarly, the operation at hidden layer 2 neurons is the same as hidden layer 1 neurons' operation. At hidden layer 2 neurons, first, the input values from hidden layer 1 are multiplied by corresponding synapses' weights and then added together to perform the ReLU activation function. The z_1 matrix stored at hidden layer 1 has an $n * 11$ dimension, whereas the dimension of the W_2 matrix is $11 * 11$, where 11 represents neurons at hidden layer 1 and their corresponding weights. The multiplication of the z_1 matrix with W_2 yields the a_2 matrix having the dimension of $n * 11$ and can be represented by the following equation:

(a) (b) (c)

(d) (e) (f)

Figure 2: Learner interaction with the Learnit app. (a) Learner login activity. (b) Learner registration activity. (c) Course selection activity. (d) Computer basics topics. (e) C++ topics. (f) JAVA topics.

$$a_2 = z_1 W_2. \tag{7}$$

After the matrix multiplication operation, the ReLU activation function is performed on each entry in matrix a_2 resulting in a new matrix called z_2. Mathematically, the equation can be written as

$$z_2 = f(a_2). \tag{8}$$

Similarly, using the forward propagation technique, matrix z_2 at hidden layer 2 is propagated to the output layer. The neurons at the output layer, after performing softmax activation function (for categorical/multiple classification results) on input data, generate $\hat{p}_j$, which is the ANN-based M-learning model-predicted value of the learner's final grade (FG). It is very likely that initially, there is a difference between the predicted result $\hat{p}_j$ and actual result "a." For

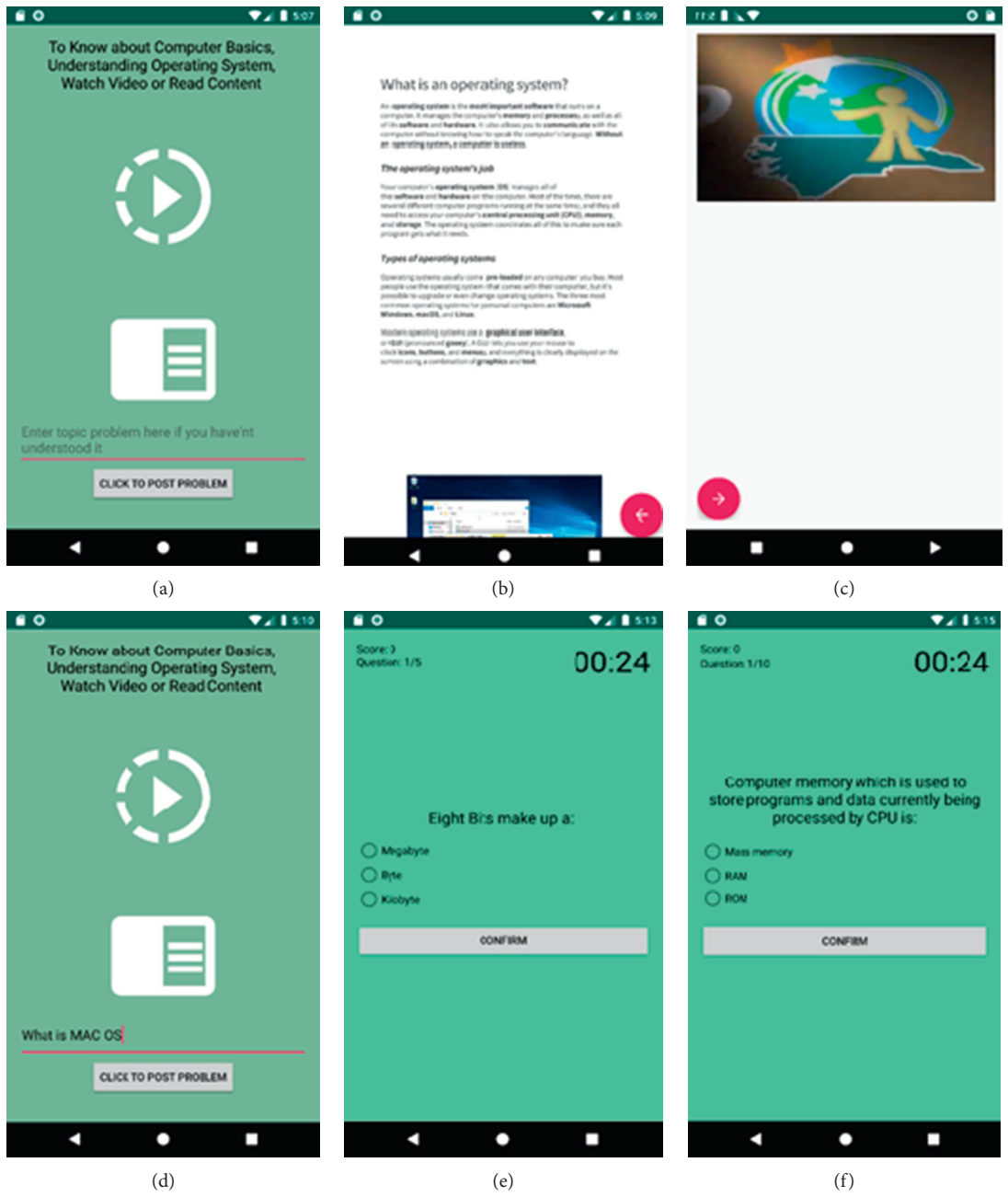

Figure 3: Learner learning, problem posting, and quiz activities. (a) Video or text selection activity. (b) Learner reading text content. (c) Learner watching video content. (d) Problem posting related to a particular topic. (e) Module quiz activity. (f) Final quiz activity.

minimizing the difference between the predicted result and the actual result, we used a technique called back-propagation, which is further explained in the following section.

4.1.1. Minimizing Model Cost/Loss Using Backpropagation. M-learning model cost or loss function tells us the difference between the predicted and actual output result. Ideally, the predicted result should be equal or very near to the actual result, but in a scenario where the difference is substantial, the backpropagation technique is used to reduce the difference between predicted and actual output results. We expressed the learner model cost function in the following expression:

$$C = \sum \frac{1}{2}\left(a - \hat{p}_j\right). \tag{9}$$

In the above expression, "a" is the learner's actual final grade, whereas $\hat{p}_j$ is the predicted final grade result. For an accurate and reliable learner model, this difference must be as minimum as possible. As a result, the M-learning model will convey an accurate picture of learners' learning behavior and performance. For minimizing the learner model cost function, we had two choices: (1) changing the input

features' data and (2) tweaking weights on ANN synapses. Input feature instances are static and will not change; therefore, the only option left with us is to adjust weights on ANN synapses. To determine suitable weights for synapses, we used a technique called stochastic gradient descent (SGD) along with backpropagation. SGD technique is efficient and reduces time and computation costs during the backpropagation process. Furthermore, when we have high-dimensional data and we are looking for suitable weights across millions of synapses, the SGD technique is more impressive in determining suitable synapses' weights and therefore reducing the overall model cost. Looking at our ANN model architecture, we can observe that the weights that are responsible for generating cost/error are spread across three matrices, i.e., W_1, W_2, and W_3. We calculated the derivative of cost C concerning W_1, W_2, and W_3, i.e., $(\mathrm{d}(C)/\mathrm{d}(W_1))$, $(\mathrm{d}(C)/\mathrm{d}(W_2))$, and $(\mathrm{d}(C)/\mathrm{d}(W_3))$, resulting in the same number of gradient values as the number of synapses' weights across the ANN. The derivative of C concerning W_1, W_2, and W_3 determines how many weight values are responsible for generating error/cost in the M-learning model.

The key in the training of the M-learning model is to find the appropriate values for weights across the model so that the model can build good approximator functions that can be applied to the new learner features for their performance prediction. In other words, we want to have a set of weights that minimize the error/loss/cost at the output layer. To demonstrate the backpropagation process in detail, consider the simplified M-learning model in Figure 5.

If the weights W_1 get changed, the values across hidden layers 1 and 2, W_1, W_2, and the output result would also change. Here, the output result is functional dependent on W_1, W_2, and W_3 weights and activation function in hidden layers 1 and 2. We can, mathematically, formulate the output as an extension of the composite function:

$$\text{output} = \text{act}\left(w_3 * \text{act}\left(w_2 * \text{act}\left(w_1 * \text{input\_data}\right)\right)\right). \quad (10)$$

If we have an error on the output layer, then it means that the error is functionally dependent on the output function which in turn is dependent on the function of input, weights, and activation functions. To know how many arbitrary weights, i.e., W_1, are responsible for the generated error, we compute the derivative of the error w.r.t w_1, and the result would be

$$\frac{\mathrm{d}(j)}{\mathrm{d}(W_1)} = \frac{\mathrm{d}(j)}{\mathrm{d}(\text{output})}.\frac{\mathrm{d}(\text{output})}{\mathrm{d}(\text{hidden2})}.\frac{\mathrm{d}(\text{hidden2})}{\mathrm{d}(\text{hidden1})}.\frac{\mathrm{d}(\text{hidden1})}{\mathrm{d}(W_1)}. \quad (11)$$

In the backpropagation process, we are traversing from the output error to W_1, taking iterative steps using the chain rule. We take the derivative of the error w.r.t the output, derivative of the output w.r.t to hidden neurons, and finally derivative of hidden neurons w.r.t weights. This process is shown in Figure 6.

To get a comprehensive intuition of the backpropagation process, we are going to determine the derivative of the error w.r.t the weights at the input layer denoted by W_{ij}^1 in Figure 7. The architecture of the learner model in Figure 7 is simplified for a better understanding of the backpropagation process. Mathematically, we are trying to obtain how many input layer weights are responsible for the error as shown in the following equation:

$$\frac{\mathrm{d}(\text{error})}{\mathrm{d}\left(W_{ij}^1\right)} = \frac{\mathrm{d}(j)}{\mathrm{d}\left(W_{ij}^1\right)}. \quad (12)$$

Looking at Figure 7, we also know that

$$\frac{\mathrm{d}(\text{error})}{\mathrm{d}\left(W_{ij}^1\right)} = \frac{\mathrm{d}(j)}{\mathrm{d}\left(W_{ij}^1\right)} = \frac{\mathrm{d}(j)\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(p_j\right)\mathrm{d}\left(W_{ij}^1\right)}. \quad (13)$$

This implies that the derivative of the output error w.r.t to input layer weights is equal to the derivative of the error w.r.t the predicted result multiplied by the derivative of the predicted result w.r.t input layer weights. The term $(\mathrm{d}(j)/\mathrm{d}(p_j))$ tells us the derivative of the error w.r.t output prediction which can be written as $(P_j - a)$ where a is the actual output result.

Therefore, the equation becomes

$$\frac{\mathrm{d}(j)}{\mathrm{d}\left(W_{ij}^1\right)} = \left(P_j - a\right)\frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(W_{ij}^1\right)}. \quad (14)$$

Going one layer further back and applying the chain rule, we can write $(\mathrm{d}(p_j)/W_{ij}^1)$ as

$$\frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(W_{ij}^1\right)} = \frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(p_j\right)}\frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(W_{ij}^1\right)}. \quad (15)$$

From equation (15), we can deduce that the derivative of the predicted result w.r.t input weights is equal to the derivative of the output w.r.t the derivative of the input (p_i) at the output layer multiplied by the derivative of the input value w.r.t input weights. Similarly, applying the chain rule, $(\mathrm{d}(p_j)/\mathrm{d}(p_j))$ can be written as

$$\frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(p_j\right)} = p_j\left(1 - p_j\right). \quad (16)$$

In the above expression, the term at the right-hand side corresponds to the derivative of the ReLU function which is equal to the output result multiplied by one minus the output. P_j denotes the output of the activation function at the output layer. Now, equation (15) can be written as

$$\begin{aligned}\frac{\mathrm{d}\left(p_j\right)}{\mathrm{d}\left(W_{ij}^1\right)} &= p_j\left(1 - p_j\right)\frac{\mathrm{d}\left(p_i\right)}{\mathrm{d}\left(W_{ij}^1\right)},\\ \frac{\mathrm{d}(j)}{\mathrm{d}\left(W_{ij}^1\right)} &= \left(p_j - 1\right)^* p_j\left(1 - p_j\right)\frac{\mathrm{d}\left(p_i\right)}{\mathrm{d}\left(W_{ij}^1\right)}.\end{aligned} \quad (17)$$

Moving backward to hidden layer 2, we can notice that the red lines spread out in multiple directions. Since multiple weight values affect the value of p_i, we must take all of them into derivative which can be represented by sigma notation:

Table 2: Preprocessed features along with their category and description.

Features' categories	Features	Description
Behavioral features	ODGP	Online discussion group participation, i.e., the number of times a learner has participated in the online discussion group (numeric)
	NPP	Number of times problem posted online related to study topics (numeric)
	NPS	Number of times solved the posted problem online by other learners during learning (numeric)
	NTAQ	Total number of quiz attempts in three modules (numeric)
	TRR	Learning topic average repetition rate in three modules (numeric)
Context features	AST	Average study time per learning topic (numeric)
	NTRA	Number of times text resource accessed during the learning process (numeric)
	NVRA	Number of times video resource accessed during the learning process (numeric)
	MP1	Learner performance in module 1, numeric: (18 to 20 = very good), (15 to 18 = good), (12 to 15 = average), (9 to 12 = satisfactory), and (0 to 9 = fail)
	MP2	Learner performance in module 2, numeric: (18 to 20 = very good), (15 to 18 = good), (12 to 15 = average), (9 to 12 = satisfactory), and (0 to 9 = fail)
	MP3	Learner performance in module 3, numeric: (18 to 20 = very good), (15 to 18 = good), (12 to 15 = average), (9 to 12 = satisfactory), and (0 to 9 = fail)
	APV	Academic places visited. Degree of academic places visited during the learning process, i.e., library and classrooms, numeric: 0 to 1
	SPV	Social places visited. Degree of social places visited during the learning process, i.e., playgrounds and hostels, numeric: 0 to 1
Final performance feature	FG	Final grades showing the final performance of a learner in three modules, categorical: (18 to 20 = very good), (15 to 18 = good), (12 to 15 = average), (9 to 12 = satisfactory), and (0 to 9 = fail)

$$\frac{d(p_j)}{d(W_{ij}^1)} = \sum_j \frac{d(p_i)}{d(z_j)} \frac{d(z_j)}{d(W_{ij}^1)}. \tag{18}$$

Looking at Figure 7, we can observe that p_i is the summation of hidden layer 2 weights multiplied by z_j (activation performed on the sum of products). Taking the derivative of pi with z_j would give us connecting weights between hidden layer 2 and the output layer and can be expressed as

$$\frac{d(p_i)}{d(z_j)} = W_{ij}^3. \tag{19}$$

We can express the rate of change of p_i and z_j w.r.t W_{ij}^1 as

$$\frac{d(p_j)}{d(W_{ij}^1)} = W_{ij}^3 \frac{d(z_j)}{d(W_{ij}^1)}. \tag{20}$$

Since we cannot quite put the derivative of z_j w.r.t W_{ij}^i into a numerical form, we continue to use the chain rule and follow the fourth red line back in Figure 7 to determine that

$$\frac{d(z_j)}{d(W_{ij}^1)} = \frac{d(z_j)}{d(z_i)} \frac{d(z_i)}{d(W_{ij}^1)} = z_j(1 - z_j) \frac{d(z_i)}{d(W_{ij}^1)}. \tag{21}$$

Since z_j is the result of applying the activation function to z_i, therefore, we use the same logic as in the previous layer and apply the ReLU derivative. The derivative of z_i w.r.t W_{ij}^1 established by the fifth line back is based on the same idea of spreading out and taking the summation of all contributions:

$$\frac{d(z_i)}{d(W_{ij}^1)} = \sum_j \frac{d(z_i)}{d(y_j)} \frac{d(y_j)}{d(W_{ij}^1)} = \sum_j W_{ij}^2 \frac{d(y_j)}{d(W_{ij}^1)}. \tag{22}$$

We move on to performing more reduction and find the derivative of y_j w.r.t W_{ij}^1.

$$\frac{d(y_j)}{d(W_{ij}^1)} = \frac{d(y_j)}{d(y_j)} \frac{d(y_i)}{d(W_{ij}^1)} = y_j(1 - y_j) \frac{d(y_i)}{d(W_{ij}^1)}. \tag{23}$$

Traversing further back in our network, we notice that, at red line number six in Figure 7, we still must deal with another activation function. The last line in the backpropagation process traverses from the input at y_i to x_j which represent the weights W_{ij}^1 between the input and hidden layer 1. In fact, W_{ij}^1 are the weights which we were attempting to backprop to. At this point, we can determine the change in y_i w.r.t W_{ij}^1 which can be expressed by the following equation:

$$\frac{d(y_i)}{d(W_{ij}^1)} = \sum_j x_j. \tag{24}$$

Notice here that the coefficient of W_{ij}^1 is x_j. Combining all the individual expressions gives us the final expression. Here, the final expression tells us how many weights to adjust at each layer to reduce to the generated error.

$$\frac{d(j)}{d(W_{ij}^1)} = (p_j - a) * (p_j(1 - (p_j))) * \sum_j \left(W_{ij}^3 * (z_j(1 - (z_j))) * \sum_j \left(W_{ij}^2 * (y_j(1 - (y_j))) * \sum_j x_j \right) \right). \tag{25}$$

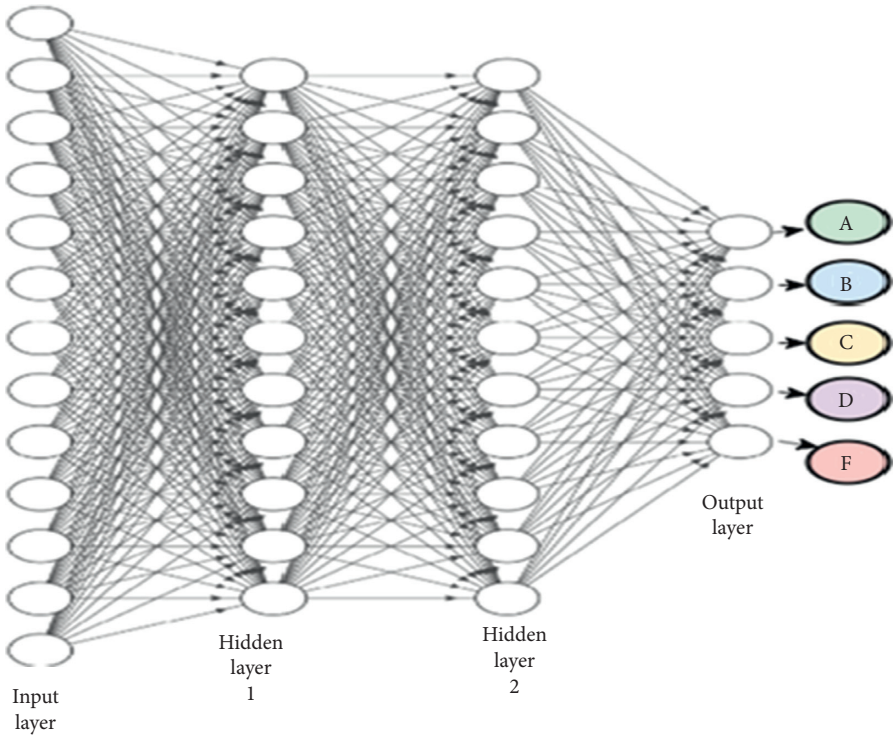

Figure 4: Artificial neural network-based M-learning model with 2 hidden layers, 1 input layer, and 1 output layer.

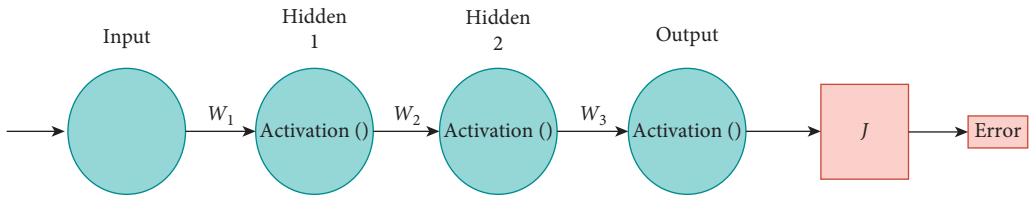

Figure 5: The simplified M-learning model learning process.

4.2. Implementation of the M-Learning Model Using the Artificial Neural Network (ANN). We developed and implemented the M-learning model using the ANN. Firstly, we initialized the weights of synapses with numbers very close to zero with the help of the Keras Dense class. Secondly, the Keras Sequential class was used to propagate dataset features' observations from the ANN input layer to the output layer. During forward propagation, the activation function at every neuron in the ANN is restricted by the weights of the synapses. The output of the M-learning model is learners' performance grades, i.e., A, B, C, D, and F. Table 3 shows learners' performance scores classified into five-level grades.

We assumed that features such as module performance 1 (MP1), module performance 2 (MP2), and module performance 3 (MP3) have the highest impact on the learners' final grades; therefore, we performed multiclass classification on four learning models as discussed in the following briefly:

(1) M-learning model 1: this model contains all the features from the dataset except final grades

(2) M-learning model 2: this model is similar to M-learning model 1 but without the MP3 feature

(3) M-learning model 3: this model is similar to M-learning model 2 but without the MP2 feature

(4) M-learning model 4: this model is similar to M-learning model 3 but without the MP1 feature

We compiled M-learning model 1 and achieved an accuracy of 90.77 percent. To better analyze and visualize the results of M-learning model 1, we constructed a confusion matrix on actual grade scores and predicted grade scores as shown in Figure 8 (without normalization and normalized confusion matrices). The confusion matrix for M-learning model 1 revealed that 107 out of 130 grades were predicted correctly, whereas the remaining 23 grades were predicted incorrectly.

The confusion matrices (without normalization and normalized) for M-learning models 2, 3, and 4 are shown in Figures 9–11. The diagonal elements in confusion matrices are those elements that are predicted accurately, whereas the remaining elements are predicted inaccurately. Those

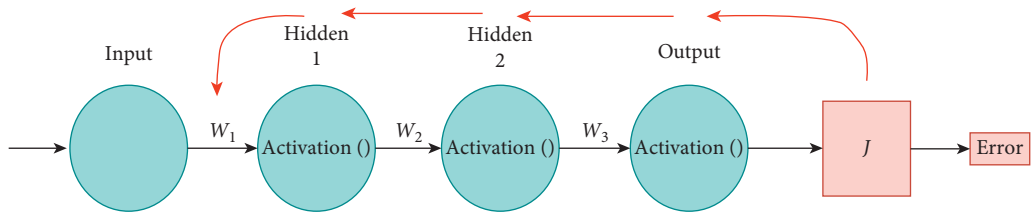

Figure 6: Backpropagation process where the generated error is backpropagated to weights W_1.

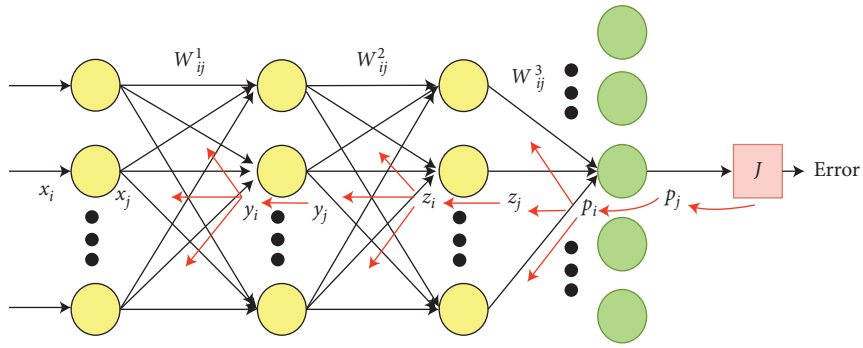

Figure 7: Simplified ANN-based M-learning model architecture.

Table 3: Learners' performance scores classified into five-level grades.

Final grades' classification	Very good	Good	Average	Satisfactory	Fail
Earned points	18–20[a]	15–17	12–14	9–11	0–8
Grades	A	B	C	D	F

[a]The number is inclusive in the range.

elements that are predicted accurately are called true positives, whereas those elements that are predicted inaccurately are called false positives.

From confusion matrices in Figures 9–11, we can observe that the accuracy of models decreases as we remove the modules' performance, i.e., features MP1, MP2, and MP3 from independent features. This concludes that module performance scores are an important factor in predicting the final grades of learners. Moreover, to know whether our M-learning models suffer from overfitting, we plot training accuracy, testing accuracy, training loss, and testing loss over the number of epochs passed.

Figures 12–15 show the model accuracy and model loss for M-learning models 1–4. In Figure 12, initially, during early epochs, the model accuracy of the training and testing dataset somewhat matched up, but later on, during the last 50 epochs, the model accuracy of the training and testing dataset somewhat deviates, but overall, it does not seem that overfitting is a huge problem for M-learning model 1, and the accuracy of the model is acceptable. The model loss of M-learning model 1 decreases as the number of epochs passes away.

This concludes that M-learning model 1 is successful in minimizing cost/loss function over time and is gradually learning the exact relationship between independent features and final grades. The results of the model accuracy and the model loss for M-learning model 2 and M-learning model 3 are somewhat similar to M-learning model 1 as shown in Figures 13 and 14, but we can observe that the case for model accuracy and model loss for M-learning model 4 is different. In Figure 15, we can perceive that, as more and more epochs are completed, the difference between the model accuracy and model loss of the training set and testing set increases.

This is because now, due to the absence of module performance features (MP1, MP2, and MP3), M-learning model 4 is not able to predict the final grades, and the model loss increases and the model accuracy decreases on the testing set. The results in Figure 15 also divulged that module performance features are important in predicting the overall final grades of learners.

4.3. Multiclass Benchmark Machine Learning Algorithms' Prediction Accuracy in Comparison with the ANN. Before fitting all four learning models to different machine learning algorithms, some preprocessing was performed. The parameters adopted for artificial neural networks were 200 epochs and a batch size of 5, whereas a total sample of 200 trees was selected for the random forest ensemble method. For support vector machines, we selected sequential minimal optimization (SMO) algorithm. To estimate the predictive accuracy of each algorithm, the 5-fold cross-validation resampling technique was used. Under such a

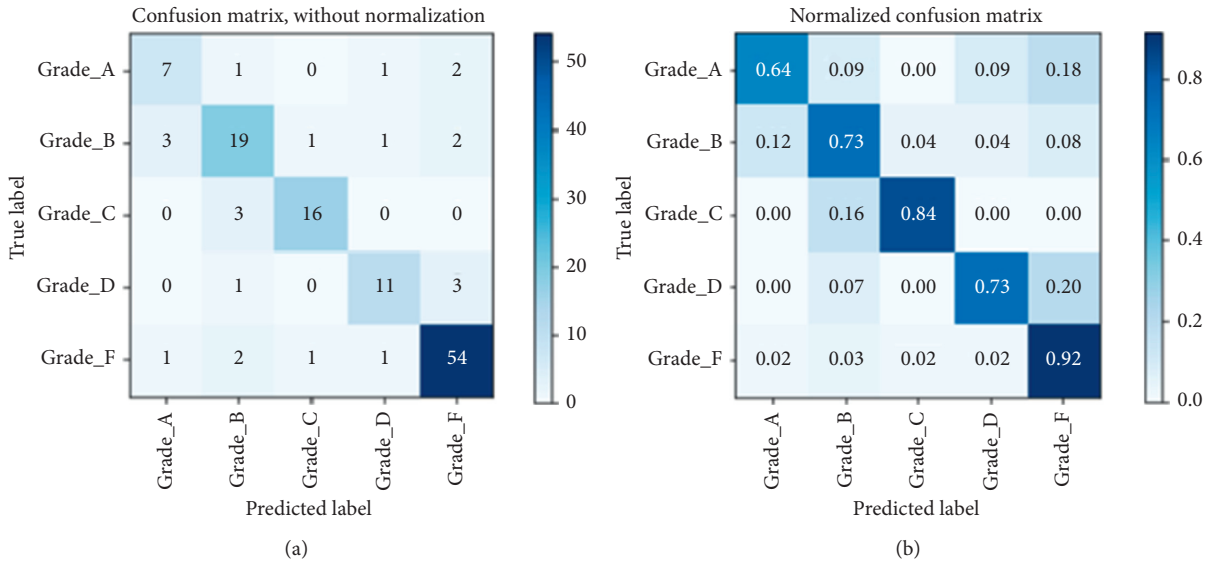

FIGURE 8: Confusion matrices of M-learning model 1, without normalization and normalized confusion matrix. (a) Without normalization. (b) With normalization.

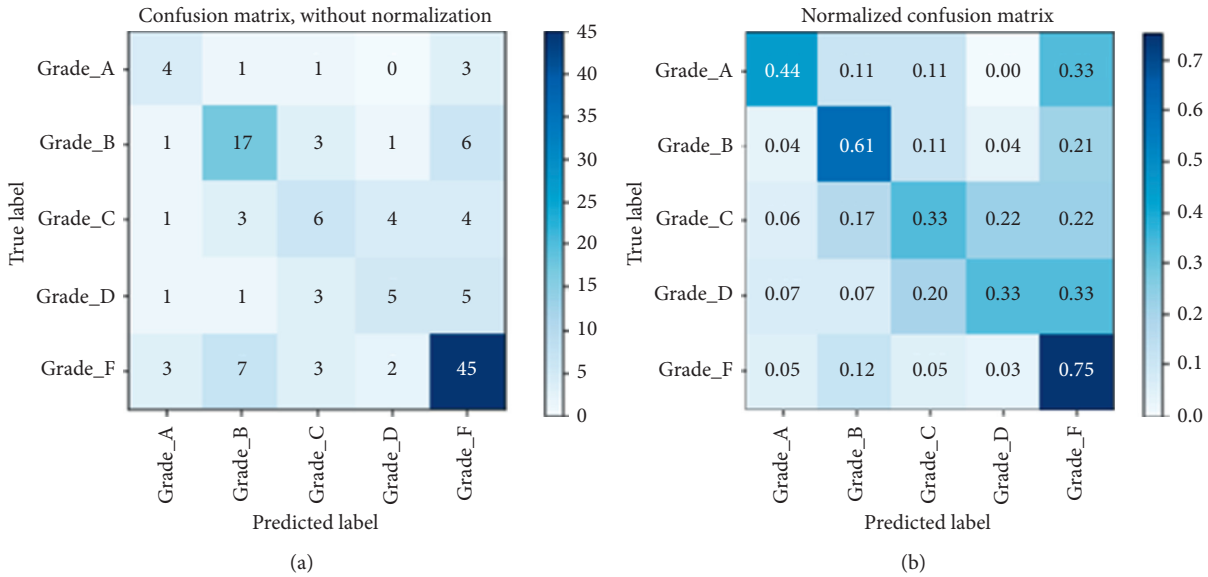

FIGURE 9: Confusion matrices of M-learning model 2, without normalization and normalized confusion matrix. (a) Without normalization. (b) With normalization.

scheme, the data are randomly divided into 5 subsets of equal size. Subsequently, out of 5 subsets, one set is used as the validation data for testing the model, while the remaining 4 subsets are used as a training set. From Table 4, it can be noted that ANN and random forest algorithms have superior prediction accuracy for all four learning models, while linear discriminant analysis exhibited inferior accuracy.

4.4. Features' Weights and Importance. We used the random forest ensemble method to determine each feature's importance and weight in predicting the final performance grades. Random forest is based on an adaptive learning methodology that combines several models to generate better results than the individual model result. Feature weights and importance give good evidence of what features contribute most in predicting the final grades of learners.

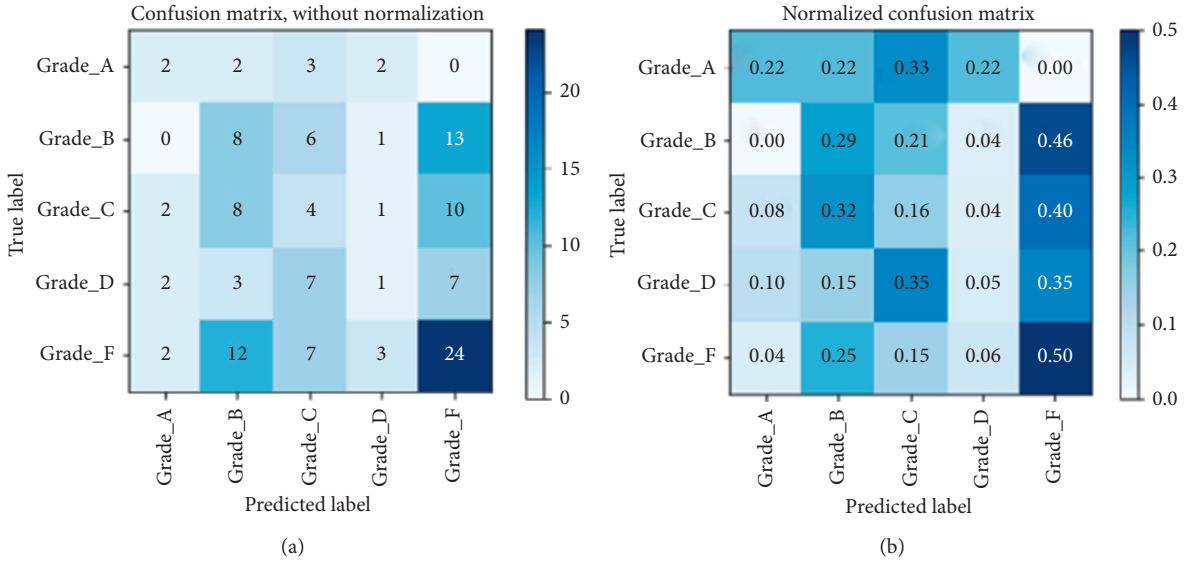

Figure 10: Confusion matrices of M-learning model 3, without normalization and normalized confusion matrix. (a) Without normalization. (b) With normalization.

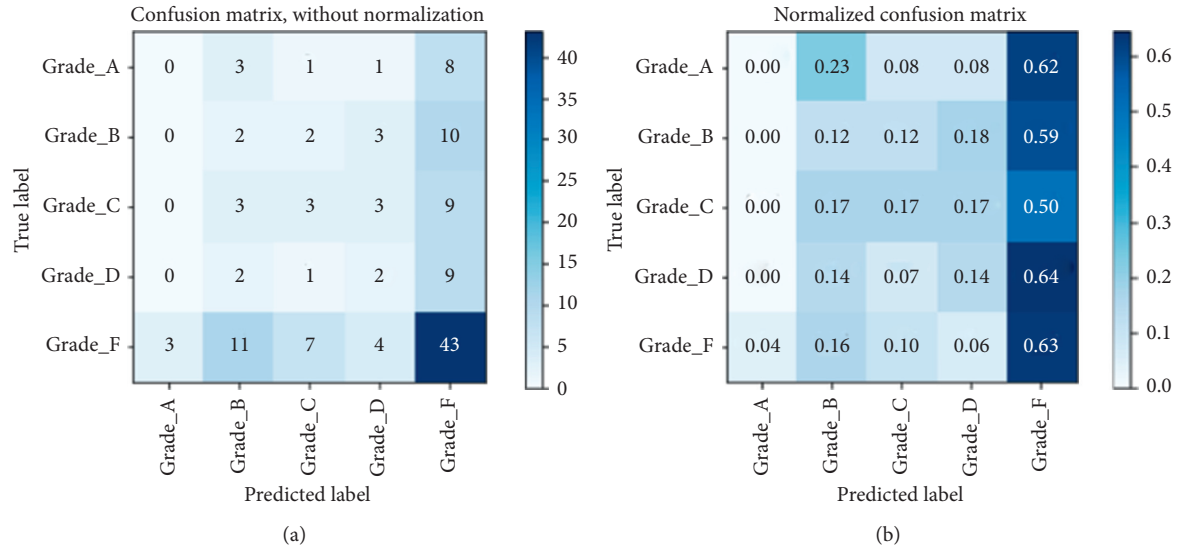

Figure 11: Confusion matrices of M-learning model 4, without normalization and normalized confusion matrix. (a) Without normalization. (b) With normalization.

The features' weights were integrated with the M-learning model to provide adaptive content and tailored navigation to learners during their interaction with the Learnit app. Table 5 shows the values of the weights of each feature in predicting the final grades.

5. Early Intervention Experiment during the Learning Process

In the last stage of our research, we conducted an early intervention experiment to figure out whether preventive measures have an influence on learners in improving their study performance. The experiment was carried out on those learners who achieved performance grades D and F (learners having 11 or less than 11 scores in their final performance quiz). At this point, our M-learning model is trained with learners' features and is matured enough to predict the learning performance of the existing and new learners. The goal of carrying out an early intervention experiment was to motivate and encourage weak learners to improve their learning performance and make M-learning model predictions false. A total of 425 learners (having D and F grades) were selected and were divided into control and experimental groups (the control group having 212 learners

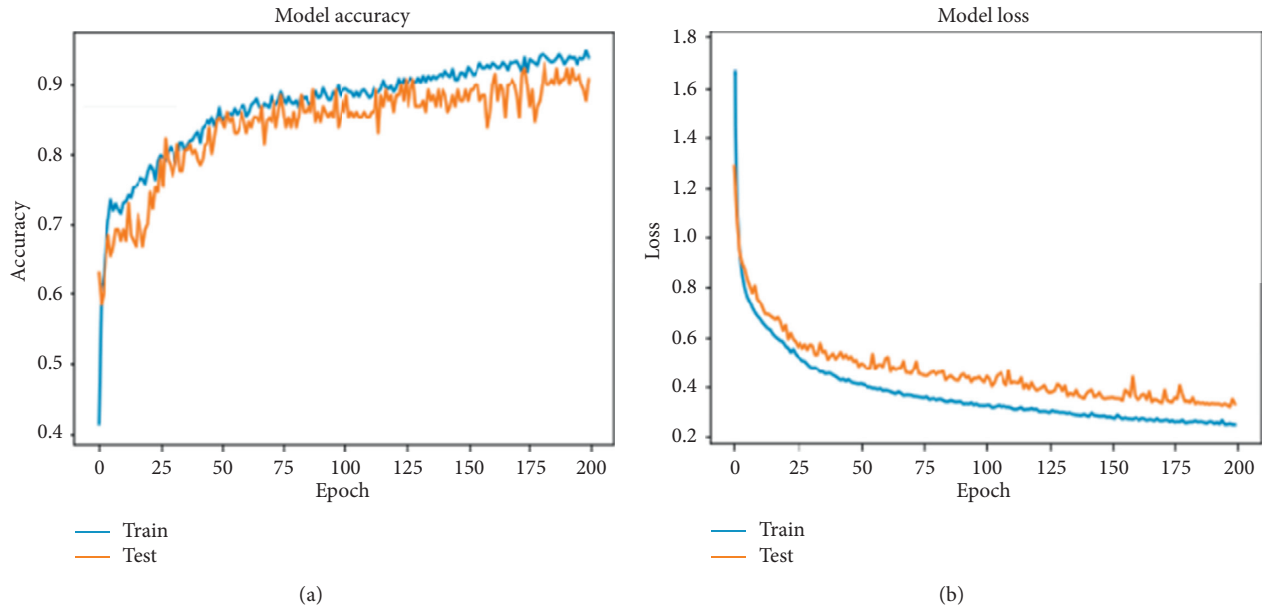

Figure 12: Model accuracy and model loss for M-learning model 1. (a) Model 1 accuracy. (b) Model 1 loss.

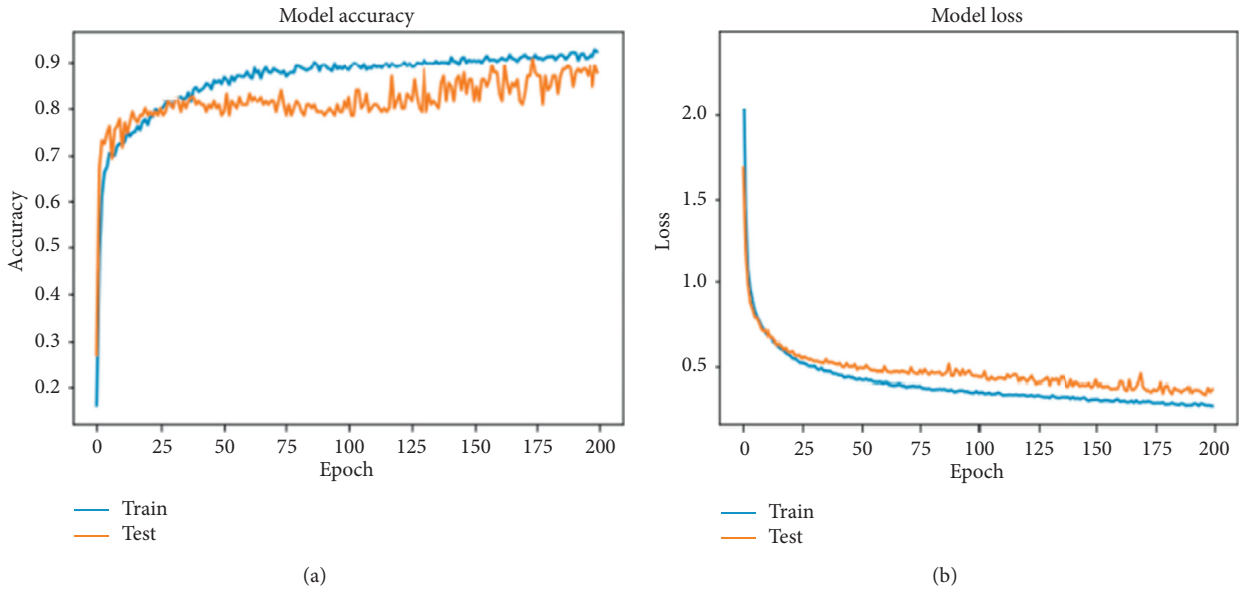

Figure 13: Model accuracy and model loss for M-learning model 2. (a) Model 2 accuracy. (b) Model 2 loss.

whereas the experimental group having 213 learners). Control group learners were independent of preventive measures and early intervention and received normal learning content, guidance, and quizzes, whereas experimental group learners received motivational triggers, adaptive triggers, and adaptive learning content. Moreover, adaptive navigational paths were recommended to experimental group learners during their learning process. The early intervention experiment lasted for 50 days.

During the early intervention experiment, different adaptive triggers were presented to experimental group learners during their learning process. The adaptive triggers were presented according to learners' preferred learning content types and how adaptive triggers are assigned to the learning content.

The timings of the adaptive triggers are before and after. Before triggers refer to those triggers that are presented to learners before they have started the learning activity, whereas after triggers mean presenting triggers to learners after they have finished a learning activity. The purpose of adaptive triggers was to assist learners during their learning process and help them in understanding exhaustive and difficult topics. Likewise, different motivational triggers were presented to learners on their smartphones according to

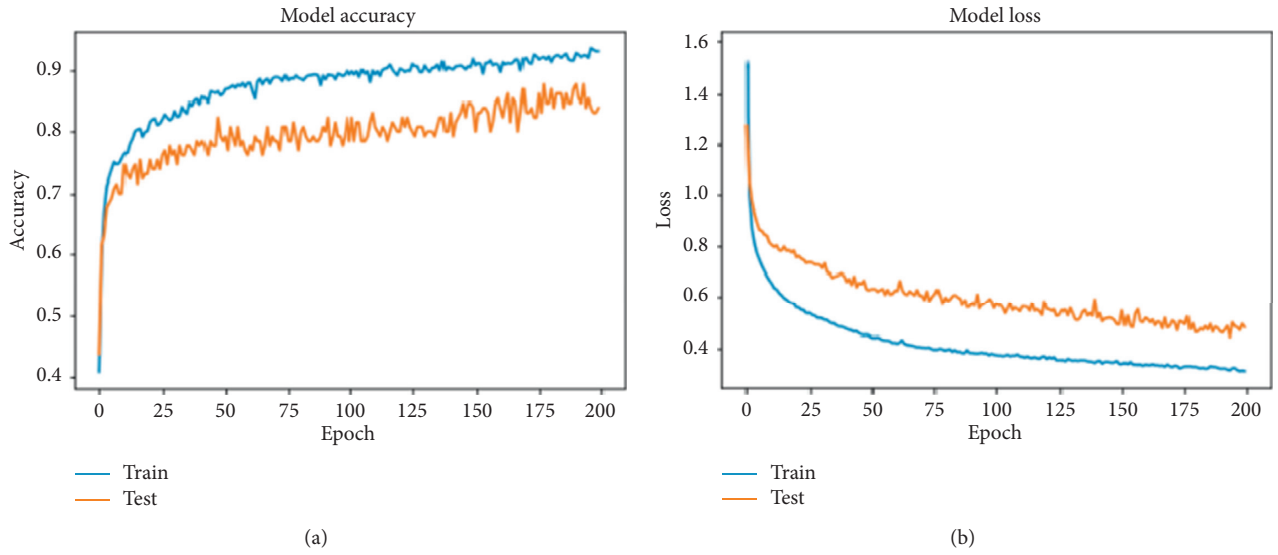

Figure 14: Model accuracy and model loss for M-learning model 3. (a) Model 3 accuracy. (b) Model 3 loss.

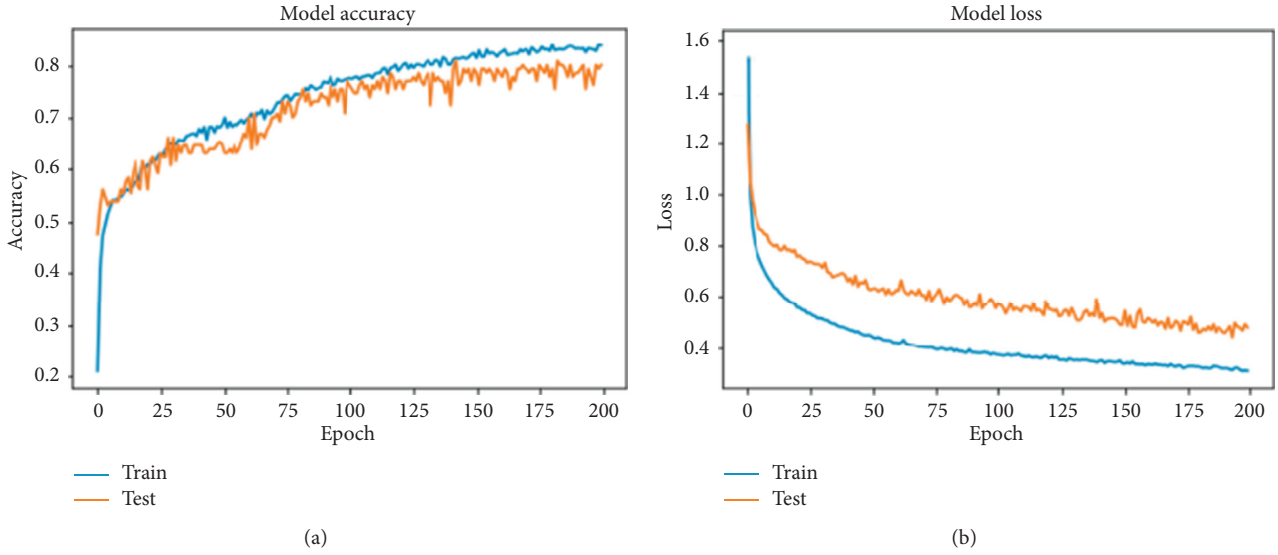

Figure 15: Model accuracy and model loss for M-learning model 4. (a) Model 4 accuracy. (b) Model 4 loss.

their performance scores. Table 6 shows learners' performance state, the assigned motivational triggers, and their examples.

BJ Fogg in his Fogg behavior model (FBM) suggests that three elements, i.e., ability, motivation, and the trigger (prompt or message), must be present at the same time for a learning behavior to occur [59]. Adaptive triggers aimed to help learners while they study, whereas motivational triggers aimed to increase learners' motivation towards learning. One of the important factors related to adaptive and motivational triggers is the opportune time. Opportune time is the right time at which adaptive and motivational triggers are sent to the learners on their smartphones. An example of sending motivational triggers at the opportune time is right after the learner quiz activity in case the performance of the learner in that quiz is not good. Similarly, praise and appreciation triggers were sent to the good consistent learners once a week as too many motivational triggers may overwhelm or annoy learners during the learning process.

6. Early Intervention Experiment Results

After 50 days, we analyzed the final score results of the experimental and control group. Figure 16 shows a line graph representing the final score of both the experimental and control group. The line graph confirms that the intervened learners showed overall better performance than unintervened learners in the final quiz. Moreover, the statistical analysis result revealed that intervened learners showed an overall 12.30% higher performance than

Table 4: Accuracy results of machine learning algorithms and ANN on five-level grades' classification.

Models/ algorithms	Linear discriminant analysis (LDA)	Multiclass logistic regression (softmax regression)	Support vector machine (SVM)	Decision tree (DT)	Random forest (RF)	Artificial neural network (ANN)
Learning model 1	85.56	88.67	87.56	88.56	90.11	90.77
Learning model 2	78.55	79.56	79.44	76.59	86.89	87.69
Learning model 3	67.78	70.45	74.45	67.43	83.67	83.85
Learning model 4	54.12	55.98	59.64	56.89	80.23	80.00

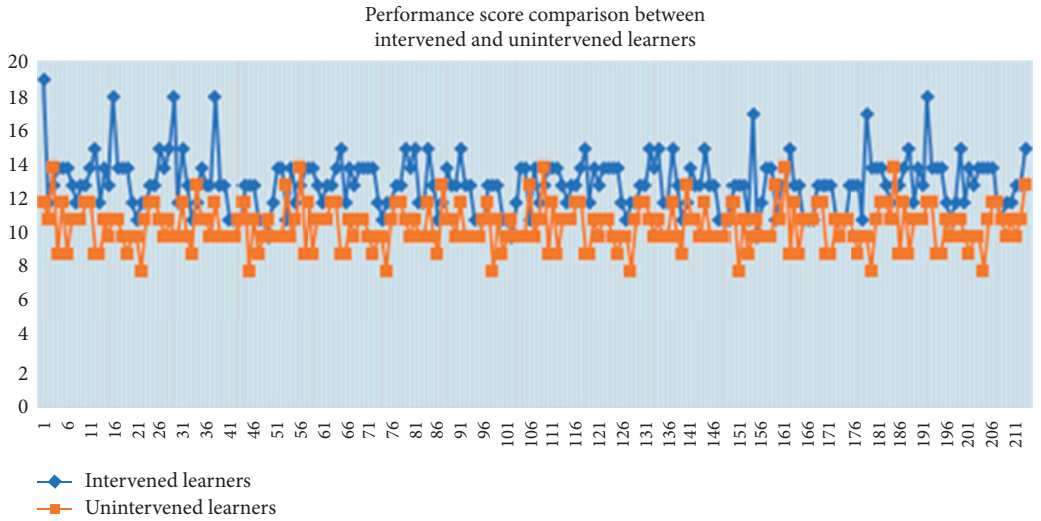

Figure 16: Performance comparison of the experimental and control groups after the early intervention experiment.

Table 5: Independent feature weights in predicting the final grades.

Feature	Feature weight	Feature	Feature weight
MP3	0.345631	NPS	0.010292
MP1	0.264391	NTRA	0.009293
MP2	0.246556	APV	0.009143
NTAQ	0.051794	SPV	0.007157
AST	0.018429	TRR	0.006407
ODGP	0.013156	NVRA	0.005492
NPP	0.012259		

unintervened learners. These results confirm that proactive or preemptive measures do affect learners' behavior and can improve their learning performance scores.

6.1. Analyzing Learners' Satisfaction Using the EUCS Model. To elicit learners' experience of using the Learnit application, end-user computing satisfaction (EUCS) model was used. As discussed in the proposed system section, the Learnit application is based on the M-learning model which uses an artificial neural network (ANN) to determine how independent learning features affect the final learning performance of learners. The EUCS model is composed of a questionnaire that can be used to measure the attitude of users towards using a computer application. With little modification in the EUCS questionnaire, an online survey was steered on 213 experimental group learners. Six dimensions of the Learnit application, namely, usefulness, ease of use, engaging, timeliness, adaptiveness, and attitude towards the Learnit application, were measured by the EUCS questionnaire. The online survey comprised 12 questions covering the 6 dimensions of the Learnit application. The 6 dimensions were measured on the 5-point Likert scale, where 5 represents "strongly agree," 4 represents "agree," 3 represents "neutral," 2 represents "disagree," and 1 represents the "strongly disagree" option. During the experiment, the mean learner satisfaction was set to greater than 4, which infers that overall, the learners were satisfied while they used

Table 6: Learners' performance state, type of motivational triggers, and their example.

Learners' performance state	Type of motivational trigger	Motivational trigger example
Below-average learners	Fear	Continuous poor performance will result in your relegation/ceased state.
	Hope	Consistent 3 hours of study per day will make you one of the best learners of the class.
	Suggestion	Refers to the class group for newly uploaded videos by the class instructor.
Average learners	Praise	Well done and congratulations. You now are in the top 5 learners of your class.
	Appreciation	Your improving academic performance was praised by instructors in the meeting. Keep it up.
	Social acceptance	Imran Ahmed is now among the top 5 learners (trigger sent to all class learners).
Good consistent learners	Reward	The chairman will award you the best performance certificate for showing consistent performance.
	Praise	You showed excellent performance in the quiz. Well done.
	Appreciation	All the instructors and chairman appreciate your consistent performance throughout the semester.

Table 7: End-user computing satisfaction (EUCS) questionnaire result.

	Dimensions	Questions	Mean score
1	Usefulness	I contemplate the use of the Learnit app enhanced my programming skills	4.68
2	Usefulness	I think the Learnit app helped me a lot in understanding computer programming	4.59
3	Ease of use	The use of the Learnit app was very easy	4.52
4	Ease of use	I think learners can use the Learnit app without the instructor's help	4.58
5	Engaging	I was interested in using the Learnit app for enhancing my programming skills	4.78
6	Engaging	Learnit app was able to involve me in learning computer programming daily	4.71
7	Timeliness	The learning content was provided on time by the Learnit app	4.34
8	Timeliness	Learnit app knew when and what to send to learners while using it	4.53
9	Adaptive	The learning content provided by the Learnit app was tailored and adaptive	4.78
10	Adaptive	Learnit app enabled me to learn computer programming at my own pace	4.23
11	Attitude towards using the application	I will use similar types of applications in the future	4.55
12	Attitude towards using the application	I will continue to use the Learnit app to enhance my programming skills	4.34

the Learnit application. Table 7 shows the 6 dimensions, the related questions, and learners' satisfaction mean score.

The response to dimensions 1 and 2 (usefulness) indicates that the Learnit app was successful in increasing the job ability, i.e., programming skills of learners ($m = 4.68$ and $m = 4.59$). The response to dimensions 3 and 4 shows that the Learnit app interaction with learners was very simple and user-friendly ($m = 4.52$ and $m = 4.58$). The response to dimensions 5 and 6 indicates that the Learnit app was able to draw the attention of learners in giving time to learn computer programming using mobile devices ($m = 4.78$ and $m = 4.71$). The response to dimensions 7 and 8 implies that the Learnit app considered the experimental group preferred learning time and sent the programming content accordingly ($m = 4.34$ and $m = 4.53$). The learner replies to dimensions 9 and 10 revealed that tailored programming content was delivered to learners according to their learning abilities and performance by the Learnit app ($m = 4.78$ and $m = 4.23$). The answer to dimensions 11 and 12 reflects that learners agreed to use a Learnit or a similar type of application in the future to increase their programming skills ($m = 4.55$ and $m = 4.34$).

7. Conclusion and Future Work

In this research study, we have developed the M-learning model powered by the ANN. M-learning model is a part of a larger system called ILS. The purpose of ILS is to process and analyze M-learners' features to determine which features have the most influence on the learners' performance. The experimental results revealed that the ANN outperformed traditional ML algorithms such as LDA, softmax regression, SVM, DT, and RF and achieved a prediction accuracy of 90.77%, 87.69%, 83.85%, and 80.00% for four M-learning models. Moreover, by using the RF algorithm to determine the most important features having a significant impact on the M-learners' final performance, we enlighten that MP3, MP1, MP2, NTAQ, and AST have the highest weights and contribute most in increasing M-learners' performance.

In the future, we would like to test the M-learning model on a larger scale and integrate it with the Learning Management System (LMS). By doing so, the M-learning model will help us in making LMS an adaptive and more user-friendly system. One of the disadvantages of traditional LMS systems is that they are based on a one-size-fits-all approach

where all learners are treated equally not considering individual weaknesses, strengths, preferences, and personalized learning behavior. By integrating the M-learning model with LMS, we might be successful in making the learning process self-paced and adaptive. Moreover, we would also like to improve approaches presented in this research study by combining other deep neural network algorithms such as recurrent neural network (RNN), deep autoencoders, self-organizing maps (SOMs), and long short-term memory (LSTM) with the proposed M-learning model [60].

Data Availability

The dataset used to support the findings of this study can be found at https://drive.google.com/file/d/1dGvbdfK8d2nexEatqwPhcusk8fwrT7ZI/view?usp=sharing.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by a grant from the Research Center of the Female Scientific and Medical Colleges, Deanship of Scientific Research, King Saud University.

References

[1] H. Hamidi and A. Chavoshi, "Analysis of the essential factors for the adoption of mobile learning in higher education: a case study of students of the university of technology," *Telematics and Informatics*, vol. 35, no. 4, pp. 1053–1070, 2018.

[2] R. Christensen and G. Knezek, "Readiness for integrating mobile learning in the classroom: challenges, preferences and possibilities," *Computers in Human Behavior*, vol. 76, pp. 112–121, 2017.

[3] H. Crompton and D. Burke, "The use of mobile learning in higher education: a systematic review," *Computers & Education*, vol. 123, pp. 53–64, 2018.

[4] T. Jagušt and I. Botički, "Mobile learning system for enabling collaborative and adaptive pedagogies with modular digital learning contents," *Journal of Computers in Education*, vol. 6, no. 3, pp. 335–362, 2019.

[5] D. Mohamed, L. Nada, and O. Ilham, "Determining the learner's profile and context profile in order to propose adaptive mobile interfaces based on machine learning," in *Proceedings of the 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*, Kenitra, Morocco, December 2020.

[6] L. S. Riza, A. D. Pertiwi, E. F. Rahman, M. Munir, and C. U. Abdullah, "Question generator system of sentence completion in TOEFL using NLP and k-nearest neighbor," *Indonesian Journal of Science and Technology*, vol. 4, no. 2, pp. 294–311, 2019.

[7] K. Liang, Y. Zhang, Y. He, Y. Zhou, W. Tan, and X. Li, "Online behavior analysis-based student profile for intelligent e-learning," *Journal of Electrical and Computer Engineering*, vol. 2017, Article ID 9720396, 2017.

[8] P. Khumrina, R. Anna, J. Terry, and K. Verspoora, "Diagnostic machine learning models for acute abdominal pain: towards an e-learning tool for medical students," in *MEDINFO 2017: Precision Healthcare Through Informatics: Proceedings of the 16th World Congress on Medical and Health Informatics*IOS Press, Amsterdam, Netherlands, 2018.

[9] A. Verma Babbar and S. Kumar Henge, "Blended environment of naive Bayes and support vector machine (SVM) for designing simulation based e-learning respiratory system," in *Proceedings of the International Conference on Intelligent and Fuzzy Systems*, Istanbul, Turkey, July 2020.

[10] L.-S. Chen and J.-Y. Hsu, "Discover users' needs in e-learning by Kano analysis and decision trees," in *Proceedings of the 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, Tokyo, Japan, April 2019.

[11] H. Anantharaman, A. Mubarak, and B. T. Shobana, "Modelling an adaptive e-learning system using LSTM and random forest classification," in *Proceedings of the 2018 IEEE Conference on e-Learning, e-Management and e-Services (IC3e)*, Langkawi, Malaysia, November 2018.

[12] J. Melesko and E. Kurilovas, "Semantic technologies in e-learning: learning analytics and artificial neural networks in personalised learning systems," in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, Novi Sad, Serbia, June 2018.

[13] R. Pelánek, "Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques," *User Modeling and User-Adapted Interaction*, vol. 27, no. 3-5, pp. 313–350, 2017.

[14] M. Ciolacu, A. F. Tehrani, R. Beer, and H. Popp, "Education 4.0—fostering student's performance with machine learning methods," in *Proceedings of the 2017 IEEE 23rd International Symposium for Design and Technology in Electronic Packaging (SIITME)*, Constanta, October 2017.

[15] M. Van Gerven and B. Sander, "Artificial neural networks as models of neural information processing," *Frontiers in Computational Neuroscience*, vol. 11, p. 114, 2017.

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *Proceedings of the 2017 International Conference on Engineering and Technology (ICET)*, Antalya, Turkey, August 2017.

[17] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," *Physica D: Nonlinear Phenomena*, vol. 404, Article ID 132306, 2020.

[18] A. A. Hameed, B. Karlik, M. S. Salman, and G. Eleyan, "Robust adaptive learning approach to self-organizing maps," *Knowledge-Based Systems*, vol. 171, pp. 25–36, 2019.

[19] J. Wang, K. Wang, Y. Wang, Z. Huang, and R. Xue, "Deep Boltzmann machine based condition prediction for smart manufacturing," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 3, pp. 851–861, 2019.

[20] S. Zhang, Y. Yao, J. Hu, Y. Zhao, S. Li, and J. Hu, "Deep autoencoder neural networks for short-term traffic congestion prediction of transportation networks," *Sensors*, vol. 19, no. 10, p. 2229, 2019.

[21] S. B. Kotsiantis, "Use of machine learning techniques for educational proposes: a decision support system for forecasting students' grades," *Artificial Intelligence Review*, vol. 37, no. 4, pp. 331–344, 2012.

[22] M. Guo, E. Chou, D.-A. Huang, S. Song, S. Yeung, and L. Fei-Fei, "Neural graph matching networks for fewshot 3d action recognition," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, September 2018.

[23] H. A. Isiyaka, A. Mustapha, H. Juahir, and P. Phil-Eze, "Water quality modelling using artificial neural network and

multivariate statistical techniques," *Modeling Earth Systems and Environment*, vol. 5, no. 2, pp. 583–593, 2019.

[24] A. F. Schmidt and C. Finan, "Linear regression and the normality assumption," *Journal of Clinical Epidemiology*, vol. 98, pp. 146–151, 2018.

[25] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789–33795, 2018.

[26] H. S. Alenezi and M. H. Faisal, "Utilizing crowdsourcing and machine learning in education: literature review," *Education and Information Technologies*, vol. 25, pp. 2971–2986, 2020.

[27] J. P. Campbell, *Utilizing Student Data within the Course Management System to Determine Undergraduate Student Academic Success: An Exploratory Study*, Purdue University, West Lafayette, Indiana, 2007.

[28] F. Chiclana, R. Kumar, M. Mittal, M. Khari, J. M. Chatterjee, and S. W. Baik, "Arm–amo: an efficient association rule mining algorithm based on animal migration optimization," *Knowledge-Based Systems*, vol. 154, pp. 68–80, 2018.

[29] A. R. Anaya and J. G. Boticario, "Application of machine learning techniques to analyse student interactions and improve the collaboration process," *Expert Systems with Applications*, vol. 38, no. 2, pp. 1171–1181, 2011.

[30] W. T. Lin, S. J. Wang, Y. C. Wu, and T. C. Ye, "An empirical analysis on auto corporation training program planning by data mining techniques," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5841–5850, 2011.

[31] B. Guo, R. Zhang, G. Xu, C. Shi, and Li Yang, "Predicting students performance in educational data mining," in *Proceedings of the 2015 International Symposium on Educational Technology (ISET)*, Wuhan, China, July 2015.

[32] B. Akram, W. Min, E. Wiebe, B. Mott, K. E. Boyer, and J. Lester, "Improving stealth assessment in game-based learning with LSTM-based analytics," in *Proceedings of the International Conference on Educational Data Mining*, Raleigh, NC, USA, July 2018.

[33] W. W. T Fok and Y. S. He, H. H. Au Yeunh et al., Prediction model for students' future development by deep learning and tensorflow artificial intelligence engine," in *Proceedings of the 2018 4th International Conference on Information Management (ICIM)*, Oxford, UK, May 2018.

[34] K. Abhinav, V. Subramanian, A. Dubey, P. Bhat, and A. D. Venkat, "LeCoRe: a framework for modeling learner's preference," in *Proceedings of the International Conference on Educational Data Mining*, Buffalo, NY, USA, March 2018.

[35] M. V. Amazona and A. A. Hernandez, "Modelling student performance using data mining techniques: inputs for academic program development," in *Proceedings of the 2019 5th International Conference on Computing and Data Engineering*, New York, NY, USA, December 2019.

[36] J. G. M. Alvarado, H. A. Ghavidel, Z. Amal, J. Jovanovic, and J. McDonald, "A comparison of features for the automatic labeling of student answers to open-ended questions," in *Proceedings of the 2018 International Educational Data Mining Society*, Raleigh, NC, USA, July 2018.

[37] B.-H. Kim, E. Vizitei, and V. Ganapathi, "Gritnet: student performance prediction with deep learning," 2018, http://arxiv.org/abs/1804.07405.

[38] M. A. Samuel-Soma, N. B. Ahmad, and S. M. Shamsuddin, "A data mining approach to predict academic performance of students using ensemble techniques," in *Proceedings of the International Conference on Intelligent Systems Design and Applications*, Vellore, India, December 2018.

[39] M. M. Alam, K. Mohiuddin, A. K. Das, M. K. Islam, M. S. Kaonain, and M. H. Ali, "A reduced feature based neural network approach to classify the category of students," in *Proceedings of the 2nd International Conference on Innovation in Artificial Intelligence*, Shanghai, China, March 2018.

[40] M. Khajah, R. V. Lindsey, and M. C. Mozer, "How deep is knowledge tracing?," 2016, http://arxiv.org/abs/1604.02416.

[41] X. Ma, Y. Yang, and Z. Zhou, "Using machine learning algorithm to predict student pass rates in online education," in *Proceedings of the 3rd International Conference on Multimedia Systems and Signal Processing*, Shenzhen, China, April 2018.

[42] A. Lalwani and S. Agrawal, "Few hundred parameters outperform few hundred thousand," in *Proceedings of the 10th International Conference on Educational Data Mining, EDM*, Wuhan, Hubei, China, June 2017.

[43] M. El Fouki and N. Aknin, "Multidimensional approach based on deep learning to improve the prediction performance of DNN models," *International Journal of Emerging Technologies in Learning*, vol. 14, no. 2, 2019.

[44] S. M. Raza Abidi, M. Hussain, Y. Xu, and W. Zhang, "Prediction of confusion attempting algebra homework in an intelligent tutoring system through machine learning techniques for educational sustainable development," *Sustainability*, vol. 11, no. 1, p. 105, 2019.

[45] K. Hadullo, R. Oboko, and E. Omwenga, "Factors affecting asynchronous e-learning quality in developing countries university settings," *International Journal of Education and Development Using ICT*, vol. 14, no. 1, 2018.

[46] W. Wang, Yu Han, and C. Miao, "Deep model for dropout prediction in moocs," in *Proceedings of the 2nd International Conference on Crowd Science and Engineering*, Beijing, China, July 2017.

[47] I. Ndukwe, B. Daniel, and R. Butson, "Data science approach for simulating educational data: towards the development of teaching outcome model (tom)," *Big Data and Cognitive Computing*, vol. 2, no. 3, p. 24, 2018.

[48] J. Whitehill, K. Mohan, D. Seaton, Y. Rosen, and D. Tingley, "Delving deeper into mooc student dropout prediction," 2017, http://arxiv.org/abs/1702.06404.

[49] Y. Chai, C.-U. Lei, X. Hu, and Y.-K. Kwok, "WPSS: dropout prediction for moocs using course progress normalization and subset selection," in *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, London, UK, June 2018.

[50] C.-K. Yeung and D.-Y. Yeung, "Addressing two problems in deep knowledge tracing via prediction-consistent regularization," in *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, London, UK, June 2018.

[51] Y. Sun, Q. Xu, M. Jia, and B. Jing, "Research on undergraduate academic prediction model based on deep learning," in *Proceedings of the 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, Singapore, June 2018.

[52] E. Tanuar, Y. Heryadi, B. S. Abbas, and F. L. Gaol, "Using machine learning techniques to earlier predict student's performance," in *Proceedings of the 2018 Indonesian Association for Pattern Recognition International Conference (INAPR)*, Jakarta, Indonesia, September 2018.

[53] J. Yang, K. Wang, X. Peng, and Y. Qiao, "Deep recurrent multi-instance learning with spatio-temporal features for engagement intensity prediction," in *Proceedings of the 20th ACM International Conference on Multimodal Interaction*, Boulder, CO, USA, October 2018.

[54] D. Islam, *Analysis and Assessment of Skill Using Data Mining*, 2018.

[55] X. Wang, P. Wu, G. Liu, Q. Huang, X. Hu, and H. Xu, "Learning performance prediction via convolutional GRU and explainable neural networks in e-learning environments," *Computing*, vol. 101, no. 6, pp. 587–604, 2019.

[56] A. A. Saa, M. Al-Emran, and K. Shaalan, "Mining student information system records to predict students' academic performance," in *Proceedings of the International Conference on Advanced Machine Learning Technologies and Applications*, Cairo, Egypt, March 2019.

[57] S. Adam, A. F. Botelho, T. Patikorn, and N. T. Heffernan, "Using big data to sharpen design-based inference in a/b tests," in *Proceedings of the Eleventh International Conference on Educational Data Mining*, Buffalo, NY, USA, July 2018.

[58] A. Sharma, A. Biswas, A. Gandhi, S. Patil, and O. Deshmukh, "Livelinet: a multimodal deep recurrent neural network to predict liveliness in educational videos," in *Proceedings of the International Educational Data Mining Society*, Raleigh, NC, USA, June 2016.

[59] J. F. Brian, "A behavior model for persuasive design," in *Proceedings of the 4th International Conference on Persuasive Technology*, Claremont, CA, USA, April 2009.

[60] T.-C. Hsia, A.-J. Shie, and L.-C. Chen, "Course planning of extension education to meet market demand by using data mining techniques - an example of Chinkuo technology university in Taiwan," *Expert Systems with Applications*, vol. 34, no. 1, pp. 596–602, 2008.