International Journal of Electrical and Computer Engineering (IJECE)
Vol. 15, No. 3, June 2025, pp. 2998~3009
ISSN: 2088-8708, DOI: 10.11591/ijece.v15i3.pp2998-3009

Deep learning for predicting drug-related problems in diabetes patients

Fatima M. Smadi, Qasem A. Al-Radaideh
Department of Information Systems, Yarmouk University, Irbid, Jordan

Article Info

Article history:

Received Jun 28, 2024
Revised Jan 3, 2025
Accepted Jan 16, 2025

Keywords:

Deep learning
Deep neural networks
Hyperparameter optimization
Long-short term memory networks
Machine learning
Tabular data

ABSTRACT

Machine learning and deep learning have made advances in the healthcare domain. In this study, we aim to apply deep learning models to predict the drug-related problems (DRPs) status for diabetes patients. Also, to determine the appropriate model to use for classification using deep learning algorithms or machine learning methods to investigate which one performed better results for tabular data by comparing the achieved deep learning results with the machine learning methods to figure out which one gives better results. To apply the deep learning models, the same criteria that were applied in the previous study have been implemented in this investigation, and the same dataset was used. The results show that the machine learning algorithms especially the random forests for predicting the DRPs status outperform the deep learning models. For classification tasks in healthcare for tabular data, the findings of this study show that machine learning methods are the appropriate model instead of using deep learning to apply classification.

Corresponding Author:

Fatima M. Smadi
Department of Information Systems, Yarmouk University
Irbid 21163, Jordan
Email: fatima.smadi13@gmail.com

1. INTRODUCTION

Recently, with large amounts of data becoming available and the improvement in memory handling capabilities of the computers making the process of handling huge amounts of data easier, and the progress of machine learning and deep learning algorithms [1], [2]. However, tabular data is the most prevalent data type employed in many real-world applications. It is used in various fields, including medicine, fraud detection, computer vision, and disease diagnostic [3].

In recent years, deep learning has gained a lot of traction in data mining during the last six years, the term “deep learning” was coined in 2006 [4]. It is used to investigate healthcare topics in order to improve patient care [5]. However, deep learning is a computational method that allows machines to analyze data like the human brain. Thus, it is a special type of machine learning that involves a deeper level. Also, deep learning differs from a neural network in that it has many layers and neurons in big numbers. On the other hand, the use of deep learning has evolved in many areas in the following applications such as speech recognition, medical research in cancer research to automatically detect cancer cells, medical imaging, drugs discovery, and natural language processing [1]–[5].

Deep neural networks (DNN) trainee the network through feed-forward neural networks and use backpropagation. The structural building block of deep learning is the perceptron (neuron). Deep learning contains the input layer, hidden layers, and output layer [4]. In addition, the DNN is characterized that it has more than one hidden layer, the direction of the information in the network is forward (there are no circles)

from input to the output through hidden layers, also known as multi-layer perceptron (MLP) [6]. Furthermore, the long-short term memory networks (LSTM) implement a more complex recurrent unit with gates to control what information is passed through. Also, it is considered a state-of-the-art model [7]. However, the LSTM contains memory blocks that control the information flow, the key building block behind LSTM is the gates, the memory cells in LSTM layers interact with each other via gates, and the gates are responsible for enabling the LSTM to be able to add or remove information to its cell state. In addition, The LSTM follows the main four steps: forgetting irrelevant history, then performing computations to store relevant information, updating the internal state, and finally generating the output to be an input to the next LSTM layer. Training LSTM network applied the backpropagation method [8], [9].

As for using deep learning techniques in the Healthcare area. Kim *et al.* [10] used the deep belief network (DBN) and different machine learning algorithms to predict cardiovascular risk prediction (low risk and high risk)). The authors evaluated the models using three performance metrics (accuracy, specificity, and sensitivity). The naïve Bayes (NB) model obtained (79%, 63%, 84%), logistic regression (LR) model obtained (80%, 69%, 82%), support vector machine (SVM) model obtained (71%, 100%, 71%), random forests (RF) model obtained (77%, 61%, 82%), deep belief network (DBN) model obtained (75%, 82%, 74%), and a statistical DBN model obtained (83, 73%, 87%), the results showed that the statistical DBN model outperformed the other classifiers.

Korotcov *et al.* [11] used SVM, RF, and logistic linear regression to compare the performance of different machine learning algorithms to the deep neural networks (DNN) to predict drug discovery. The results of the performance metrics showed accuracy, precision, and recall, respectively. The DNN model obtained (87%, 68%, 70%), SVM model obtained (83%, 59%, 74%), logistic linear regression model obtained (78%, 58%, 74%), RF model obtained (79%, 59%, 70%), the results showed that the DNN model outperformed the machine learning algorithms.

Other research used deep learning and machine learning to predict brain tumors in the brain [12], the authors used deep learning and machine learning algorithms to predict brain tumors in the brain. They used the deep neural network and k-nearest neighbor algorithms (*k*-NN) in their study. The results of the performance metrics showed accuracy, precision, and recall, respectively. The DNN model obtained (96%, 97%, 97%), *k*-nearest neighbor model with *k*=1 obtained (95%, 95%, 95%), The *k*-NN model with *k*=3 obtained (86%, 89%, 86%).

Ayon and Islam [13] used DNN to diagnose diabetes (absence and present). The results of the performance metrics for ten-folds showed accuracy, specificity, and sensitivity, respectively. The DNN model obtained (97%, 96%, 97%).

Massaro *et al.* [14] used LSTM, LSTM using artificial records (LSTM-AR), and MLP to predict diabetes by classifying diabetes status and no-diabetes status. The reported result of the performance metrics showed accuracy. The LSTM model obtained (75%), LSTM-AR model obtained (84%), and MLP model obtained (77%), by applying the LSTM-AR there was an improvement in results compared with MLP and LSTM.

Tigga and Garg [15] used LR, *k*-NN, SVM, NB, decision tree (DT), and RF to predict type 2 diabetes patients. The results of the performance metrics showed accuracy. The LR model obtained (85%), *k*-NN model obtained (77%), SVM model obtained (86%), NB model obtained (80%), DT model obtained (84%), and the RF obtained (94%). The results showed that the RF model outperformed the other algorithms.

Shwartz-Ziv and Armor [16] demonstrated that when working with tabular data in classification and regression, deep learning models are not all you need. In their study, they compared tree ensemble models such as XGBoost with deep learning models to see which of them gives better results for tabular data. The study found that the XGBoost outperforms the deep models, it requires much less tuning. So, they recommend using ensemble models when using tabular data.

Hairani and Priyanto [17] used SVM and RF combined with the synthetic minority over sampling technique (SMOTE), edited nearest neighbors (ENN), and hybrid SMOTE-ENN methods. The reported result of the performance metrics showed accuracy, sensitivity, and specificity, respectively. The SVM with SMOTE model obtained (74%, 70%, 77%), SVM with ENN model obtained (85%, 85%, 86%), SVM with SMOTE-ENN model obtained (90%, 91%, 88%), RF with SMOTE model obtained (82%, 86%, 78%), RF with ENN model obtained (87%, 86%, 87%), RF with SMOTE-ENN model obtained (95%, 98%, 92%). The findings showed that the RF with SMOTE-ENN model outperformed SMOTE and ENN individually in terms of average performance.

Chu *et al.* [18] used DNN to develop a cardiovascular disease (CVD) risk prediction. The results of the performance metrics showed accuracy, specificity, and sensitivity respectively. The DNN model obtained (87.50%, 87.23%, 88.06%).

The study contributes by applying deep learning techniques to new data that has not been applied before using deep learning models, the data was collected from six Majour hospitals in Jordan especially focusing on diabetic patients to predict the status of drug-related problems. By comparing the results of deep

learning techniques with machine learning methods, the study achieves high accuracy in DRP status prediction and outperforms previous studies. This study highlights a foundation for future research in predicting drug-related problems.

Many researchers have used deep learning models and machine learning methods in healthcare systems such as breast cancer, brain cancer, and drug discovery [19], [20]. When handling classification tasks in data mining with tabular data in healthcare systems, it becomes unclear to determine which model to use deep learning models or machine learning techniques. Deep learning models have shown the ability to handle large datasets, but they require computational resources and large amounts of data. However, machine learning methods, are known for their effective practical, and easy implementation. This study aims to investigate the best structure of the used deep learning techniques (DNN and LSTM) to predict the status of drug-related problems, and whether to apply deep learning models or machine learning methods when dealing with tabular data for classification. To perform this, the same dataset used in the [21] has been used. In addition, we compared the results obtained from deep learning models with the machine learning classifiers.

2. RESEARCH METHOD

The main purpose of this study is to investigate the best structure of the DNN and LSTM to predict the status of drug-related problems. Also, to find out the effect of applying deep learning compared with the machine learning methods. Additionally, whether to apply deep learning models or machine learning methods when dealing with tabular data for classification. Also, to find out which will give high performance for tabular data compared with the study in [21]. To perform this study, the methodology's implementation is subdivided into several steps. Figure 1 summarizes the overall methodology steps in detail. The deep learning models that were experimented in this study are briefly described in the following subsection. Deep learning or neural networks books will provide a detailed description.

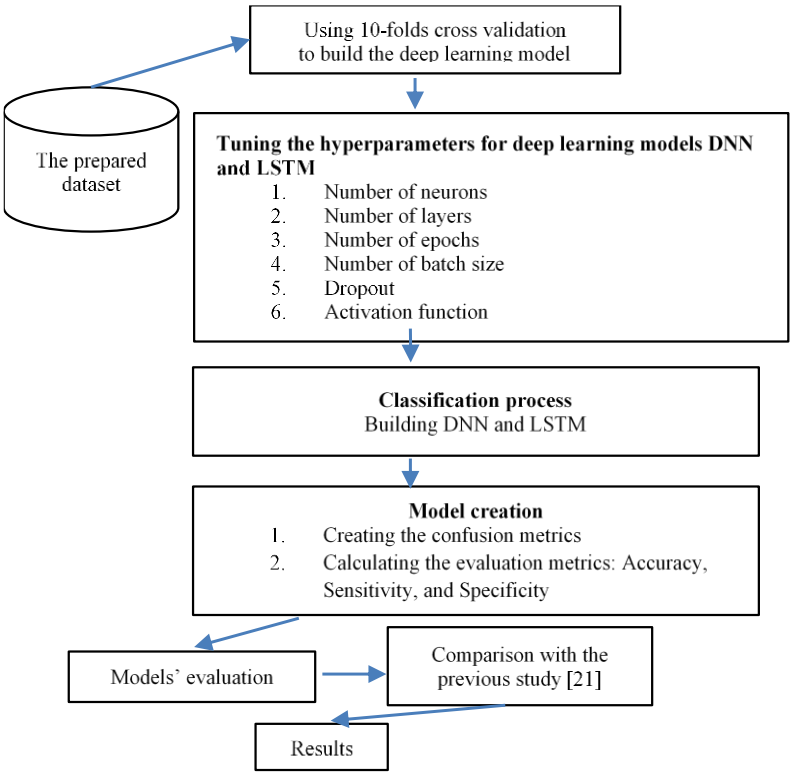

Figure 1. The research methodology

DNN are a kind of neural network inspired by the design of the human nervous system and the brain's architecture [22]. Additionally, DNN characterized that it has more than one hidden layer, the direction of the information in the network is forward (there are no circles) from input to the output through hidden layers, it also known as MLP [6]–[13]. However, the layers of the DNNs are the input layer, hidden

layers (multiple hidden layers), and output layer. The hidden layer's location is located between the input layer and the output layer. Although the input layer is responsible for passing the inputs from the dataset to the next layer, the hidden layers are responsible for applying the non-linear transformation, and the output layer is responsible for producing the output. Furthermore, the essential elements of DNN are neurons, weights, non-linear activation function, and bias. In DNN there is a fully connected layer that is defined through the dense class while every neuron in the layer is considered an input to the neurons in the next layers, the model uses stochastic gradient descent in training the model. As well, The DNN training going through phases that starts with forward propagation followed by the backpropagation and the adjustment process [23].

The forward propagation method in the neural network passes the data through the network starting with the input layer by passing multiple inputs at once. Each input value (x_i) needs to be multiplied by the corresponding weights (w_i) and then added with all the other results for each neuron, then calculating the sum of the weighted inputs to each neuron, and the bias (b) is added in (1), then applying a non-linear activation function. The result of the activation function is considered an output for the layer and an input for the neurons in the next hidden layer. This process will be applied in each layer until we reach the output layer to get the prediction (Y) [8].

$$Z = \sum_i x_i w_i + b \quad (1)$$

The backpropagation method is used to minimize the error in the network, when a neural network is trained using forward propagation by randomly initializing the weights of all neurons to apply the prediction, if the prediction for the model is incorrect (error in the prediction), the backpropagation must be used to minimize the error, which is calculated by the difference between the output from the forward propagation (prediction output) and the expected outputs (desired output). The goal is to let the error close to zero. When the prediction error is generated through the forward propagation, it will be in a high number, by applying the backward propagation from the output layer and updating the weights to achieve the first layer, then applying the forward propagation for the second time and generating the prediction error for the second time, the model keeps doing these steps for all samples in the training data until the specified epochs are reached and the error is minimized [8].

The LSTM networks is considered a special type of recurrent neural networks (RNN). It has an input layer, LSTM hidden layers, and an output layer. According to Takeuchi *et al.* [24], the RNN has a difficulty in training because of the vanishing gradient problem, thus the LSTM overcomes this problem by composing an input gate, an output gate, and a forget gate. However, the LSTM architecture contains a set of memory blocks, each block containing recurrently connected memory cells that are connected via gates that allow the LSTM to add or remove information from the cell. The memory cell's activation function allows storing a state for either a short moment or an extended amount of time. Also, each memory cell applies several steps to move the state to the next LSTM hidden layer and implement the same process to reach the output layer. The LSTM is trained through forward propagation and backpropagation methods [25]. There are many applications of LSTM such as Automatic image caption generation and automatic translation of the text. Also, the LSTM has become increasingly used in health care in recent years.

2.1. Experiments design

In this study, we used the same method that was used in the previous study [21], the cross-validation method with 10 folds has been applied to build the deep learning predictive models. The following steps were followed to perform this study:

a. Using the cross-validation with 10 folds in deep learning models.
b. Tuning the hyper-parameters to get the best structure for the DNN and LSTM through building and training the models by the training data.
c. Using the testing data to apply the prediction.
d. Using the confusion matrix metrics to evaluate the deep learning models.
e. Comparing the evaluated model results with the previous study [21].

2.2. Performance measures to evaluate the models

To evaluate the deep learning models, the same criteria that were applied for the previous study [21] have been implemented in this investigation. Moreover, the overall performance metrics were generated via the confusion metrics to discover the model's performance including accuracy, sensitivity, and specificity. The confusion metrics were calculated using (2)-(4):

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (2)$$

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

$$\text{Specificity} = \frac{TN}{TN+FP} \quad (4)$$

3. RESULTS AND DISCUSSION

We attempted to tune the best network structure to achieve the best performance. There were many different values from hyper-parameters to train the DNN and LSTM models to enhance it and get the best performance of the DNN and LSTM models based on many experiments to choose the best structure of the DNN and LSTM. Applying the process of training the model with the write hyper-parameters is based on experiments and results by trial and error. It depends on the experience with many experiments by tuning the classifier model, it takes time and effort to tune the hyper-parameters, when tuning hyper-parameters, we have to select some parameters with the correct configuration for each parameter one at a time and then continue to configure the next parameter and so on to get the best performance you are looking for [23].

3.1. Deep neural networks experiments and results

The DNN algorithm was used to create the classifier model, we applied the cross-validation method. Based on several experiments, we started by defining the baseline architecture to start building the model of the deep neural networks model with the hyper-parameters specified. The baseline architecture that was used to start building the DNN model: a fully connected neural network with three hidden layers with 31 neurons in each hidden layer, respectively, the activation function for each layer was rectified linear unit (ReLU), the dropout was 0.5, for the optimizer was Adam, the epochs was 50, and the batch size was 32. The results for the baseline were accuracy: 88.54, sensitivity: 87.50, and specificity: 89.58. The following experiments were performed to get the best structure for the DNN model.

3.1.1. Tune the number of hidden neurons in each hidden layer

The following experiments were used to tune the number of hidden neurons in each hidden layer. Table 1 displays the experiment results on different numbers of hidden neurons. When using 78, 109, 124, 155, and 171 hidden neurons in the three hidden layers, we tested these neurons with epochs (50), and batch size (32), the best result occurred when we used the 124 hidden neurons, there was an improvement in the results until we reached the 124 hidden neurons used in the DNN model, after (124) we noted that there was no improvement in the results, we concluded that (124) hidden neurons used in the DNN model was the best.

Table 1. The hidden neurons experiment for DNN

Number of hidden neurons	Accuracy %	Sensitivity %	Specificity %
78	90.10	91.66	88.54
109	89.06	88.54	89.58
124	**91.14**	**88.54**	**93.75**
155	88.54	92.70	84.37
171	87.50	94.79	80.20

3.1.2. Tune the number of hidden layers

The following experiments were used to tune the number of hidden layers. Table 2 displays the experiment results on different numbers of hidden layers. When using 3, 4, 5, 6, and 7 hidden layers, we tested these hidden layers with (124) hidden neurons in each hidden layer respectively, epochs (50), and batch size (32), the best result occurred when we used three hidden layers, we concluded that the number of hidden layers used in the DNN model was the best, as there was no improvement in the results when increasing the number of hidden layers.

Table 2. Number of hidden layers experiment for DNN

Number of hidden layers	Accuracy %	Sensitivity %	Specificity %
3	**91.14**	**88.54**	**93.75**
4	86.97	89.58	84.37
5	85.41	90.62	80.20
6	79.68	96.87	62.50
7	72.91	95.83	50.00

3.1.3. Tune the Epochs for DNN

The following experiments were used to tune the epochs. Table 3 displays the experiment results on different Epochs. When using 50, 200, 300, 400, and 500 epochs, we tested these epochs with (124) hidden neurons in each hidden layer respectively, with three hidden layers, and batch size (32), the best result occurred when we used 400 epochs, there was an improvement in the results until we reached the 400 epochs in the DNN model, after 400 epochs, we noted that there was no improvement in the results, we concluded that (400) epochs used in the DNN model were the best.

Table 3. Number of Epochs experiment for DNN

Number of Epochs	Accuracy %	Sensitivity %	Specificity %
50	91.14	88.54	93.75
200	95.31	97.91	92.70
300	94.79	94.79	94.79
400	**95.57**	**95.31**	**95.83**
500	92.70	95.83	89.58

3.1.4. Tune the batch size for DNN

The following experiments were used to tune the batch size. Table 4 displays the experiment results on different batch sizes. When using 16, 32, 64, 128, and 256 batch size, we tested these batch sizes with (124) hidden neurons in each hidden layer respectively, with three hidden layers, and epochs (400), the best result occurred when we used 32 batch size, we concluded that the 32-batch size used in the DNN model was the best, as there was no improvement in the results when testing different epochs.

Table 4. Number of batch size experiment for DNN

Number of batch size	Accuracy %	Sensitivity %	Specificity %
16	94.79	94.79	94.79
32	**95.57**	**95.31**	**95.83**
64	95.31	98.95	91.66
128	96.87	98.95	94.79
256	92.70	96.87	88.54

3.1.5. Tune the dropout rate for DNN

The following experiments were used to tune the dropout. Table 5 displays the experiment results on different dropout rates. When using 20%, 30%, 40%, and 50% dropout rates, we tested these dropouts with (124) hidden neurons in each hidden layer respectively, with three hidden layers, and epochs (400), and batch size (32), the best result occurred when we used 50% (0.5) dropout rate, we concluded that the 0.5 dropout used in the DNN model was the best, as there was no improvement in the results when testing different dropouts.

Table 5. Number of dropout experiments for DNN

Number of dropouts	Accuracy %	Sensitivity %	Specificity %
0.2	95.31	98.95	91.66
0.3	96.35	98.95	93.75
0.4	93.75	97.91	89.58
0.5	**95.57**	**95.31**	**95.83**

3.1.6. Tune the activation function for DNN

The following experiments were used to tune the activation function. Table 6 displays the experiment results on different activation functions. When using ReLU, Tanh, and Sigmoid activation functions, we tested these activation functions with (124) hidden neurons in each hidden layer respectively, with three hidden layers, and epochs (400), dropout (0.5), and batch size (32), the best result occurred when we used ReLU activation function, we concluded that the ReLU activation function used in the DNN model was the best, as there was no improvement in the results when testing different activation functions. Accordingly, Figure 2 summarizes the structure of DNN.

Table 6. The activation functions experiment for DNN

The activation functions	Accuracy %	Sensitivity %	Specificity %
ReLU	**95.57**	**95.31**	**95.83**
Tanh	94.27	95.83	92.70
Sigmoid	93.75	94.79	92.70

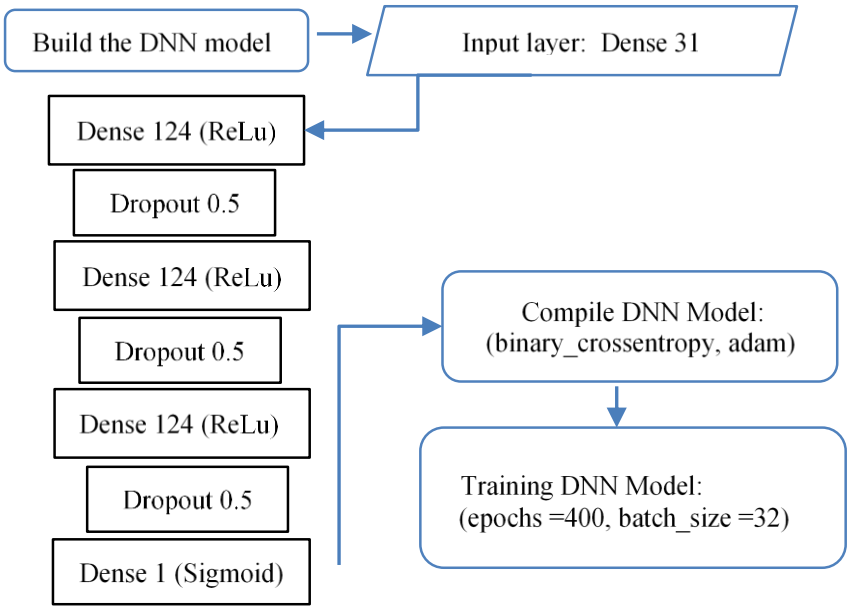

Figure 2. The structure of deep neural networks

3.2. LSTM networks experiments and results

Based on several experiments, we started by defining the baseline architecture to start building the model of the LSTM with the hyper-parameters specified. The results for the baseline architecture were Accuracy: 86.45, Sensitivity: 84.37, and Specificity: 88.54. The number of LSTM layers was (3), the hidden neurons were (47) in each LSTM layer, the epochs were (50), the batch size was (16), the dropout was (0.3), and the optimizer was Adam. The following experiments were performed to get the best structure for the LSTM model.

3.2.1. Tune the number of hidden neurons in each LSTM layers

The following experiments were used to tune the number of hidden neurons in each hidden layer. Table 7, displays the experiment results on different numbers of hidden neurons. When using 31, 62, 78, 109, and 124 hidden neurons in the three hidden layers, we tested these neurons with epochs (50), batch size (16), and dropout (0.3), the best result occurred when we used the 31 hidden neurons, there was an improvement in the results when tested 31 hidden neurons used in the LSTM model, after (31) we noted that there was no improvement in the results, we concluded that (31) hidden neurons used in the LSTM model was the best.

Table 7. The hidden neurons experiment for LSTM

Number of hidden neurons	Accuracy %	Sensitivity %	Specificity %
31	**88.02**	**84.37**	**91.66**
62	84.89	88.54	81.25
78	82.81	91.66	73.95
109	88.02	85.41	90.62
124	87.50	84.37	90.62

3.2.2. Tune the number of LSTM layers

The following experiments were used to tune the number of LSTM hidden layers. Table 8 displays the experiment results on different numbers of LSTM hidden layers. When using 3, 4, 5, 6, and 7 LSTM hidden layers, we tested these hidden layers with (31) hidden neurons in each LSTM hidden layer

respectively, epochs (50), and batch size (16), and dropout (0.3), the best result occurred when we used five LSTM hidden layers, there was an improvement in the results until we reached the five LSTM layers used in the LSTM model, after five LSTM layers we noted that there was no improvement in the results, we concluded that five LSTM layers used in the LSTM model were the best.

Table 8. Number of hidden layers experiment for LSTM

Number of LSTM layers	Accuracy %	Sensitivity %	Specificity %
3	88.02	84.37	91.66
4	86.45	91.66	81.25
5	**88.02**	**82.29**	**93.75**
6	86.97	83.33	90.62
7	86.45	86.45	86.45

3.2.3. Tune the epochs for LSTM

The following experiments were used to tune the epochs. Table 9 displays the experiment results on different Epochs. When using 50, 100, 200, 300, and 400 epochs, we tested these epochs with (31) hidden neurons in each LSTM hidden layers respectively, with five LSTM hidden layers, and batch size (16), the best result occurred when we used 50 epochs, we concluded that the 50 epochs used in the LSTM model were the best, as there was no improvement in the results when testing different epochs.

Table 9. Epochs experiment for LSTM

Number of Epochs	Accuracy %	Sensitivity %	Specificity %
50	**88.02**	**82.29**	**93.75**
100	88.02	85.41	90.62
200	90.62	90.62	90.62
300	90.10	97.91	82.29
400	92.70	97.91	87.50

3.2.4. Tune the batch size for LSTM

The following experiments were used to tune the batch size. Table 10 displays the experiment results on different batch sizes. When using 16, 32, 64, 128, and 256 batch size, we tested these batch sizes with (31) hidden neurons in each LSTM hidden layers respectively, with five LSTM hidden layers, dropout (0.3), and epochs (50), the best result occurred when we used 32 batch size, there was an improvement in the results when tested 32 batch size used in the LSTM model, after (32) we noted that there was no improvement in the results, we concluded that (32) batch size used in the LSTM model was the best.

Table 10. Batch size experiment for LSTM

Number of batch size	Accuracy %	Sensitivity %	Specificity %
16	88.02	82.29	93.75
32	**86.97**	**79.16**	**94.79**
64	85.93	80.20	91.66
128	84.89	85.41	84.37
256	84.89	83.33	86.45

3.2.5. Tune the dropout rate for LSTM

The following experiments were used to tune the dropout. Table 11 displays the experiment results on different dropout rates. When using 20%, 30%, 40%, and 50% dropout rates, we tested these dropouts with (31) hidden neurons in each LSTM hidden layers respectively, with five LSTM hidden layers, and epochs (50), and batch size (32), the best result occurred when we used 40% (0.4) dropout rate, there was an improvement in the results when tested 0.4 dropouts used in the LSTM model, after (0.4) we noted that there was no improvement in the results, we concluded that (0.4) dropout used in the LSTM model was the best. Accordingly, Figure 3 summarizes the structure of LSTM.

After trying many experiments to find out the best structure, the trials showed that the best structure for the DNN algorithm was when using 124 hidden neurons in three hidden layers, with a dropout rate of 0.5, epochs (400), batch size (32), and ReLU activation function. Also, the trials showed that the best structure for the LSTM algorithm was when using 31 hidden neurons in five LSTM hidden layers, with a dropout rate 0.4, epochs (50), and batch size (32). A similar specificity achieved by the DNN and LSTM models is observed.

Whereas the accuracy and the sensitivity for the DNN model were higher than the LSTM model. Table 12 demonstrates the results for the DNN and LSTM to get the best structure.

Table 11. Number dropout experiment for LSTM

Number of dropouts	Accuracy %	Sensitivity %	Specificity %
0.2	86.97	81.25	92.70
0.3	86.97	79.16	94.79
0.4	**87.50**	**79.16**	**95.83**
0.5	86.45	79.16	93.75

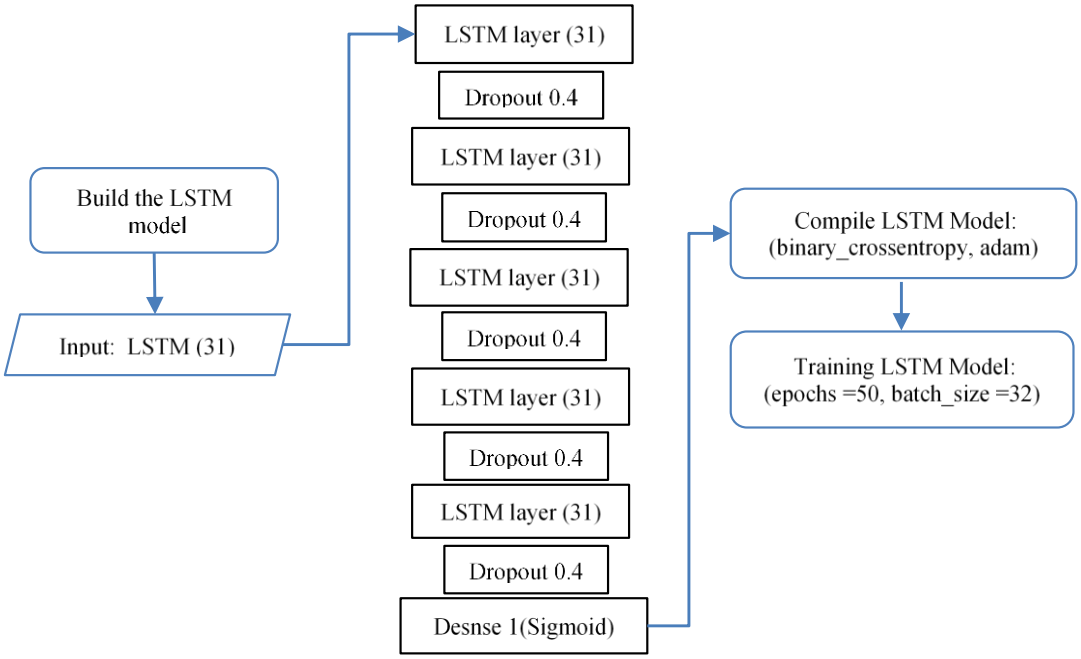

Figure 3. The structure of long-short term memory networks

Table 12. Summary of the results obtained by DNN and LSTM best structure

Classifiers	Hyperparameters	Options	Accuracy %	Sensitivity %	Specificity %
DNN	No. of hidden neurons	[78,109,**124**,155,171]	91.14	88.54	93.75
	No. of hidden layers	[**3**,4,5,6,7]	91.14	88.54	93.75
	Epochs	[50,200,300,**400**,500]	95.57	95.31	95.83
	Batch size	[16,**32**,64,128,256]	95.57	95.31	95.83
	Dropout	[0.2,0.3,0.4,**0.5**]	95.57	95.31	95.83
	Activation function	[**ReLU**, Tanh, Sigmoid]	95.57	95.31	95.83
	DNN best structure	[124,3,400,32,0.5]	95.57	95.31	95.83
LSTM	No. of hidden neurons	[**31**, 62,78,109,124]	88.02	84.37	91.66
	No. of LSTM layers	[3,4,**5**,6,7]	88.02	82.29	93.75
	Epochs	[**50**,100,200,300,400]	88.02	82.29	93.75
	Batch size	[16,**32**,64,128,256]	86.97	79.16	94.79
	Dropout	[0.2,0.3,**0.4**,0.5]	87.50	79.16	95.83
	LSTM best structure	[31,5,50,32,0.4]	87.50	79.16	95.83

The findings of this study indicate that the obtained results showed that the best structure for the DNN algorithm was when using 124 hidden neurons in three hidden layers, with a dropout rate of 0.5, epochs (400), batch size (32), and ReLU activation function. Also, the best structure for the LSTM algorithm was when using 31 hidden neurons in five LSTM hidden layers, with a dropout rate (0.4), epochs (50), and batch size (32). Additionally, the experiment results showed that the DNN obtained an accuracy of 95.57%, sensitivity of 95.31%, and specificity of 95.83%. On the other hand, the LSTM obtained an accuracy of 87.50%, sensitivity of 79.16%, and specificity of 95.83%. After comparing the results for the deep learning models and the machine learning algorithms particularly the random forests for predicting the drug-related

problems (DRPs) status applied in [21], the random forests algorithm outperformed the deep learning models in terms of accuracy and sensitivity when working with tabular data in classification tasks. In the healthcare field, accuracy is essential. Choosing the appropriate model for the data can have a major effect on patient outcomes. Our study recommends that pharmacists should use machine learning methods when working on identifying the DRPs status of diabetic patients for classification tasks in healthcare to increase the quality of healthcare services and identify the DRPs status for diabetic patients.

The primary objective of this study was to apply deep learning models to predict the DRPs status of diabetes patients. We investigate the best structure of the DNN and LSTM to predict the status of drug-related problems. Moreover, to find out the effect of applying deep learning compared with machine learning methods. Additionally, whether to apply deep learning models or machine learning methods when dealing with tabular data for classification and to find out which will give high performance for tabular data compared with the study in [21].

Choosing the right model for the tabular data depends on our understanding of the nature of the data, whether to use deep learning or machine learning. As demonstrated in the study [16], the authors compared tree ensemble models such as XGBoost with deep learning models to determine which performs better results for tabular data. Their results showed that the XGBoost outperforms the deep learning models and requires much less tuning. Therefore, instead of applying deep learning models when working with tabular data, they recommend using ensemble models. Additionally, our results show that machine learning, particularly the random forests method applied in [21], performed better than previous studies with high accuracy (97.39%), specificity (95.83%), and sensitivity (98.95%) as shown in Table 13. Table 13, summarizes the performance comparison of deep learning and machine learning with previous studies.

According to Al-Radaideh *et al.* [21], the random forests method achieved the following results accuracy of 97.39%, sensitivity of 98.95%, and specificity of 95.83%). In this study, the experiment results showed that the DNN obtained an accuracy of 95.57%, sensitivity of 95.31%, and specificity of 95.83%. Additionally, the LSTM obtained an accuracy of 87.50%, sensitivity of 79.16%, and specificity of 95.83%. When comparing the results of the DNN and LSTM with the random forests results, we noted that the random forests algorithm applied in [21] has achieved the same specificity metric results compared with the DNN and LSTM models as shown in Table 13. Additionally, we noted that the random forests outperformed the results of the DNN in terms of accuracy with an increase of (1.82%) and sensitivity with an increase of (3.64). Also, it outperformed the results of the LSTM in terms of accuracy with an increase of (9.89%) and sensitivity with an increase of (19.79%).

As a result, when comparing the results, it was found that using machine learning algorithms particularly the random forests used in [21] to predict the DRPs status obtained the best results compared to the deep learning models in terms of accuracy and sensitivity when working with tabular data in classification tasks. Compared to deep learning models, the machine learning algorithms applied in [21] appear to be more effective and offer a significant improvement over previous deep learning models in terms of accuracy and sensitivity as well as improved outcomes. In addition, tuning the deep learning hyper-parameters to achieve the best structure is a complex process that depends on trial and error and requires time and effort. It also requires a long run-time to train the model, unlike machine learning which requires less time and effort.

The data used in this study was collected from six major hospitals in Jordan. Because of this regional restriction, the dataset may be skewed toward the particular regions where these hospitals are located and might not represent large populations, such as those in the US. Future research should expand the data to include data from other regions and populations. This will improve the generalizability of the machine learning method and offer a deeper understanding of drug-related problems across different geographic groups.

Table 13. A performance comparison between the applied deep learning models and previous studies

Studies	Models	Accuracy %	Specificity %	Sensitivity %
[10]	Statistical DBN	83	73	87
[11]	DNNs	87	-	-
[12]	DNNs	96	-	-
[13]	DNNs	97	96	97
[14]	LSTM using AR (LSTM-AR)	84	-	-
[15]	RF	94	-	-
[17]	RF with SMOTE-ENN	95	92	98
[18]	DNNs	87	87	88
[21]	RF	97	95	98
Our study (DNN)	DNNs	95	95	95
Our study (LSTM)	LSTM networks	87	95	79

4. CONCLUSION

The results of this study indicate that the machine learning algorithms appear to be more effective and offer a significant improvement over deep learning models in terms of accuracy and sensitivity, as well as showing improved outcomes. The results indicated that the random forests algorithm performed better than the deep learning models when compared to machine learning techniques. In the research field, these findings assist pharmacists in accurately identifying the drug-related problems status of diabetes patients, allowing them to enhance patient care and improve the standards of healthcare generally. Finally, we recommend that instead of using deep learning models for tabular data, pharmacists employ the random forests classifier, as machine learning has been demonstrated to produce better results.

Moreover, pharmacists at hospitals in Jordan can implement the random forests classifier. This will enhance their ability to recognize any drug related problems quickly and accurately, which will lead to better patient outcomes, and reduced unwanted complications arising from DRPs, and improved patient outcomes. This classifier ensures greater accuracy in healthcare services.

FUNDING INFORMATION

The authors declare that no funding was received for this study.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Fatima M. Smadi		✓	✓	✓	✓	✓	✓	✓	✓		✓			
Qasem A. Al-Radaideh	✓	✓	✓	✓		✓	✓			✓	✓	✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the author, QAAR, upon reasonable request.

REFERENCES

[1] G. Rebala, A. Ravi, and S. Churiwala, *An introduction to machine learning*. Cham: Springer International Publishing, 2019.
[2] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: a survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 7499–7519, Jun. 2024, doi: 10.1109/TNNLS.2022.3229161.
[3] Y. Luo, H. Zhou, W.-W. Tu, Y. Chen, W. Dai, and Q. Yang, "Network on network for tabular data classification in real-world applications," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, Jul. 2020, pp. 2317–2326, doi: 10.1145/3397271.3401437.
[4] K. G. Kim, "Book review: deep learning," *Healthcare Informatics Research*, vol. 22, no. 4, p. 351, 2016, doi: 10.4258/hir.2016.22.4.351.
[5] A. Esteva *et al.*, "A guide to deep learning in healthcare," *Nature Medicine*, vol. 25, no. 1, pp. 24–29, Jan. 2019, doi: 10.1038/s41591-018-0316-z.
[6] H. S. Das and P. Roy, "A deep dive into deep learning techniques for solving spoken language identification problems," in *Intelligent Speech Signal Processing*, Elsevier, 2019, pp. 81–100.
[7] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Interspeech 2014*, Sep. 2014, pp. 338–342, doi: 10.21437/Interspeech.2014-80.
[8] C. C. Aggarwal, *Neural networks and deep learning*. Cham: Springer International Publishing, 2018.
[9] C. Tian, J. Ma, C. Zhang, and P. Zhan, "A deep neural network model for short-term load forecast based on long short-term memory network and convolutional neural network," *Energies*, vol. 11, no. 12, p. 3493, Dec. 2018, doi: 10.3390/en11123493.

[10] J. Kim, U. Kang, and Y. Lee, "Statistics and deep belief network-based cardiovascular risk prediction," *Healthcare Informatics Research*, vol. 23, no. 3, p. 169, 2017, doi: 10.4258/hir.2017.23.3.169.
[11] A. Korotcov, V. Tkachenko, D. P. Russo, and S. Ekins, "Comparison of deep learning with multiple machine learning methods and metrics using diverse drug discovery data sets," *Molecular Pharmaceutics*, vol. 14, no. 12, pp. 4462–4475, Dec. 2017, doi: 10.1021/acs.molpharmaceut.7b00578.
[12] H. Mohsen, E.-S. A. El-Dahshan, E.-S. M. El-Horbaty, and A.-B. M. Salem, "Classification using deep learning neural networks for brain tumors," *Future Computing and Informatics Journal*, vol. 3, no. 1, pp. 68–71, Jun. 2018, doi: 10.1016/j.fcij.2017.12.001.
[13] S. I. Ayon and M. M. Islam, "Diabetes prediction: a deep learning approach," *International Journal of Information Engineering and Electronic Business*, vol. 11, no. 2, pp. 21–27, Mar. 2019, doi: 10.5815/ijieeb.2019.02.03.
[14] A. Massaro, V. Maritati, D. Giannone, D. Convertini, and A. Galiano, "LSTM DSS automatism and dataset optimization for diabetes prediction," *Applied Sciences*, vol. 9, no. 17, p. 3532, Aug. 2019, doi: 10.3390/app9173532.
[15] N. P. Tigga and S. Garg, "Prediction of type 2 diabetes using machine learning classification methods," *Procedia Computer Science*, vol. 167, pp. 706–716, 2020, doi: 10.1016/j.procs.2020.03.336.
[16] R. Shwartz-Ziv and A. Armon, "Tabular data: deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, May 2022, doi: 10.1016/j.inffus.2021.11.011.
[17] H. Hairani and D. Priyanto, "A new approach of hybrid sampling SMOTE and ENN to the accuracy of machine learning methods on unbalanced diabetes disease data," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 8, 2023, doi: 10.14569/IJACSA.2023.0140864.
[18] H. Chu *et al.*, "Roles of anxiety and depression in predicting cardiovascular disease among patients with type 2 diabetes mellitus: a machine learning approach," *Frontiers in Psychology*, vol. 12, Apr. 2021, doi: 10.3389/fpsyg.2021.645418.
[19] M. A. Jabbar, S. Samreen, and R. Aluvalu, "The future of health care: machine learning," *International Journal of Engineering and Technology(UAE)*, vol. 7, no. 4, pp. 23–25, 2018, doi: 10.14419/ijet.v7i4.6.20226.
[20] S. Shamshirband, M. Fathi, A. Dehzangi, A. T. Chronopoulos, and H. Alinejad-Rokny, "A review on deep learning approaches in healthcare systems: Taxonomies, challenges, and open issues," *Journal of Biomedical Informatics*, vol. 113, p. 103627, Jan. 2021, doi: 10.1016/j.jbi.2020.103627.
[21] Q. A. Al-Radaideh, F. M. Smadi, S. Al-Azzam, E. M. Al-Shawakfa, and A. M. Smadi, "Using machine learning to predict drug-related problems for diabetes patients," in *2022 13th International Conference on Information and Communication Systems (ICICS)*, Jun. 2022, pp. 199–204, doi: 10.1109/ICICS55353.2022.9811122.
[22] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.
[23] J. Brownlee, "Deep learning with Python: develop deep learning models on Theano and TensorFlow," *Machine Learning Mastery*, pp. 1–17, 2017.
[24] D. Takeuchi, K. Yatabe, Y. Koizumi, Y. Oikawa, and N. Harada, "Real-time speech enhancement using equilibriated RNN," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 851–855, doi: 10.1109/ICASSP40776.2020.9054597.
[25] S. Skansi, *Introduction to deep learning*. Cham: Springer International Publishing, 2018.

BIOGRAPHIES OF AUTHORS

Fatima M. Smadi received the B.Sc. degree in computer information systems from Jordan University of Science and Technology, Irbid, Jordan, in 2016 and M.Sc. degree in computer information systems from Yarmouk University, Irbid, Jordan, in 2021. Her research interest includes data mining, healthcare applications for artificial intelligence, machine learning, and deep learning. She can be contacted at email: fatima.smadi13@gmail.com.

Qasem A. Al-Radaideh is a professor of data science and artificial intelligence at Yarmouk University. Currently, Dr. Al-Radaideh is the dean for the Faculty of Information Technology and Computer Sciences at Yarmouk University. Dr. Al-Radaideh received his Ph.D. in the data mining field from the University Putra Malaysia (UPM) in 2005. He received the Hisham Hijjawi Award for Applied Science – ITC track for the year 2011. His research interest includes data mining and knowledge discovery in database; rough set-based knowledge reduction and classification, Arabic language computation, natural language processing, and information retrieval. He has several publications in the areas of data mining and Arabic language computation. In addition, Dr. Al-Radaideh was involved in several local and international Joint projects such as Erasmus + and TEMPUS projects. He can be contacted at email: qasemr@yu.edu.jo.