Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

# Pediatric Diabetes Prediction Using Deep Learning

**Abeer El-Sayed El-Bashbishy** ( abeerelbashbishy@gmail.com )
Mansoura University
**Hazem El-Bakry**
Mansoura University

**Article**

**Keywords:** Deep Learning, Classification, Cross Validation, Prediction, Accuracy

**Posted Date:** July 7th, 2023

**DOI:** https://doi.org/10.21203/rs.3.rs-3146306/v1

**License:** This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

# Abstract

The present study proposes a novel technique for the early prediction of diabetes with the utmost accuracy. Recently, the contemporary methodologies of artificial intelligence and in particular Deep Learning (DL), have proven to be expeditious in the diagnosis of diabetes. The model that is supported has been constructed with the implementation of two hidden layers and a multitude of epochs of Deep Learning Neural Network (DLNN) utilizing the Multi-Layer Perceptron (MLP) technique. We proceeded to meticulously adjust the hyperparameters within the fully automated DLNN architecture, with the aim of optimizing data pre-processing, classification and prediction. This was accomplished by a novel dataset of Mansoura University Children's Hospital Diabetes (MUCHD), which allowed for a more comprehensive evaluation of the system's performance. The system is validated and tested on a sample of 548 patients, each exhibiting 18 significant features. Various validation metrics were employed to ensure the accuracy and reliability of the results like K-folds, leave-one-subject-out and cross-validation approaches with various statistical measures of accuracy, f-score, precision, sensitivity, specificity and dice similarity coefficient. The high-performance level of the proposed system can help clinicians to accurately diagnose health and different diabetes grades with a remarkable accuracy rate of 99.8%. According to our analysis, the implementation of this method results in a noteworthy increase of 4.15% in overall system performance when compared to the current state-of-the-art. As such, we highly recommend the utilization of this method as a promising tool for forecasting diabetes.


# 1. Introduction

Diabetes rates exhibit an alarming increase annually particularly when left untreated. Diabetes is a chronic pathological condition that arises due to the excessive presence of glucose in the bloodstream. Individuals with diabetes mellitus are unable to produce an adequate amount of insulin, which is a hormone secreted by the pancreas. Insulin plays a crucial role in regulating glucose levels in the cells, which is essential for energy production. Many complications may occur if diabetes remains untreated such as visual impairment, cardiology problems, dental diseases, stroke and microvascular complications which lead to retinopathy, kidney failure and nerve damage [2]. Diabetes management tools are essential to monitor glucose and insulin levels. These tools include activity bands, glucose meters, continuous glucose monitoring devices, and sensor-augmented insulin pumps to control meal ingestion [3]. The primary objective of glucose management is to prevent undesired glycemia and related events [4]. It is important to note that there are three distinct types of diabetes: type 1, type 2 and gestational. In type 1 diabetes, the pancreas produces little or no insulin. Insulin therapy is a necessary treatment for diabetes. It is usually seen in young individuals (age < 30), as well as children. Type 2 diabetes, on the other hand, is usually caused by insulin resistance and is more prevalent in older patients (age > 65), as well as those who are obese. Gestational diabetes is hyperglycemia that occurs during pregnancy is also a concern [5]. So, early detection of diabetes is critical for timely treatment and to prevent the disease from progressing. To reduce the number of diabetes-related deaths, a useful DL technique can aid in automatic disease detection. DL is a new technology that extends the machine

learning (ML) technique, which is a sub-domain of artificial intelligence technology. In recent years, DL has made significant progress in medical applications including diabetes. DL allows the input of raw data and requires minimal feature engineering work on data pre-processing to learn representation automatically by exploiting DLNN. This helps the healthcare providers detect the disease in its early stages to detect and saves time for the clinicians [6].

There are three widely DLNN architectures see Fig. 1. These are Reinforcement Learning (RL), Supervised learning (SL) and Unsupervised learning (USL). RL is the preferred method for controlling problems with high-dimensional environments. SL, on the other hand, utilizes labeled input data during iterative model optimization and backward propagation. This method is Convolutional Neural Network commonly used in classification and regression tasks. In USL the predefined labels or classes of inputs are not required for the model training. Instead, the algorithm aims to infer hidden structures and representations from input datasets without supervision. USL is commonly used in cluster analysis, density estimation, and dimension reduction. The common types of SL include a recurrent neural network that captures temporal features by containing information from the previous time, as seen in speech recognition, natural language processing, Convolutional Neural Networks (CNN) that process multi-dimensional arrays to achieve high-performance image recognition tasks and Deep Multi-Layer Perceptron (DMLP). DMLP is a feed-forward neural network associated with a set of bias scalars, weight, and activation functions. The USL types include restricted Boltzmann machines which are used as feature detectors to extract representations from data, and autoencoders, where the training target is the same as the input [7].

The rest of this paper is organized as follows: in the second section, we present the related work. Section 3 details the proposed architecture and the corresponding features. The results and discussion are presented in section 4. The conclusion and future work are presented in section 5.

# 2. Related work

Traffic light Several scholars used DL technology to predict diabetes disease using the Pima Indian diabetes (PID) dataset, which consists of 9 attributes, and 768 records describing female patients [8].

Khanam and Foo [9] have also employed the PID dataset from the UCI repository to build a NN model with different hidden layers of various epochs. their observations indicate that the NN with two hidden layers provides an accuracy of 88.6% for the training data.

In addition, García-ordás et al. [10] have carried out data augmentation in samples using variational autoencoders and features sparse autoencoders on the PID dataset. They obtained an accuracy of 92.31% when the CNN classifier is trained jointly by the sparse autoencoders to predict diabetes disease.

Kumar et al. [11] employed the multi-layer feed-forward neural networks to classify diabetes using the PID dataset. They utilize various activation functions and learning algorithms to handle missing values for enhancing the classification accuracy of the diabetes dataset. In addition, they compared the

performance of two ML algorithms, Naive Bayes and Random Forest. The result of their study demonstrated a classification accuracy of 84.17%.

Krishnan [12] has presented an automatic classification of diabetes disease using DL techniques. They use MLP with the SVM classifier to achieve the best therapeutic management using the PID dataset. The results indicated that The MLP with the SVM classifier achieved a classification accuracy of 77.47%. in comparison, the SVM classifier alone only correctly classified with an accuracy of 65.1042%.

The study conducted by Perveen et al. [13] is a noteworthy contribution to the field of diabetes mellitus classification. The researchers employed AdaBoost and bagging ensemble techniques, with a decision tree as a base learner, with a data mining technique, to classify patients based on diabetes risk factors. This classification was carried out across three different ordinal adult groups in the Canadian Primary Care Sentinel Surveillance network. The overall performance of the AdaBoost method is superior to that of bagging with a decision tree.

Zhou, Myrzashova, and Zheng [14]. constructed a model utilizing hidden layers with dropout regularization to prevent the over-fitting of a DLNN. They adjusted multiple parameters with the binary cross-entropy loss function. achieving 94.02% accuracy for the diabetes-type dataset and 99.41% training accuracy for the PID dataset.

According to previous studies, it appears that there is still a notable problem with the accuracy of predicting diabetes, and we strive to elevate the accuracy to a higher level. Additionally, we must identify further features that can contribute to the precise prediction of diabetes. In addition, the study includes a small age group of patients, who may not possess the ability to express or possess a comprehensive understanding of their disease.

# 3. The Proposed Model for Predicting Diabetes Using DL

The disease classification technique is a method of assigning patients information into a binary category based on their medical attributes. This is accomplished by building a model through a train and test cycle. It is important to note that this technique is considered a supervised classification technique, as a set of predefined labeled diagnoses is provided as a training set to aid in the classification of new, unknown datasets. However, the size of the dataset can present a significant challenge due to the high dimension of irrelevant attributes, which causes poor classification performance. This problem can be handled through a pre-processing step be taken, utilizing perfect classification algorithms that are designed using statistics and control theory to analyze and retrieve knowledge from experience.

Following the proposed model, the DLNN is partitioned into three distinct layers. The initial layer known as the input layer, is responsible for receiving the raw input data from the domain without any prior computation. The input features are then subjected to a Neural Network (NN) that comprises one or more hidden layers. The input features are assigned an initialized weight value that ranges between (1 and −1). All the neurons in the hidden layers are attached to both the previous and the next layers. At the

hidden layers, each node provides an abstraction to the NN with all feature computations. The result is then transferred to the output layer, which assimilates the information learned through the hidden layer and provides the binary classification output. The parameters like weights and biases, are initially randomized and subsequently adjusted to optimize the prediction model. The proposed model will yield greater efficiency in the medical disease classification field. The model has been presented utilizing the DMLP technique, with fine-tuned hyperparameters that assist in building a robust DLNN technique for the classification of diabetes disease, based on the medical records of the MUCHD dataset.

The model under consideration comprises a sequence of phases, as presented in Fig. 2. The system includes a data-gathering phase, a data pre-processing and classification phase, a train/test phase, an evaluation phase, an optimizing phase, and a prediction phase. The data is sourced from the MUCHD dataset. During the pre-processing phase, it is of utmost importance to enhance the features of missing values and remove un-significant features. After that, the features are scaled to a normalized scale. The normalized pre-processed features are then fed to accept in the classifier. The model learns from the trained labeled data and tests its performance with unlabeled test data. Following The validation phase, the classifier performance is evaluated, followed by the optimizing phase and ultimately culminating in the prediction phase.

We present below the six phases of the proposed system in detail:

# 3.1. Data Collection Phase

The data is the very foundation of all the DLNN models. The model will be powerful when picking the right data. The quality of the data quantity is a crucial factor that cannot be overlooked. The reliability of data plays a significant role in all the model classification phases, which leads to useful predictions. During this phase, careful consideration is given to the selection of data features and the number of required samples. Assumptions are made regarding the most related data to diabetes disease. Hence good data lead to good performance and a successful model.

# 3.2. Dataset Availability Statement

The proposed system employs a MUCHD dataset of pediatric patients, encompassing individuals ranging from one year to nineteen years old both gender male and female. This dataset is obtained from the Mansoura University Children's Hospital repository system, Medicine Faculty, Dakahlia Governorate of Egypt, and is one of the main hospitals of Mansoura University. The data is gathered through examination, laboratory tests, patient medical records and admission notes.

The MUCHD dataset comprises 548 records classified into two distinct classes: diabetes and non-diabetes patients (healthy). These records are associated with 18 attributes, which include Age, Sex, Duration, Cholesterol, Creatinine, Glycated Haemoglobin(HbA1c), Insulin level, Post Prandial C-Peptide(PCPeptide), Fast C-Peptide(FCPeptide), two hours Post Prandial Blood Glucose(PBGlucose), Fast Blood Glucose(FBGlucose), Random Blood Glucose(RBGlucose), Blood Gases include Blood

Acidosis(PH), Bicarbonate(HCO3), Sodium(Na), Potassium(K) and the output target feature named Diagnosis. Figure 3 shows the ‘Diagnosis' output attribute which consists of the binary values. It has one value either 1 indicating the presence of diabetes with 397 patients, or 0 indicating non-diabetes patients with 151 patients.

Table 1 presents the MUCHD dataset of eighteen attributes along with their comprehensive descriptions. The diagnosis attribute is considered the dependent output variable, while the remaining seventeen attributes are regarded as independent input features.

Table 1
The Input Attributes of the MUCHD Dataset:

| Attribute | Description |
|---|---|
| **Age** | Age (Years). |
| **Sex** | Gender. |
| **Duration** | period time of diabetes symptoms per week. |
| **Cholesterol** | Cholesterol. |
| **Creatinine** | Creatinine. |
| **Acetone** | Acetone. |
| **HbA1c** | Hemoglobin(A1c). |
| **Insulin** | Insulin Level. |
| **PCPeptide** | Post Prandial C-Peptide. |
| **FCPeptide** | Fast C-Peptide. |
| **PBGlucose** | Post-Prandial Blood Glucose. |
| **FBGlucose** | Fast Blood Glucose. |
| **RBGlucose** | Random Blood Glucose. |
| **PH** | Blood Acidosis of Blood Gases. |
| **HCO3** | Bicarbonate of Blood Gases. |
| **Na** | The Sodium of Blood Gases. |
| **K** | Potassium of Blood Gases. |
| **Diagnosis** | Diagnosis of Diabetes is 1 for a positive test and 0 for a negative test. |

# 3.3. Data Pre-processing & Classification Phase

This step is performed in the NN. This phase is utilized to ensure that the input data is presented in a clear and organized format. The primary advantage of feature extraction is its ability to identify the most

effective features for the model classifier to learn the representation [15]. certain errors may arise due to human mistakes during the data collection phase, resulting in labeling errors.

# 3.3.1. Formatting

The dataset might not be in the right format like a database or CSV file. To address this, we have meticulously prepared the MUCHD dataset in CSV file format.

# 3.3.2. Missing Values

Dealing with noisy missing values poses a significant challenge when gathering data for DL techniques that extremely land with the perfect dataset which will probably take a significant chunk of time. Missing data samples often arise from errors in data collection as a blank space on the diagnosis feature that is not applicable [20]. The missing values are typically denoted by Nan or null indicators. It is necessary to delete redundant columns and clean individual columns. Before inputting the data into the model, certain issues must be addressed, such as some algorithms unable to handle the missing values. consequently, there are two recommended approaches to tackle this problem. The first one involves eliminating the samples of the missing values, but it is risky to delete relevant information. the second one is to impute the missing values by replacing them with the mean value for each input feature. In Fig. 3, the missing values appear as white blank values.

The feature selection method provides the highest correlated values, reduces the execution time and avoids data over-fitting [16]. The feature selection is used to reduce the feature set dimensionality by removing the least correlated attributes. These leads to improve performance efficiency and decreases the computational requirements. We investigate the MUCHD dataset using Pearson's correlation method applied to the Python programming language. The coefficient values remain in the range between − 1 and 1. A value below − 0.5 is indicative of a weak correlation, while a value above 0.5 indicates a strong correlation. A value of zero, on the other hand, signifies no correlation. The irrelevant or outlier attributes are removed from the dataset. Table 2 provides a comprehensive overview of the most significant relevant features of the diagnosis attribute, namely HbA1c, Insulin, PCPeptide, FCPeptide, PBGlucose, FBGlucose, and RBGlucose. The missing values for these features are calculated as shown in Table 3.

Table 2
The Correlation Between Input/output Attributes:

| Attributes | Correlation Coefficient |
|---|---|
| Age | 0.07 |
| Sex | 0.059 |
| Duration | 0.24 |
| Cholesterol | 0.12 |
| Creatinine | 0.22 |
| Acetone | 0.24 |
| HbA1c | 0.71 |
| Insulin | 0.61 |
| PCPeptide | 0.37 |
| FCPeptide | 0.31 |
| PBGlucose | 0.86 |
| FBGlucose | 0.82 |
| RBGlucose | 0.73 |
| PH | 0.23 |
| HCO3 | 0.16 |
| Na | 0.16 |
| K | 0.18 |

Table 2
The Missing Values of The MUCHD Dataset:

| Attributes | Missing Values |
|---|---|
| Age | 0 |
| Sex | 0 |
| Duration | 0 |
| Cholesterol | 0 |
| Creatinine | 0 |
| Acetone | 0 |
| HbA1c | 194 |
| Insulin | 76 |
| PCPeptide | 156 |
| FCPeptide | 145 |
| PBGlucose | 66 |
| FBGlucose | 78 |
| RBGlucose | 61 |
| PH | 0 |
| HCO3 | 0 |
| Na | 0 |
| K | 0 |
| Diagnosis | 0 |

# 3.3.3. Sampling

Sometimes you may have too much data over what you required. This predicament can result in increased computational and memory requirements. We take into consideration the appropriate number of samples, which will faster the processing steps involved in exploring and prototyping the solution. The size of data samples is determined according to the requirements for faster convergence and to reduce disk space.

# 3.3.4. Feature Scaling

In the pre-processing phase, it is crucial to take a specific step. The majority of DLNN algorithms perform much better when dealing with features that are on the same scale [21]. The features are implemented to

reduce uncertainty, incorrect results, or cost/processing time. One effective method for achieving this is feature scaling, which involves transforming the smallest value of any feature to 0.0 and the largest value to 1.0. there are two common techniques for feature scaling. The first is normalization, which rescales features to a range between 0.0 and 1.0. The second technique is standardization, which involves centering the field at a mean of 0.0 and a standard deviation of 1.0. The columns feature has the same parameters as a standard normal distribution which is zero mean and unit variance. This makes it much easier for the learning algorithms to learn the weights of the parameters. In addition, it keeps valuable information on the outliers, thereby rendering the algorithms less susceptible to their influence. Generally, when we input data to DL algorithms, it is customary to manipulate the input data in such a way that the values are adjusted to a balanced scale. The missing values are substituted with the corresponding mean value after the normalization process. The normalization of the data is a crucial step in ensuring that the model can be generalized appropriately. Scaling the data is achieved through the utilization of Eq. 3.2.4.1 [17].

$$Z = \frac{X_i - \mu}{\sigma} \quad (3.2.4.1)$$

Where $X_i$ data is rescaled into Z with $\mu = 0$ and $\boldsymbol{\sigma} = 1$.

The statistical computations: mean, standard deviation (std) minimum (min) and maximum (max) values for all attributes before and after the normalization process are shown in Table 4 and Table 5.

Table 4
The Statistical Computations Before Normalization

| Attributes | Mean | Std | Min | Max |
|---|---|---|---|---|
| Age | 9.19 | 4.07 | 1 | 19 |
| Sex | 1.48 | 0.50 | 1 | 2 |
| Duration | 57.41 | 90.02 | 0.3 | 768 |
| Cholesterol | 167.22 | 44.47 | 46 | 518 |
| Creatinine | 0.69 | 0.30 | 0.2 | 5.3 |
| Acetone | 2.39 | 0.52 | 1 | 6.1 |
| HbA1c | 7.60 | 1.68 | 0 | 14.6 |
| Insulin | 2.01 | 2.78 | 0.1 | 65 |
| PCPeptide | 1.09 | 0.33 | 0 | 5 |
| FCPeptide | 0.80 | 0.15 | 0.1 | 3 |
| PBGlucose | 277.09 | 95.60 | 80 | 640 |
| FBGlucose | 228.29 | 86.49 | 70 | 600 |
| RBGlucose | 414.71 | 75.10 | 39 | 750 |
| PH | 7.32 | 0.23 | 2.3 | 8 |
| HCO3 | 20.80 | 5.11 | 2.2 | 44 |
| Na | 137.59 | 6.83 | 124.8 | 235.4 |
| K | 3.91 | 0.52 | 0.4 | 6.8 |

Table 5 presents the statistical computations: mean value for the most significant features.

Table 5
The Statistical Computations after Normalization:

| Attributes | Mean | Std | Min | Max |
|---|---|---|---|---|
| Age | 9.19 | 4.07 | 1 | 19 |
| Sex | 1.478 | 0.50 | 1 | 2 |
| Duration | 37.62 | 94.07 | 0 | 768 |
| Cholesterol | 102.84 | 92.80 | 0 | 518 |
| Creatinine | 0.44 | 0.45 | 0 | 5.3 |
| Acetone | 0.65 | 1.18 | 0 | 6.1 |
| HbA1c | 7.59 | 1.67 | 0.038 | 14.6 |
| Insulin | 2.04 | 2.78 | 0.05 | 65 |
| PCPeptide | 0.14 | 0.48 | 0 | 5 |
| FCPeptide | 0.05 | 0.24 | 0 | 3 |
| PBGlucose | 277.05 | 95.60 | 80 | 640 |
| FBGlucose | 228.27 | 86.49 | 70 | 600 |
| RBGlucose | 414.73 | 75.10 | 39 | 750 |
| PH | 4.59 | 3.56 | 0 | 8 |
| HCO3 | 12.75 | 11.35 | 0 | 44 |
| Na | 77.83 | 68.60 | 0 | 235.4 |
| K | 2.039 | 2.03 | 0 | 6.8 |

# 3.3.5. The Construction of a DLNN

1. The input layer passes dataset features without any computation to the hidden layers.
2. The hidden layers perform the computations and pass the information to the output layer.
3. The output layer represents the results after training a newly created model.

# 3.3.6. Train/Test

The dataset is split into three subsets a train set, a validation set and a test set. The Train set is utilized for training the model, while the validation set is employed to evaluate its performance. Finally, the test set serves as the ultimate benchmark, allowing us to assess the model's effectiveness in real-world scenarios. We divide the MUCHD dataset into smaller batches with 70% of the data allocated for training and validation purposes, validation, and 30% reserved for testing data. Then we feed those beaches into the DLNN technique.

**Algorithm 1: The Train Algorithm:**

1: **Start**

2: The input features are fed to the input layer of the NN.

3: Randomly initialize the hyperparameters of the NN.

4: Compute the output of each neuron in the hidden layers and output layers.

5: Calculate the gradients of the hyperparameters.

6: Apply the activation function to the output of each neuron computed in step 4.

7: Update each parameter using the gradient optimization.

8: Update the weights of the NN based on the propagation approach.

9: Repeat steps 3–8 until all conditions are completed.

If the output is > = 0.5 then

diagnosis = "diabetic"

else

diagnosis = "non-diabetic"

End If.

10: **End**

The feature vector is directly fed into the input nodes. These nodes have initialized a random number of weights and fine-tuned parameters to the DMLP. Each node generates an output using an activation function. The outputs are then connected to the next hidden layers. The activation functions vary across the different hidden layers. Then, the features are retrieved and concatenated to create a new feature vector. The new feature vector is then received by the classifier to determine the confidence of each relation. Then the classifier produces a binary output vector. Training the classifier is the most crucial aspect of the classification process. The role of this phase is to generate a model by training it with a predefined diagnosis class label, that will be used later to classify unlabeled diagnoses. The training data is essentially a means of learning the classifier model. In the feed process, after the data has been fed, forward propagation occurs. The losses are compared against the loss function, and the parameters are adjusted accordingly based on the incurred loss. throughout the training process, the algorithm searches for patterns that correlate with the desired output, as declared in Eq. 3.3.6.1.

In a neural network, hidden neurons perform a calculation involving the weighted sum according to (1) of its input, along with the addition of a bias term, and then decide whether it should be ‘fired’ or not. So, a specific neuron will be as follows.

The value of Y can be -∞ to +∞. So, the neuron can’t decide whether it will fire or not. Here the activation function is used to decide where the neuron will fire or not. We have used ReLU as an activation function [18].

$$Y = \sum (input * Weight) + bias \quad (3.3.6.1)$$

Where Y is the activation function, input means input features.

The maximum number of hidden neurons that won't result in over-fitting [19] is calculated as shown in Eq. 3.3.6.2.

$$\mathrm{Nh} = \mathrm{Ns}\,(\alpha^{*}\,(\mathrm{Ni} + \mathrm{No})) \quad (3.3.6.2)$$

$\mathrm{Nh}$ = 548 / (4*(17 + 1) =7.61 ~ 8 neurons

Where Nh is the number of neurons in the hidden layers, Ns is the total number of samples in the MUCHD dataset, α is an arbitrary scaling factor that usually has any value between 2 to 10 and No is the number of the output layers.

The NN is trained using the gradient descent algorithm to control the range of weight values throughout the training phase. We used the Rectified Linear Units (ReLU) activation function in the first hidden layer, as depicted in Fig. 4 (a), and a Sigmoid activation function in the second layer is declared in Fig. 4 (b). Generally, the sigmoid and ReLU activation functions are employed for binary classification outputs, whereas the Softmax and the Logarithmic activation function are typically utilized for the multi-classification output [20].

In the Validation process, we run the suggested model on various subsets of both training and validation datasets, then we get model quality measures [23]. This step can be further categorized into two techniques: exhaustive and non-exhaustive cross-validation. In the exhaustive cross-validation approach, training, and testing are performed on all data samples. A portion of the dataset is designated for testing purposes, while the remaining portions are used for training. It is also divided into:

- Leave-P-Out Validation: leave p data points out of training data [21].
- Leave One Out Cross-Validation: the folds count numbers are equal to the total number of the dataset samples [21].

In a non-exhaustive cross-validation approach, the dataset is divided into multiple subsets, each consisting of several blocks. Each block is divided into subsets of training samples and test samples. So, the overall result is the average of all test samples. It is divided to:

- K-fold Cross-Validation involves splitting the data into k subsets. One of the k subsets is used as the validation set, while the other k-1 subsets are used as the training set [22–23].
- The holdout method removes a portion of the training dataset and sends it to the model to train on the rest of the dataset [21].
- Stratified K-fold Cross-Validation works on an imbalanced dataset. Each fold contains approximately the same strata of samples for each output class.

In our model, the data is divided into 5 pieces. Each fold contains 20% of the full dataset portion. We employ K- folds = 5, which means the training portions are 4/5 and only one block is used for validation. In iteration one, we designate the first fold as the validation set and utilize the remaining folds for training. This is valuable for quantitative evaluation to measure the model quality based on a 20% holdout set. We repeat this process, using each fold once as the holdout set according to the number of iterations and the error is averaged see Fig. 5.

The stratify parameter is a valuable tool for addressing imbalances in data. It means that if the number of diabetes patients is 75% of class one diabetes, and the non-diabetes patients, comprise 25% of class zero, then the stratify parameter will make sure that the same percentage portion of the data split is remaining true. The validation structure is presented in Fig. 6.

The data with the unlabeled classes has been prepared previously in the pre-processing phase. The mapping function will be employed to classify unseen or unlabeled data to determine which label it belongs to. In the test phase, the model finds the data features that correlate to a defined class. Then the classification technique is tasked with the responsibility of assigning an accurate class label diagnosis to unlabeled cases specifically distinguishing between diabetes and non-diabetes disease. We train the DLNN sequential model using various hyperparameters. By experimenting with different values to determine the best-fit parameter such that epochs (the number of training times) is of three values (10, 50, 100), batch size (the number of sub-samples fed to NN after updating the parameter) are of six values (10, 20, 40, 60, 80, 100), optimizer (a computed past squared gradients) are of seven values ('SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam'), activation function ('softmax', 'softplus', 'softsign', 'ReLU', 'tanh', 'sigmoid', 'hard_sigmoid', 'linear'), weight constraint consists of five values (0.01,0.02,0.03,0.04,0.05), neurons are of eleven values (1, 5, 6, 7, 8, 10, 15, 17, 20, 25, 30), momentum are of six values (0.0, 0.2, 0.4, 0.6, 0.8, 0.9) and learning rates are of five values (0.001, 0.01, 0.1, 0.2, 0.3)). We utilized the Keras and Tensor-Flow libraries to create a NN of sequential models. In NN, the Stochastic Gradient Descent (SGD) optimizer is required to reduce the output error during the feed-forward approach. We used the train-test split and cross-validation functions from the scikit-learn library to perform the splitting task.

# 3.4. Evaluation Phase

The evaluation phase allows us to test the model on the validation set to accurately assess its performance in real-world scenarios. The normality assumption states that the difference between the actual output and the predicted output of a model is normally distributed and checked by histograms or a

standard normal distribution. We train the proposed DLNN model using the updated parameters specific to the MUCHD dataset. We also express the classification loss function by employing binary cross-entropy, which is used to compute the system error see Eq. (3.3.7.1).

$$Loss function = \mathrm{Y} - \mathrm{Ypred} \quad (3.3.7.1)$$

Where Y is the actual output and $\mathrm{Ypred}$ is the predicted output.

# 3.5. Optimizing Phase

After completing the evaluation process, there is a high chance that our model could be optimized further. We began by initializing the weights and biases with initial values. Then we meticulously fine-tuned the back-propagation process. The challenge in training the DLNN model lies in the careful selection of the most significant features. To overcome the issue of over-fitting, we fine-tuned the hyperparameters of the DLNN by conducting different experiments only on 70% of the dataset. These parameters encompass the number of hidden layers in the network, the number of neurons, and the activation function used to estimate the output of each neuron. One way to achieve this is by repeatedly utilizing the model and increasing the number of epochs. The learning rate is a crucial parameter in training the DLNN model with a small positive value within the range between 0.0 and 1.0. A smaller learning rate necessitates more training epochs, as it leads to smaller changes made in the weights during each update. These values have a significant impact on both accuracy of the model and the duration of the training process, especially complex models. The most common problems are encountered when a model performs well on the training data but fails to generalize effectively to unseen data. This happens when the model learns a pattern specific to the training dataset that isn't relevant to other unseen data. There are two ways to avoid over-fitting. The first and most effective solution is to acquire more data. However, Reducing the network capacity excessively will result in under-fitting, rendering the model incapable of learning the relevant patterns in the trained data. Unfortunately, there are no magical formulas to determine the ideal balance. It necessitates testing and evaluation by manipulating the number of parameters and observing subsequent performance. The other way to avoid over-fitting is to apply weight regularization to the model. A common way can be achieved by assessing the complexity of the network and ensuring that the weights are constrained to only small values. Regularizing the distribution of weight values involves incorporating a cost into the loss function of the NN. This cost comes in two ways. The first is called L1 regularization, where the cost is regarded as the absolute value of the weight coefficient. The second is called L2 regularization, which adds cost based on the square value of the weight’s coefficient.

# 3.6. Prediction Phase

The output of the optimizing phase declares that the system has been successfully trained, tested, and is now ready to run the classification model for any new data with unknown class labels. This contributes to diabetes disease prediction to obtain the final recommended classifier.

The proposed technique can also be utilized to determine whether an individual has Type 1 diabetes or not. According to the C-Peptide test, diabetes types can also be defined easily. If the C-Peptide value is less than 0.2 nmol/l, then the diabetes patient will be Type 1. A high level of C-peptide can be an indicator of other diabetes types. Table 6 presents the DLNN classifier with the corresponding prediction value using the confusion matrix.

Table 6
Confusion matrix for the DLNN classifier.

| Method | | Predicted No | Predicted Yes |
|---|---|---|---|
| **DLNN** | Actual No | 151 | 0 |
| | Actual Yes | 1 | 396 |

The DLNN technique is executed within the Jupyter Notebook, utilizing the Python language for programming purposes.

# 4. Results and Evaluation

Early detection of diabetes disease is crucial for improving the overall health and well-being of patients. Recently, The DLNN technique has been employed to develop accurate prediction models for diabetes. The suggested model is executed using the MUCHD dataset. The dataset is sourced from Mansoura University Children's Hospital repository system. Comprised of 548 samples, each sample contains 18 features including Age, Sex, Duration, Cholesterol, Creatinine, HbA1c, Insulin, PCPeptide, FCPeptide, PBGlucose, FBGlucose, RBGlucose, PH, HCO3, Na and K. There are 151 samples labeled as non-diabetes patients (class 0) and 397 samples belonging to the diabetes patients (class 1). The supported DLNN technique consists of these five phases: pre-processing the MUCHD dataset to clean and remove outlier data, normalizing the dataset, replacing missing values with the mean values for each feature, training the DLNN model with optimized tuning hyperparameters, and splitting the dataset into three parts: training, validation, and testing. The training and validation dataset consists of 384 samples, which is 70% of the total, while the remaining 164 samples, equivalent to 30% are reserved for testing the classifier. The confusion matrix is presented in Table 7. The children patients who have a diabetes disease and are predicted as having a diabetes disease are called True Positives (TP), the children patients who are non-diabetes disease but are predicted as having diabetes disease are called False Positives (FP), the children patients who are non-diabetes disease and they are predicted as non-diabetes disease are called True Negatives (TN) and the children patients who have diabetes disease and they are predicted as non-diabetes disease are called False Negatives (FN) [24].

Table 7
The Confusion matrix.

| | Predicted No(0) | Predicted Yes(1) |
|---|---|---|
| **Actual No (0)** | TN | FP |
| **Actual Yes (1)** | FN | TP |

TN = True Negative, FP = False Positive, FN = False Negative and TP = True Positive.

The classification algorithms are used to evaluate the performance efficiency in terms of Accuracy, Precision, Recall and F1-Score, Training Scores, Mean Square Error and R2 Score for all samples as shown in Table 8.

The Specificity, Dice Similarity Coefficient, Precision, Recall, F1-Scoreand Accuracy performance measures matrices can be calculated from Equations 4.1–4.6 [25].

The Specificity asks about how many normal cases are correctly predicted as shown in Eq. 4.1:

$$Specificity = \frac{TN}{TN+FP} \quad (4.1)$$

Dice Similarity Coefficient determines how many samples are classified correctly as shown in Eq. 4.2:

$$Dice\ Similarity\ Coefficient = \frac{2*TP}{2*TP+FP+FN} \quad (4.2)$$

Precision Score provides the accuracy of positive diabetes predictions as shown in Eq. 4.3:

$$Precision = \frac{TP}{TP+FP} \quad (4.3)$$

Recall Score (Sensitivity) is the ratio of correctly predicted positive cases to all samples as shown in Eq. 4.4:

$$Recall = \frac{TP}{TP+FN} \quad (4.4)$$

F1-Score is the weighted average of Precision and Recall as shown in Eq. 4.5:

$$F1 = \frac{2*(Precision*Recall)}{Precision+Recall} \quad (4.5)$$

Accuracy indicates DL classifier correctness in the diagnosis process of whether the patient has diabetes or non-diabetes see Eq. 4.6:

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (4.6)$$

Table 8

The DLNN technique performance measures using accuracy, Precision, Recall, F1-Score, Training Score, Mean Squared Error and R2 Score.

| Test Method | DLNN Result |
|---|---|
| **Accuracy** | 99.8% |
| **Specificity** | 0.99 |
| **Dice Similarity Coefficient** | 0.98 |
| **Precision** | 0.99 |
| **Recall** | 0.99 |
| **F1-Score** | 0.99 |
| **Training Score** | 100.0 |
| **Mean Squared Error** | 0.001 |
| **R2 Score** | 0.99 |

Based on the conducted experiments, the hyperparameters configurations are tuned to achieve optimal results as batch size: 10, epochs:50, the optimizer: SGD, activation function: ReLU in the first hidden layer and the sigmoid activation function in the second hidden layer with a weight constraint: 3, the learning rate: 0.01 and momentum: 0.4. The DLNN model summary of tuning hyperparameters is reported from Table 9 to Table 13.

Table 9
Batch size, Epochs and accuracy

| Test | Batch size | Epochs | Accuracy |
|---|---|---|---|
| 1 | 10 | 10 | 80.4% |
| | 10 | 50 | 85.7% |
| | 10 | 100 | 79.8% |
| 2 | 20 | 10 | 79.5% |
| | 20 | 50 | 83.3% |
| | 20 | 100 | 82% |
| 3 | 40 | 10 | 77.3% |
| | 40 | 50 | 80% |
| | 40 | 100 | 81.5% |

Table 10
Optimizer and Accuracy

| Test | Optimizer | Accuracy |
|---|---|---|
| 1 | SGD | 100% |
| 2 | RMSprop | 96% |
| 3 | Adagrad | 99% |
| 4 | Adadelta | 34% |
| 5 | Adam | 95% |
| 6 | Adamax | 92% |
| 7 | Nadam | 91% |

Table 11
Activation Function and Accuracy

| Test | Activation Function | Accuracy |
|---|---|---|
| 1 | Softmax | 99% |
| 2 | Softplus | 94.5% |
| 3 | Softsign | 99% |
| 4 | ReLU | 100% |
| 5 | sigmoid | 100% |
| 6 | Tanh | 98.9% |
| 7 | hard_sigmoid | 98% |
| 8 | Linear | 33% |

Table 13
Learning rate, Momentum and Accuracy

| Test | Learning rate | Momentum | Accuracy |
|---|---|---|---|
| 1 | 0.001 | 0.0 | 96% |
| | | 0.2 | 96% |
| | | 0.4 | 100% |
| | | 0.6 | 100% |
| | | 0.8 | 94% |
| | | 0.9 | 100% |
| 2 | 0.01 | 0.0 | 100% |
| | | 0.2 | 99% |
| | | 0.4 | 100% |
| | | 0.6 | 100% |
| | | 0.8 | 100% |
| | | 0.9 | 100% |
| 3 | 0.1 | 0.0 | 100% |
| | | 0.2 | 99% |
| | | 0.4 | 100% |
| | | 0.6 | 100% |
| | | 0.8 | 100% |
| | | 0.9 | 100% |
| 4 | 0.2 | 0.0 | 99.6% |
| | | 0.2 | 100% |
| | | 0.4 | 100% |
| | | 0.6 | 100% |
| | | 0.8 | 100% |
| | | 0.9 | 100% |
| 5 | 0.3 | 0.0 | 100% |
| | | 0.2 | 100% |
| | | 0.4 | 99% |

| Test | Learning rate | Momentum | Accuracy |
|---|---|---|---|
| | | 0.6 | 100% |
| | | 0.8 | 100% |
| | | 0.9 | 66% |

The proposed system with the DLNN classification algorithm is recommended to predict any other binary classification disease systems.

In this section, we provide a comparison between the proposed system and the other methods found in the literature. In 2021, Khanam and Foo [9] achieved an accuracy of 88.6%, which falls short of ours by a mere 1.2%. In 2021, García-ordás et al. [10] achieved an accuracy of 92.31%, which is 7.49% lower than ours. In 2018, Kumar et al. [11] achieved an accuracy of 84.17%, trailing behind ours by 15.63%. In 2021, Krishnan. [12] achieved an accuracy of 77.47%, which is a substantial 22.33% lower than ours. In 2020, Zhou, Myrzashova, and Zheng [14] achieved an accuracy of 94.02%, which is less than ours by 5.96%. Based on these previous studies, the proposed system achieved the best results compared to the others.

The advantages of the proposed DLNN system are as follows. First, it employs a comprehensive approach to diagnose pediatric diabetes in small ages who may struggle to articulate their symptoms accurately. This ensures that no potential signs of the disease are overlooked. Second, the system is versatile and can be applied to different real datasets that involve different diseases. This adaptability allows for a wider range of medical conditions to be analyzed and diagnosed effectively Third, it saves time, memory, computational cost and effort by using the pre-trained DLNN model. Finally, it is one of the few studies conducted specifically on pediatrics. In contrast, most of the conducted research in analyzing diabetes disease applied to adults with a small number of significant features that can’t detect effectively diabetes disease correctly.

# 5. Conclusion

In this research, a groundbreaking study has been presented for the early detection of diabetes disease. We suggested a new model using the DL technique for diabetes disease classification and prediction. Experimental results have unequivocally confirmed the efficiency of the designed system, boasting an impressive accuracy rate of 99.8%. To evaluate the effectiveness of our model, we conducted a meticulous analysis on a dataset consisting of 548 children patients using the MUCHD dataset, which encompasses 18 attributes. The proposed system constructed a robust DLNN model through a follow-up of phases. In the preprocessing phase, the dataset is cleaned, and the missing values are replaced by the mean value for each input feature of the dataset. Then the train/test phase is presented, followed by the evaluation phase, optimizing phases, and ended by the prediction phase. Several hyperparameters are fine-tuned for generating an ideal model. We employ two activation functions: ReLU and sigmoid, which play a vital role in the DLNN diabetes prediction model. The model demonstrated good results in various quality measures, including Accuracy, Precision, Recall, F1-Score, Training Score, Mean Squared Error, and

R2 Score are essential metrics that offer valuable insights into the performance of a technique. These metrics ensure that the supported technique works well rapidly.

The predicted diabetes information can be of great value as a warning signal for both young patients as well as pediatricians. This aids in making informed decisions and effectively managing the problem of diabetes. Additionally, the use of DL techniques can also solve feature extraction problems and achieve high success in the multi-label classification of problem-solving. The Simulation results demonstrate that the proposed model outperforms the existing models in terms of accuracy and effectiveness.

To achieve Further improvement in accuracy, it is recommended to train the model on a larger dataset. The work can be used to automate diabetes disease prediction with the help of some successful machine learning techniques that can be incorporated as a base learner in the proposed framework. Our research can also be extended to determine the types of diabetes disease. Furthermore, it is advisable to explore the potential enhancement of the proposed system by combining it with other medical scan images, such as those of the eye, chest, and cardiology, as well as the clinical biomarkers. furthermore, this system can be applied universally to diagnose different pediatric diabetes symptoms that have a significant impact on the overall well-being and quality of these young individuals.

# Declarations

### Data Availability

The dataset used is taken from Mansoura University Children's Hospital system with written approval.

### Compliance with Ethical Standards

This research received no specific grant from any funding agency.

### Conflict of Interest

All authors declare that they have no conflict of interest.

### Ethical approval

This article contains a study on the MUCHD dataset with pediatric participants performed by the authors.

# References

1. L. Fregoso-Aparicio, J. Noguez, L. Montesinos, and J. A. García-García, “Machine learning and deep learning predictive models for type 2 diabetes: a systematic review,” *Diabetol. Metab. Syndr.*, vol. 13, no. 1, 2021, doi: 10.1186/s13098-021-00767-9.
2. G. Swapna, R. Vinayakumar, and K. P. Soman, “Diabetes detection using deep learning algorithms,” *ICT Express*, vol. 4, no. 4, pp. 243–246, 2018, doi: 10.1016/j.icte.2018.10.005.

3. J. Freiburghaus, A. Rizzotti-Kaddouri, and F. Albertetti, “A deep learning approach for blood glucose prediction of type 1 diabetes,” *CEUR Workshop Proc.*, vol. 2675, pp. 131–135, 2020.
4. D. Care and S. S. Suppl, “Classification and diagnosis of diCare, D., & Suppl, S. S. (2018). Classification and diagnosis of diabetes: Standards of medical care in Diabetesd2018. Diabetes Care, 41(January), S13–S27. https://doi.org/10.2337/dc18-S002abetes: Standards of medical care,” *Diabetes Care*, vol. 41, no. January, pp. S13–S27, 2018, doi: 10.2337/dc18-S002.
5. M. F. Aslan and K. Sabanci, “A Novel Proposal for Deep Learning-Based Diabetes Prediction: Converting Clinical Data to Image Data,” *Diagnostics*, vol. 13, no. 4, 2023, doi: 10.3390/diagnostics13040796.
6. S. Albahra *et al.*, “Artificial intelligence and machine learning overview in pathology & laboratory medicine: A general review of data preprocessing and basic supervised concepts,” *Semin. Diagn. Pathol.*, vol. 40, no. 2, pp. 71–87, 2023, doi: 10.1053/j.semdp.2023.02.002.
7. T. Zhu, K. Li, P. Herrero, and P. Georgiou, “Deep Learning for Diabetes: A Systematic Review,” *IEEE J. Biomed. Heal. Informatics*, vol. 25, no. 7, pp. 2744–2757, 2021, doi: 10.1109/JBHI.2020.3040225.
8. D. Sisodia and D. S. Sisodia, “Prediction of Diabetes using Classification Algorithms,” *Procedia Comput. Sci.*, vol. 132, no. Iccids, pp. 1578–1585, 2018, doi: 10.1016/j.procs.2018.05.122.
9. J. J. Khanam and S. Y. Foo, “A comparison of machine learning algorithms for diabetes prediction,” *ICT Express*, vol. 7, no. 4, pp. 432–439, 2021, doi: 10.1016/j.icte.2021.02.004.
10. M. T. García-ordás, C. Benavides, J. A. Benítez-andrades, H. Alaiz-moretón, and I. García-rodríguez, “Computer Methods and Programs in Biomedicine Diabetes detection using deep learning techniques with oversampling and feature augmentation,” vol. 202, 2021, doi: 10.1016/j.cmpb.2021.105968.
11. S. Kumar, B. Bhusan, D. Singh, and D. Choubey, “Classification of Diabetes using Deep Learning,” no. Dl, pp. 651–655, 2020.
12. K. T. Krishnan, “Classification of Diabetes Using Deep Learning and SVM Techniques,” no. January, 2021, doi: 10.31782/IJCRR.2021.13127.
13. S. Perveen, M. Shahbaz, A. Guergachi, and K. Keshavjee, “Performance Analysis of Data Mining Classification Techniques to Predict Diabetes,” *Procedia Comput. Sci.*, vol. 82, no. March, pp. 115–121, 2016, doi: 10.1016/j.procs.2016.04.016.
14. H. Zhou, R. Myrzashova, and R. Zheng, “Diabetes prediction model based on an enhanced deep neural network,” *Eurasip J. Wirel. Commun. Netw.*, vol. 2020, no. 1, 2020, doi: 10.1186/s13638-020-01765-7.
15. S. Larabi-Marie-Sainte, L. Aburahmah, R. Almohaini, and T. Saba, “Current techniques for diabetes prediction: Review and case study,” *Appl. Sci.*, vol. 9, no. 21, 2019, doi: 10.3390/app9214604.
16. X. Ying, “An Overview of Overfitting and its Solutions,” *J. Phys. Conf. Ser.*, vol. 1168, no. 2, 2019, doi: 10.1088/1742-6596/1168/2/022022.
17. A. Berengolts and M. Lindenbaum, “On the distribution of saliency,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, no. May, 2004, doi: 10.1109/cvpr.2004.1315211.

18. S. Islam Ayon and M. Milon Islam, “Diabetes Prediction: A Deep Learning Approach,” *Int. J. Inf. Eng. Electron. Bus.*, vol. 11, no. 2, pp. 21–27, 2019, doi: 10.5815/ijieeb.2019.02.03.
19. Colchero, “Supporting Information Supporting Information,” *Aldenderfer, Mark S., Craig, Nathan M., Speak. Robert Jeff, Popelka-Filcoff, Rachel S.*, vol. 2, no. 1, pp. 1–5, 1997.
20. T. Szandała, “Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. Available: https://link.springer.com/chapter/10.1007/978-981-15-5495-7_11,” 2018.
21. I. K. Nti, O. Nyarko-Boateng, and J. Aning, “Performance of Machine Learning Algorithms with Different K Values in K-fold CrossValidation,” *Int. J. Inf. Technol. Comput. Sci.*, vol. 13, no. 6, pp. 61–71, 2021, doi: 10.5815/ijitcs.2021.06.05.
22. F. Y. H. Ahmed, Y. H. Ali, and S. M. Shamsuddin, “Using K-fold cross validation proposed models for SpikeProp learning enhancements,” *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 145–151, 2018, doi: 10.14419/ijet.v7i4.11.20790.
23. P. Hounguè and A. G. Bigirimana, “Leveraging Pima Dataset to Diabetes Prediction: Case Study of Deep Neural Network,” *J. Comput. Commun.*, vol. 10, no. 11, pp. 15–28, 2022, doi: 10.4236/jcc.2022.1011002.
24. S. P. Chatrati *et al.*, “Smart home health monitoring system for predicting type 2 diabetes and hypertension,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 3, pp. 862–870, 2022, doi: 10.1016/j.jksuci.2020.01.010.
25. T. Beghriche, M. Djerioui, Y. Brik, B. Attallah, and S. B. Belhaouari, “An Efficient Prediction System for Diabetes Disease Based on Deep Neural Network,” *Complexity*, vol. 2021, 2021, doi: 10.1155/2021/6053824.

# Figures

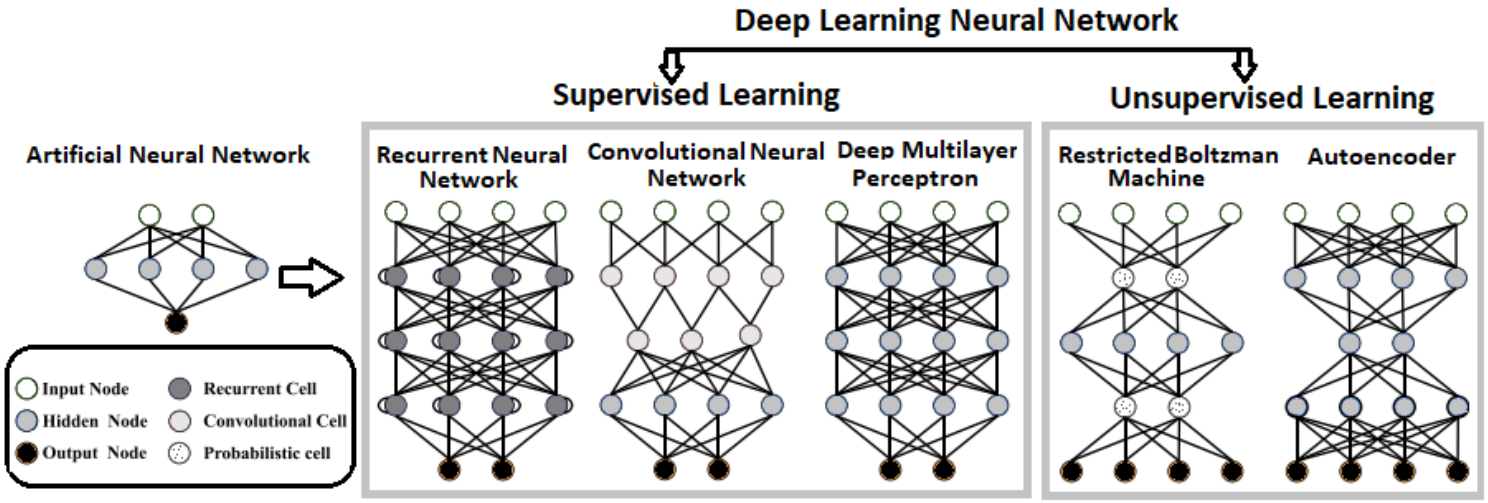
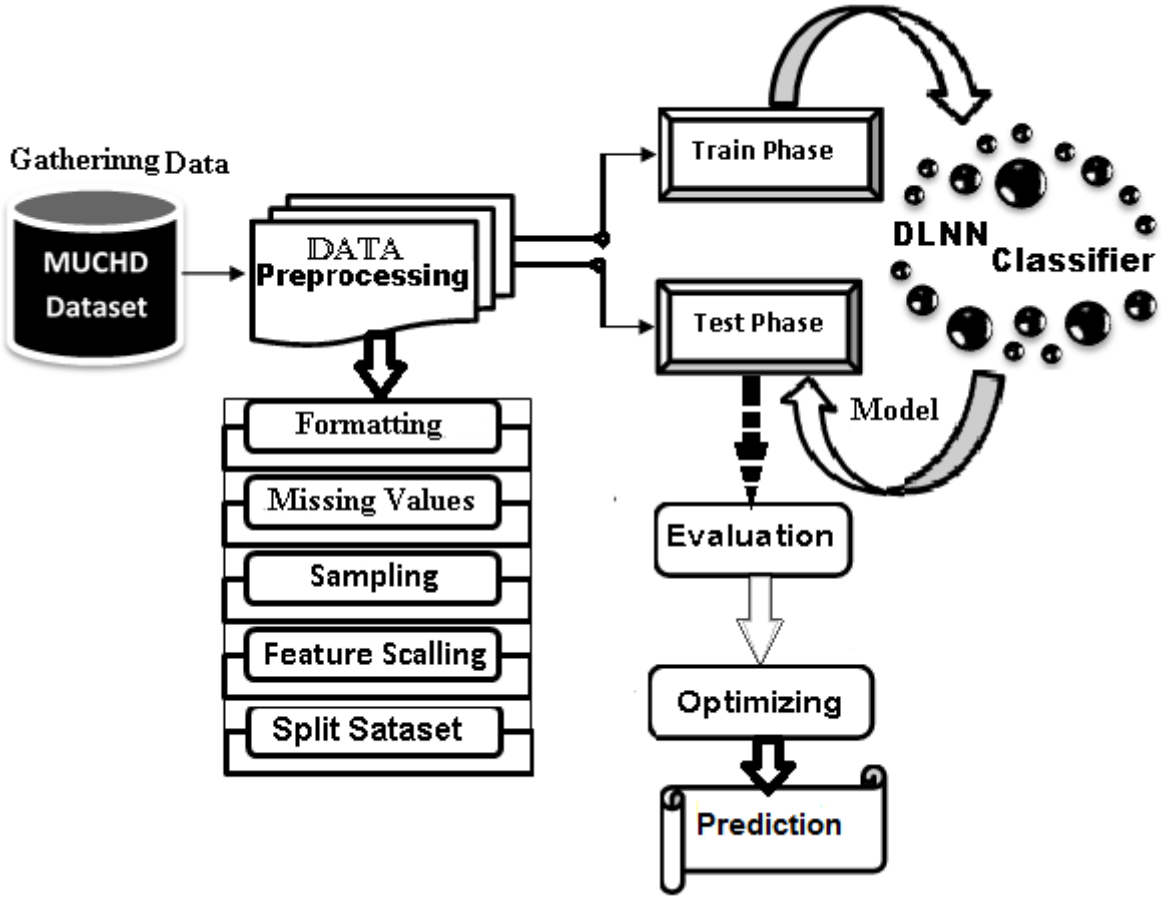



**Figure 1**

DLNN Architecture



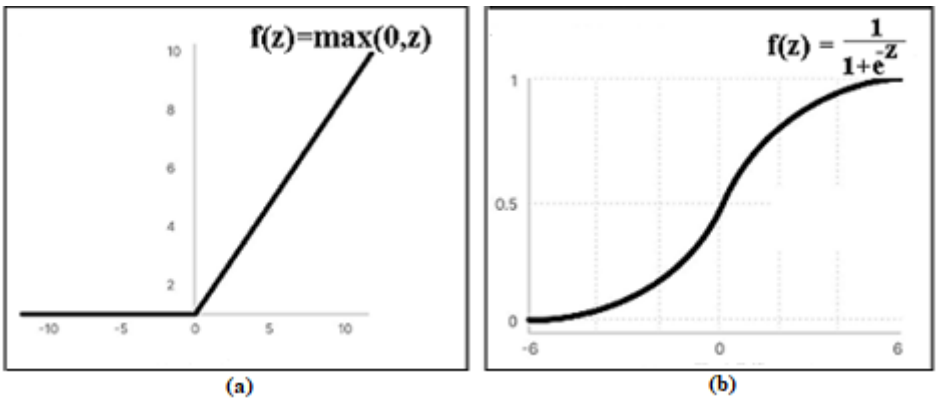

## Figure 2

Diabetes System Architecture

| Age | Sex | Duration | Cholesterc | Creatinine | Acetone | HbA1c | Insulin | PCPeptide | FCPeptide | PBGlucos | FBGlucos | RBGlucos | PH | HCO3 | Na | K |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 1 | 72 | 197 | 0.4 | 1 | 13.6 | 0.6 | 1.2 | | 350 | 211 | 446 | 7.365 | 15.1 | 157 | |
| 13 | 2 | 192 | 195 | 0.6 | | 7.5 | 0.7 | | 0.2 | 500 | 350 | | 7.439 | 23.8 | 133.3 | 4 |
| 3 | 2 | 2 | | 0.5 | 3 | 7.7 | 0.6 | 0.774 | 0.326 | 587 | 544 | 600 | 7.443 | 4.6 | 138.9 | 4.37 |
| 2 | 2 | 10 | 190 | | | 4.6 | 0.8 | 0.407 | | 525 | 400 | 500 | | 24.3 | 136.6 | 4.4 |
| 8 | 1 | 1 | 110 | 0.3 | 4 | 8.5 | 1.3 | 0.37 | 0.96 | 490 | 480 | 498 | 7.438 | 22.5 | 132.5 | |
| 8 | 2 | 3 | | 0.7 | 2 | 6.7 | 1.1 | 1.01 | 0.294 | 412 | 300 | 498 | 7.39 | 24.3 | 129.6 | 5.56 |
| 7 | 2 | 96 | 200 | 0.5 | | 6.8 | 0.7 | | 1.33 | 498 | 350 | 266 | 7.34 | 15.2 | 131.7 | 4.08 |
| 12 | 2 | 16 | 216 | 0.6 | 3 | 5.9 | 0.6 | 2.44 | 1.11 | 466 | 390 | 390 | 7.408 | 23.7 | 149.2 | 4.69 |
| 5 | 2 | 2 | 162 | 0.9 | 2 | 7.1 | 1 | 0.578 | 0.434 | 500 | 400 | 480 | 7.332 | 27.9 | 141.5 | 4.13 |
| 14 | 1 | | 171 | | | 8 | 0.8 | 0.376 | 0.9 | 326 | 233 | 500 | 7.316 | 24.3 | 131.1 | 3.47 |
| 5 | 1 | 2 | | 0.8 | 2 | 4.5 | | 0.238 | | 322 | 302 | 350 | 7.184 | 12.7 | 128 | 4.81 |
| 12 | 2 | 8 | 210 | 1.1 | 1 | 7 | 0.7 | 0.01 | | 433 | 300 | 450 | 7.297 | 19.2 | 138.7 | 5.02 |
| 5 | 2 | 120 | 200 | 0.4 | 1 | 5.6 | 1.2 | 1.82 | 0.565 | | 400 | | 7.427 | 17.7 | 147.9 | 2.24 |
| 3 | 1 | 4 | 145 | 0.5 | 4 | 8 | 0.7 | 2.59 | | 525 | 350 | 351 | 7.4 | 25.7 | 137.3 | 3.99 |
| 13 | 1 | 1.5 | | 0.4 | | 7.3 | 0.8 | 0.2 | 0.1 | 500 | | 525 | 7.325 | 27.7 | 139.9 | 4.27 |
| 12 | 1 | 1 | 164 | 0.4 | 1 | 7.5 | 0.7 | 0.21 | 0.03 | 454 | 362 | 290 | 7.349 | 26.1 | 139.9 | 4.29 |

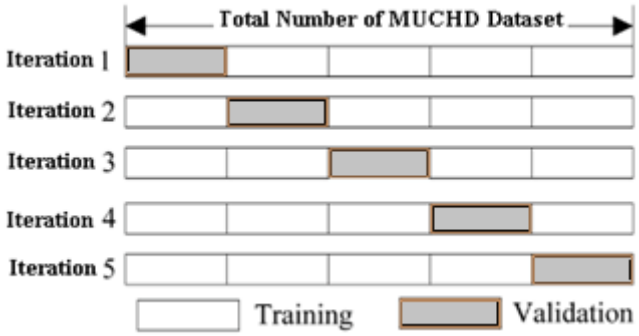
## Figure 3

Feature Attribute Missing Value



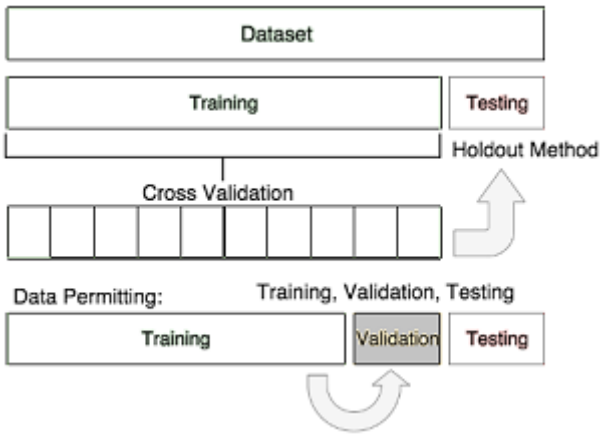

## Figure 4

(a) ReLU Function. (b) Sigmoid Function.




## Figure 5

K- fold Validation.




## Figure 6

Validation Process.