IDD-HPO: A Proposed Model for Improving Diabetic Detection using Hyperparameter Optimization and Cloud Mapping Storage

Eman H. Zaky[1]*, Mona M. Soliman[2], A. K. Elkholy[3], Neveen I. Ghali[4]
Department of Mathematical and Computer Science, Al-Azhar University, Cairo, Egypt[1, 3]
Department of Information Technology, Cairo University, Cairo, Egypt[2]
Department of Information Technology, Future University, Cairo, Egypt[4]

***Abstract*—Readmission to the hospital is an important and critical procedure for the quality of health care as it is very costly and helps in determining the quality level of the point of care provided by the hospital to the patient. This paper proposes a group model to predict readmission by choosing between Machine Learning and Deep Learning algorithms based on performance improvement. The algorithms used for Machine Learning are Logistic Regression, K-Nearest Neighbors, and Support Vector Machine, while the algorithms used for Deep Learning are a Convolutional Neural Network and Recurrent Neural Network. The reasons for the appearance of the efficiency of the model depend on the are preparation of correct parameters and the values that control the learning. This paper aims to enhance the performance of both machine learning and deep learning based readmission models using hyperparameter optimization in both Personal Computer environments and Mobile Cloud Computing systems. The proposed model is called improving detection diabetic using hyperparameter optimization , the proposed model aims to achieve the best rate of between prediction rate accuracy for hospital readmission at the same time minimizing resources such as time delay and energy consumption. Results achieved by proposed model for Logistic Regression, K-Nearest Neighbors, and Support Vector Machine are (accuracy=0.671, 0.883, 0.901, time delay=5, 7, 20, and energy consumed=25, 32, 48) respectively, for Recurrent Neural Network and Convolutional Neural Network are (accuracy=0.854, 0.963, time delay=25, 660 energy consumed=89, 895) respectively. However, this proposed model takes a lot of time and energy consumed especially in Convolutional Neural Network. So, the experiments were conducted again, but in the cloud environment, based on the existence of two types of storage to preserve the accuracy but decreasing time and energy, the proposed model in cloud environment achieve for Logistic Regression, K-Nearest Neighbors, and Support Vector Machine (accuracy=0.671, 0.883, 0.901, time delay=2, 3, 8, and energy consumed=8, 9, 11) respectively, for Recurrent Neural Network, Convolutional Neural Network (accuracy=0.854, 0.963, time delay=15, 220, and energy consumed=20, 301) respectively.**

Keywords*—Machine learning; deep learning; diabetes; hospital readmission; hyper parameter optimization; cloud computing; mobile cloud computing

I. Introduction

With the change of lifestyle and the massive expansion in many countries, diabetes, which is considered a chronic and non-communicable disease, has become one of the most deadly diseases. Despite this, many deaths can be prevented through data analysis [18]. Therefore, in most developing countries, diabetes is a primary health care concern, the healthcare sector collects and processes medical data for diabetic patients in huge quantities of diverse sizes and structures [19].

The meaning of readmission to a hospital is the time it takes for the patient to return to the hospital again. Hospital quality is measured and health care costs are reduced by measuring readmission hospitals are financially penalized for exceeding the permitted rate of 30-day readmissions [21]. Machine Learning (ML) algorithms can be used to create objective models which then can be used to measure risk. These models are more complex but may be able to create more accurate risk predictions that should lead to improved diabetic patient outcomes [20]. Deep Learning (DL) algorithms have recently attracted a lot of interest in educational circles and commercialism because of their effective impact in various fields of research, such as speech recognition, natural language processing, and brain computer interface [16].

When creating ML and DL algorithms, there are many possibilities to define the architecture of the learning model. Often, the optimal model architecture for a given model is not known, and thus a range of possibilities must to be able to explore. In the way of ML and DL, the machine is asking to perform this exploration and automatically determine the optimal model structure. The parameters that define the structure of the model are referred to as hyperparameters [13], hence the process of searching for the ideal structure of the model is referred to as hyperparameter optimization [27].

Despite all this, ML and DL face an important challenge, as the performance of the algorithm depends heavily on its choice of parameters. DL requires hyperparameter optimization more than ML because (1) DL has more hyperparameters to be optimized, (2) DL has a higher dependency on the configuration of hyperparameters.

The accuracy of using deep learning changes drastically from 32.2% to 92.6% due to the variable selection of hyperparameters as reported [17]. Therefore, an effective hyperparameter optimization method is necessary for ML and DL.

*Corresponding Author

Other models used different ML and DL algorithms with different preprocessing methods but could not predict readmission for diabetic patients with high accuracy, nor did it take into account the point of saving resources such as reducing the time and energy consumed in the prediction process.

The goal of the paper is to 1) develop an accurate and generalized machine learning and deep learning models that is applicable to predicting 30-day readmission for diabetic patients by using hyperparameter optimization to control learning of machine; 2) to demonstrate the efficiency of (IDD-HPO) in mobile cloud computing environment with saving resources to minimize the total cost; 3) compare (IDD-HPO) results with the results of state-of-the-art models.

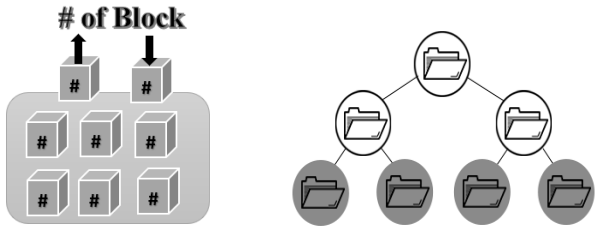
II. Background

Cloud Computing (CC) has recently emerged as a new framework to facilitate and implement online services [22], storing, analyzing, and displaying data requires significant modifications to the current cloud model, which in turn requires financial constraints and computational cost [23,24]. Many CC platforms provide these web services for ML and DL. The most popular of these are Amazon Web Services, Microsoft Azure, Google Cloud, and IBM Cloud [25]. Cloud storage is the online storage of data on the cloud. The data on the mobile is sent and stored in the cloud, the mobile device can access this data any time anywhere by sending data requests.

The two most common technologies of the system cloud storage are block-level storage and file-level storage. These two storage levels are described as follows [26]:

Block-level storage: Data is stored in blocks on a device with fixed sizes for each block (e.g., 512 Bytes). Data is stored according to the data format, type of ownership for each block, data stored as blocks in hard drives, which are installed in remote storage, a request is sent from the filing system to the storage this request is responsible for writing data to certain blocks and then retrieve it as shown in Fig. 1(a). Block storage is built to simplify larger workloads and improve Input/Output Operations per Second (IOPS), they are apt to be more expensive than file storage systems. However, this seriously depends on the chosen seller, conditions, features, cost of the storage operating system (OS), and some other variables.

File-level storage: Simply having an efficient, easily reached, and existing location to store files and data folders continues to be the most important requirement of any organization. For file-level storage, the only thing that is required is having a location to unload the data as shown in Fig. 1(b). File storage is stores data in a hierarchical architecture so the data and its metadata are stored in the form of files and folders. File storage systems are usually less costly than block storage. However, this seriously depends on the chosen seller, conditions, features, cost of the storage operating system (OS), and some other variables.



(a) Block-level Storage. (b) File-level Storage.

Fig. 1. Types of System Cloud Storage.

This paper proposes a model with the objective of predicting diabetic readmission rate in hospitals, this model is named “Improving Detection Diabetic Using Hyperparameter Optimization” ((IDD-HPO)) by using ML algorithms (e.g., Logistic Regression (LR), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM)) and DL algorithms (e.g., Convolutional Neural Network (CNN), Recurrent Neural Network (RNN)) for predicting hospital readmission among diabetics using hyperparameter optimization method in two scenarios, the first in personal computer (anaconda 3) and the second in the cloud environment.

This paper aims to: Improving the prediction of readmission to hospitals using hyperparameter optimization for both ML and DL algorithms. These prediction models are tested in two environments: personal computer and mobile cloud computing MCC system. The aim of these models is to achieve the best performance with high accuracy and decreasing total cost (e.g. time delay and energy consumed) based on mapping storage in the cloud. As a case study, the proposed model ((IDD-HPO)) was applied in a personal computer and cloud computing system.

The rest of this paper is organized as follows; the next section presents some related works. Then the proposed method will present in Section III. Subsection A in Section III presents the dataset description, subsection B presents the details of Machine Learning and Deep learning models tuning Using Hyperparameter Optimization, subsection C presents the details of the Cost Model for sending and receiving data to/from a server, and subsection D presents the details of the proposed Improving Detection of a Diabetic using Hyperparameter Optimization (IDD-HPO) model. Section IV presents the experiments for (IDD-HPO) on Personal computer system, subsection B presents the experiments for (IDD-HPO) on MCC system, subsection C presents the Comparative analysis Against State-of-the-Art Models with Results. Finally, conclusion and future works.

III. Related Work

In the field of health care, readmission to the hospital is considered a high priority because it represents whether the hospital is good or not, as it also aims to reduce costs. Little research has focused on readmission for diabetes, although diabetes on hospitalized patients has a large, growing, and costly burden.

In [2], this study was focused on the application of data mining techniques to predict the early hospital readmission by using ML algorithms, the most efficient algorithm was Random Forest with 0.898% of accuracy.

In this study [3], the prediction is done using Support Vector Machine (SVM), Random Forest (RF), Multilayer Perceptron (MLP), and Deep Neural Network if the discharged patient will be back in 30 days or not and the best presented accuracy value is 0.840%.

In [4], the model used in this study consists of 5 models tested and selected from 15 models. These blends are variables from logistic regression, decision trees, neural networks, and a naive Bayesian enhancer [5]. The performance of the model was on an unbalanced data set, and these models were selected after several tests and analysis of their accuracy, the accuracy value is 0.635%.

In [6], in the problem of readmission of diabetic patients to hospitals, the deep convolutional neural network (CNN) was used as an effective prediction method. This model is based on sample size scaling and data engineering processes. Using normalization is key to improving deep learning performance. This model achieves 92% better performance than other ML models.

In [7], Deep learning models (CNN, RNN) and machine learning algorithms (LR, KNN, SVM) were used to solve the problem of readmission of diabetic patients to hospital, this model is based on the use of data without normalization and the results were compared in the case of normalization and without normalization for (CNN, RNN, LR, KNN, SVM). The optimal performance in training and testing was in CNN in the case of using the data without normalization, where the accuracy ratio was achieved 0.924%.

In [8], the proposed Multilayer Perceptron model based on preprocessing process included comprehensive data cleaning, data reduction, and transformation aiming at better optimizing and selecting prominent features for 30-day unplanned readmission among diabetes patients. Random Forest algorithm is used for feature selection and SMOTE [9] algorithm for data balancing. a model consisting of two hidden layers with dropout [10] to achieve high overall accuracy and ROC. The proposed model with feature engineering is improving the performance of the others Machine learning algorithms, the accuracy value is 0.95%.

In [11], the researchers analyzed the readmission of diabetic patients using unsupervised methods and proposes an approach of generating embeddings for categorical features concatenated with normalized continuous features were fed into a neural network, the accuracy achieved is 0.952%.

In conclusion, most recent models relied on several methods for presenting data. These methods include: using Ensemble, normalization, non-normalization, Ensemble by age groups, and using data mining, all these methods do not achieve a high rate of accuracy. In this paper, the proposed model aims to use hyperparameter optimization methods to control the learning and evaluate the performance for the optimal values with the highest possible accuracy, but hyperparameters take a lot of time delay and energy consumed for the learning process. Hence the idea of using a hyperparameter in the cloud environment to reduce the time delay and energy consumed is proposed.

IV. The Proposed Method

This work proposes a model for improving the readmission rate. This model for improving detection of a diabetic using hyperparameter optimization ((IDD-HPO)) is implemented and compared in two environments: personal computers and cloud environment, then calculate accuracy, time delay, and energy consumed for each environment.

Fig. 2 summarizes the overall architecture of the proposed (IDD-HPO) model. In the following subsection, the model will be illustrate in more detail for each stage to build.

An intelligent model ((IDD-HPO)) based on hyperparameter optimization technique is proposed to enhance a choice between two classes (0, 1) where 0 is not readmitted and 1 is readmitted). Models used to apply ((IDD-HPO)) are CNN and RNN as DL algorithms and KNN, LR, and SVM as ML classifiers for the prediction of readmission.

In MCC, diabetes data set sent from mobile to cloud and retrieve from it to mobile based on the existing two levels of storage, file and block level.

The problem at the first applying ((IDD-HPO)) for data set in a personal computer system (anaconda3) and calculate the accuracy and total cost (time delay and energy consumed) for training and testing. Second applying ((IDD-HPO)) for data set in the cloud according to how to map mobile data item taking into account the limited resources of a mobile device by selecting the most suitable storage level for each data item to decreasing the time of training band testing for hyperparameter tuning and calculate accuracy and total cost (time delay and energy consumed).

In the rest of this section, the model is introduced then the (IDD-HPO) problem will be formulated.

A. Dataset Description

This proposed study is performed on a dataset represent 10 years (1999-2008) of clinical care at 130 hospitals across the United States and is provided by the Center for Clinical and Translational Research at Virginia Commonwealth University [1]. This data was used to predict the probability of readmission within the next 30 days for a patient with diabetes. The extracted information from the database for interviews must meet the following global criteria [14]:

1) All data taken during the meeting are from hospital cases.

2) Only diabetics are the ones to take data from at the meeting.

3) Range of stay patient in the hospital about 1-14 days.

4) Laboratory tests were carried out during the meeting.

5) Medicines were provided during the meeting.

101,766 encounters were identified to fulfill all of the above five inclusion criteria and were used in further analysis.

The data set was generated through three steps:

First, important features were extracted from the database. It was found that there were 55 features to be used in the study.

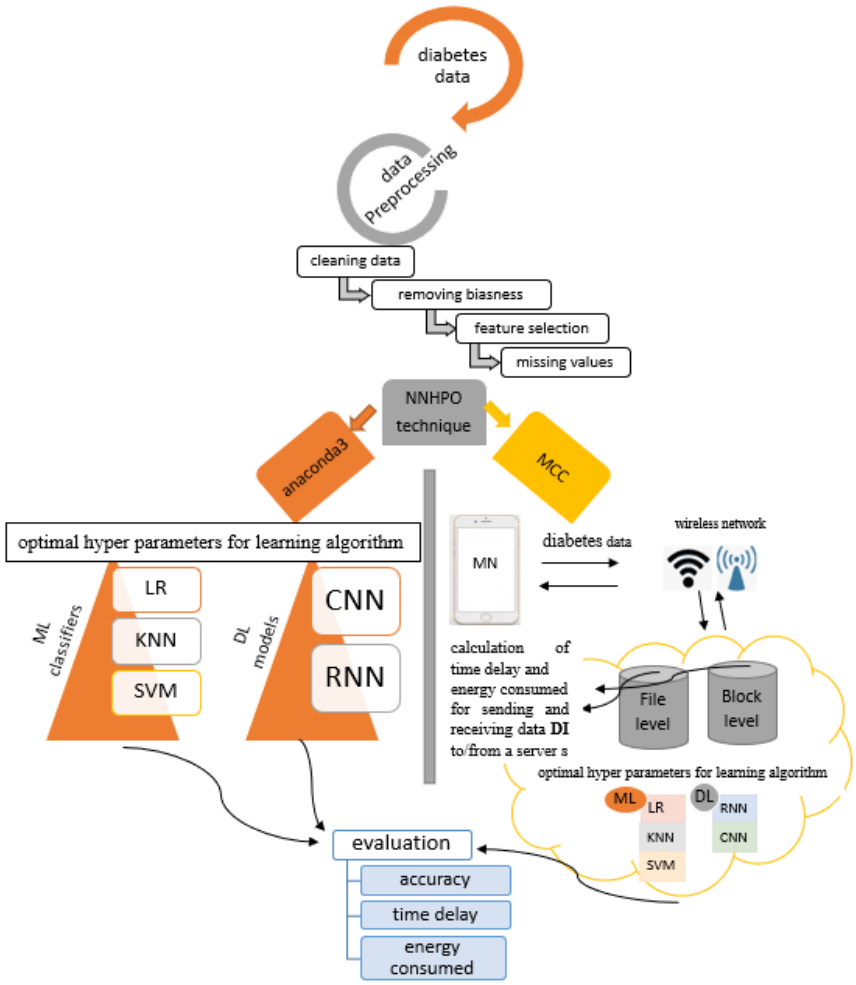


Fig. 2. The Proposed (IDD-HPO) Model in Personal Computer and MCC System.

Second, the data was preprocessed to extract useful information for research [11].

Third, the data set was upload to the cloud and store in two levels of storage file and block.

Data preprocessing will be implemented through the following steps:

- Cleaning Of Data: Firstly, delete records of patients who appear more than once to ensure that each patient has a unique identity. Then remove the features "patient_id,encounter_id" which tells about the Patient Number and Unique Identifier of a patient respectively.
- Removing Biasness: Also remove patient data dead or discharged to a hospice. Some rows are also repeat because the number of patients admitted within 30 days is very low.
- Feature Selection: There are a total of 55 features in this dataset, 23 of them medicine related features. After visualizing each other's dependency with the readmission feature found that the drugs the least role it plays in readmission, so 22 out of 23 medical features have been removed.
- Missing Values: Features that have a large percentage of missing values such as weight contain 97% so cannot be used in the analysis. So, the features with more than 30% missing values are removed. There is a feature, "medical specialty" that defines the specialty of attending physician which has some missing data so fill "Missing" in the missing place as this is the important feature for analysis. Then convert all the string categorical data into Integer categorical data to do analysis.
- Non-normalization technique: Data set will use without using any normalization. The advantage of the non-normalization technique is the facility to fix all features. It allows the classifier model to get the benefit of all features. Normalization is a packed down data between either -1 and 1 or 0 and 1. So data are need to use without any normalization to get correct output [7].

B. Machine Learning and Deep Learning Models Tuning using Hyperparameter Optimization

In ML and DL, the problem of selecting a set of optimal hyperparameters for a learning algorithm is hyperparameter optimization or tuning [10]. A hyperparameter is a parameter whose value is used to control the learning

process and evaluates the problem to finding a set of optimal hyperparameters y^* in the domain Y that return the best performance as evaluated on a validation set y:

$$y^* = \arg\min_{y \epsilon Y} f(y) \quad (1)$$

where the optimal solution is defined as the minimum of objective function *f*(*y*) that commonly corresponds to a loss function or an error rate.

The learning algorithm uses hyperparameters when it learns but it is not part of the resulting model. Parameters of the model have been trained which is effectively considered as a model at the end of the learning process, hyperparameters that were used during training are not part of this model. For example, the values of the hyperparameters that were used to train a model can not know from the model itself, the parameters of the model that were learned only know [12].

Tuning ML models is a kind of optimization problem. A set of hyperparameters exist and aim to find the correct combination of their values which can help us find the maximum (e.g. precision) of a function. For the proposed model, sets of ML and DL algorithms are used with different hyperparameters. The following is the list of used algorithms with their associated parameters.

- Logistic Regression (LR) tuning

Logistic regression does not contain any important hyperparameters to adjust. But sometimes, there are differences in performance with different solvers (*solver*).

- solver in ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga'].
- Regularization (*penalty*) can sometimes be helpful.
- penalty in ['none', 'l1', 'l2', 'elasticnet'].
- The C parameter controls the penality strength, which can also be effective.
- C in [100, 10, 1.0, 0.1, 0.01].

- K-Nearest Neighbors (KNN) tuning.

The important hyperparameter for KNN is the number of neighbors (*n_neighbors*).

- Test values between at least 1 and 21.
- weights in ['uniform', 'distance']

- Support Vector Machine (SVM) tuning

There are a large number of hyperparameters in SVM algorithm to tune. For example, the parameter kernel will control how the input variables will be projected.

- kernels in ['linear', 'poly', 'rbf', 'sigmoid']

Another parameter that can take on a range of values is the penalty (C).

- C in [100, 10, 1.0, 0.1, 0.001]

- CNN and RNN Tuning

Grid Search and Random Search are applied for DL using Kera Classifier, it is possible to apply in the same way when using scikit-learn ML models. Some of CNN and RNN parameters will be optimized such as: how many epochs to use in each layer, the number of batch size, and which activation function and optimizer to use as shown in Fig. 3.

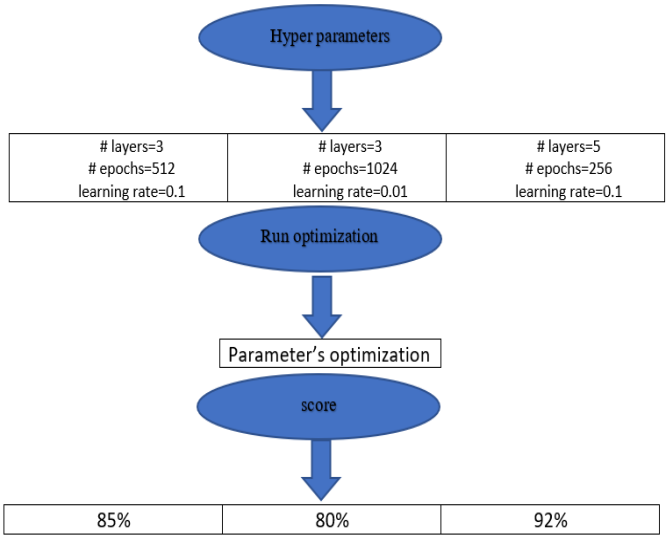


Fig. 3. CNN and RNN Hyperparameters.

C. Cost Model for Sending and Receiving Data DI to/from a Server s

In this subsection, the cost model is described for using the file and the block servers.

- Cost by using block cloud server

Block storage is built to simplify larger workloads, each data item needed to classify before storing it in the appropriate block. So, the cost for sending includes sending, classifying, and searching costs, and the cost for receiving includes retrieving and searching costs, they are apt to be more expensive than file storage systems. However, this cost depends on the chosen seller, conditions, features, cost of the storage operating system (OS), and some other variables.

- Cost for time delay in block cloud server

The cost of the time delay for sending DI to the block cloud server bcs is calculated as follows

$$CTD(S)_{bcs}(DI) = \sum_{d_i \in DI} ctd_s(d_i, bcs) \quad (2)$$

The cost of the time delay for receiving DI to the block cloud server bcs is calculated as follows:

$$\mathrm{CTD(R)_{bcs}(DI)} = \sum_{\mathrm{d_i} \in \mathrm{DI}} \mathrm{ctd_r(d_i, bcs)} \quad (3)$$

By using Equations 2, and 3 the total cost for time delay by using bcs can be defined as follows:

$$\mathrm{TCTD_{bcs}(DI)} = \mathrm{CTD(S)_{bcs}\ (DI) + CTD(R)_{bcs}(DI)} \quad (4)$$

- Cost for energy consumed in block cloud server

The cost of the energy consumed for sending DI to the block cloud server bcs is calculated as follows:

$$\mathrm{CEC(S)_{bcs}(DI)} = \sum_{d_i \in DI} \mathrm{cec_s}(d_i, \mathrm{bcs}) \quad (5)$$

The cost of the energy consumed for receiving DI to the block cloud server bcs is calculated as follows:

$$\mathrm{CEC(R)_{bcs}(DI)} = \sum_{d_i \in DI} \mathrm{cec_r}(d_i, \mathrm{bcs}) \quad (6)$$

By using Equations 5, and 6 the total cost for energy consumed by using bcs can be defined as follows:

$$\mathrm{TCEC_{bcs}(DI)} = \mathrm{CEC(S)_{bcs}(DI)} + \mathrm{CEC(R)_{bcs}(DI)} \quad (7)$$

By using Equations 4, and 7 the total costs by using bcs is defined as follows.

$$\mathrm{TC_{bcs}(DI)} = \mathrm{TCTD_{bcs}(DI)} + \mathrm{TCEC_{bcs}(DI)} \quad (8)$$

- Cost by using file cloud server.

File storage systems are usually less costly than block storage because the cost for sending includes sending without classification and searching costs and the cost for receiving includes retrieving and searching costs. However, this cost depends on the chosen seller, conditions, features, cost of the storage operating system (OS), and some other variables.

- Cost for time delay in file cloud server.

The cost of the time delay for sending DI to the file cloud server fcs is calculated as follows.

$$\mathrm{CTD(S)_{fcs}(DI)} = \sum_{d_i \in DI} \mathrm{ctd_s}(d_i, \mathrm{fcs}) \quad (9)$$

The cost of the time delay for receiving DI to the file cloud server fcs is calculated as follows.

$$\mathrm{CTD(R)_{fcs}(DI)} = \sum_{d_i \in DI} \mathrm{ctd_r}(d_i, \mathrm{fcs}) \quad (10)$$

By using Equations 9, and 10 the total cost for time delay by using fcs can be defined as follows.

$$\mathrm{TCTD_{fcs}(DI)} = \mathrm{CTD(S)_{fcs}\ (DI)} + \mathrm{CTD(R)_{fcs}(DI)} \quad (11)$$

- Cost for energy consumed in file cloud server.

The cost of the energy consumed for sending DI to the file cloud server fcs is calculated as follows:

$$\mathrm{CEC(S)_{fcs}(DI)} = \sum_{d_i \in DI} \mathrm{cec_s}(d_i, \mathrm{fcs}) \quad (12)$$

The cost of the energy consumed for receiving DI to the file cloud server fcs is calculated as follows:

$$\mathrm{CEC(R)_{fcs}(DI)} = \sum_{d_i \in DI} \mathrm{cec_r}(d_i, \mathrm{fcs}) \quad (13)$$

By using Equations 12, and 13 the total cost for energy consumed by using fcs can be defined as follows:

$$\mathrm{TCEC_{fcs}(DI)} = \mathrm{CEC(S)_{fcs}\ (DI)} + \mathrm{CEC(R)_{fcs}(DI} \quad (14)$$

By using Equations 11, and 14 the total costs by using fcs are defined as follows;

$$\mathrm{TC_{fcs}(DI)} = \mathrm{TCTD_{fcs}(DI)} + \mathrm{TCEC_{fcs}(DI)} \quad (15)$$

D. (IDD-HPO) Proposed Model

The (IDD-HPO) model is used at the beginning with the ML classifiers (e.g., LR, KNN, and SVM), and DL algorithms (e.g., CNN, RNN), then the performance of (IDD-HPO) is measured by Accuracy, time delay, and energy consumed. Comparing it with the state of the art and proving the effectiveness of using (IDD-HPO) for diabetes data, (IDD-HPO) model is performed in a personal computer system (anaconda3) environment and both the time and energy consumed in the training and testing process are measured.

To reduce the time and energy consumption of (IDD-HPO) model in both ML and DL algorithms, the experiment environment is changed by using (MCC).

In a personal computer system (anaconda3) environment, the main goal of (IDD-HPO) model is using hyperparameters to control the learning process and evaluates the problem to finding a set of optimal hyperparameters that return the best performance as evaluated on a validation set.

In MCC, the main goals of (IDD-HPO) model are (1) minimizing the cost of time delay and energy consumed spent for sending and retrieving data. So, based on the previously described cost model in subsection C, the goal is finding the best cloud server (e.g. file cloud server (DI_{fcs}) or block cloud server (DI_{bcs})) to store data in the cloud. Such that no data item can be mapped to bcs and fcs at the same time. Then used a hyperparameter on the best cloud server controls the learning process and evaluates the problem to finding a set of optimal hyperparameters that return the best performance with the smallest time delay and energy consumed compared to time and energy on a personal computer.

To conduct training and testing in the cloud, depending on that the data is stored on the mobile, the data is sent from the mobile to the cloud and stored in the cloud depending on two types of storage, file, and block. The time and energy consumed in the process of storing data in each of the two types of storage (file and block) are calculated to determine the best type of storage according to the type of data used in the cloud. The time and energy consumed to conduct the training and testing process using (IDD-HPO) model are compared in both personal computer systems (anaconda3) and cloud computing environments. Here, MCC system model involves (1) mobile node, MN, (i.e., mobile device with a user) which has a set of data for diabetes DI = $\{d_i, 1 \leq i \leq n\}$, and each data item di represents different types of data (e.g., text, numbers, symbols, etc.). (2) a file cloud server, fcs, which stores all received data elements di from MN as files. (3) a block cloud server, bcs, which stores all received data elements d_i from MN as blocks. The energy consumed for sending and receiving a data item d_i to/from a server s are denoted as $cec_s(d_i; s)$ and $cec_r(d_i; s)$, respectively. Wheres can be a file server fcs or a block server bcs. Also, the time delay for sending and receiving a data item d_i to/from a server s are denoted as $ctd_s(d_i; s)$ and $ctd_r(d_i; s)$, respectively.

To improve the performance of the hospital readmission problem, the basic idea of (IDD-HPO) model is based on the following three issues: (a) using an (IDD-HPO) model for each model (ML, DL) in a personal computer and calculate

the accuracy, time delay, and energy consumed. (b) estimating the total cost data sets DI on the file and block cloud servers by calculating their costs which were determined by using Equations (4,7,10,11). (c) comparing the calculated total costs and selecting the best appropriate mapping server based on the needs of a mobile user (e.g., minimum mapping costs).

The steps of these two cases are described as follows in Fig. 4 and 5.

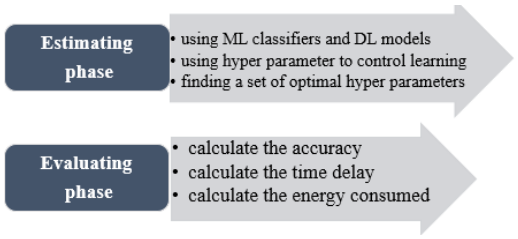


Fig. 4. Proposed (IDD-HPO) in Case 1 (a Personal Computer).

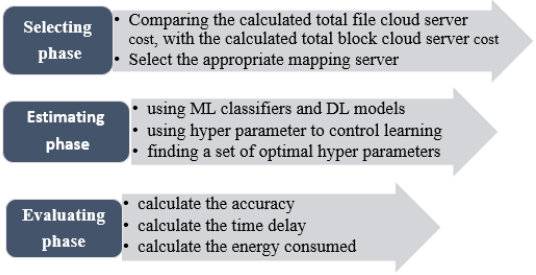


Fig. 5. Proposed (IDD-HPO) in Case 2 (MCC System).

V. Experimental Result

This section evaluates the performance of the proposed method for predicting hospital readmission using (IDD-HPO) model. through comparing it in two cases. The first one is when applying learning and testing for ML algorithms and DL algorithms in personal computers (anaconda3) with the state of the art, and calculate the time delay and energy consumed for (IDD-HPO) model. The second case is when applying learning and testing for ML algorithms and DL algorithms in the MCC system.

For the first experiment: (IDD-HPO) model with ML classifiers and DL algorithms, as the intelligent model, is built in Spyder Python 3.7 environment with processor intel(R) Core (TM), i5-2500 CPU @3.30 GHz.

For the second experiment: at the first the OMNet ++ [15] simulator was used to evaluate the best level of storage (e.g., fcs or bcs) for decreasing the total cost. Also, each experiment is repeated 5 times and the average was taken.

Then The (IDD-HPO) model is developed using Spyder python computing environment. Prediction models using (IDD-HPO) in the research were developed using the deep learning toolkit provided by the Anaconda software. The network configuration is initially determined based on the model that is built. The model is implemented on a GPU-enabled system with an Intel Core i7 processor with a capacity of 16 GB RAM.

CNN and RNN models are applied with one input layer, three hidden layers with uniform initialization, and one output layer. Softmax activation function was chosen for the output layer, while PRelu activation function was chosen for input layers. Added Dropout with rate=0.1 after hidden layers to limit overfitting and hence DL algorithms.

The selected optimization algorithms were [rmsprop, adam, sgd].

(IDD-HPO) model is used to find optimal hyperparameters (Table I). The classifiers with the optimal hyperparameters were tested on the holdout test set. This approach ensures that the training, validation, and evaluation data are completely separated.

A. Experiments for (IDD-HPO) on Personal Computer System (Anaconda3)

In this experiment, both ML classifiers against DL algorithms are used with (IDD-HPO) model. A hyperparameter is used to control the learning process and evaluates the problem to finding a set of optimal hyperparameters. For LR a hyperparameter (Regularization (*penalty*) is used to choose from the range [none, 11, 12, elasticnet], KNN a hyperparameter (Number of neighbors) is used to choose from the range [1-21], SVM a hyperparameter (kernels) is used to choose from the range [linear, poly, rbf, sigmoid], RNN hyperparameters (learning rate *lr* and Optimizer**)** are used to choose from the range [0.004, 0.008, 0.0012] and [rmsprop, adam, sgd] respectively, and CNN hyperparameters (epochs and batch_size) are used to choose from the range [50, 100, 150, 200] and [16, 32, 64] respectively. As shown in Table I.

TABLE I. (IDD-HPO) Model for ML and DL

Classifier	hyperparameter	Range	optimal
LR	Regularization (*penalty*)	[none, 11, 12, elasticnet]	none
KNN	Number of neighbors	[1–21]	3
SVM	kernels	[linear, poly, rbf, sigmoid]	linear
RNN	learning rate *lr*,	[0.005, 0.01, 0.015]	0.005
	the regularization coefficient λ	[0.004, 0.008, 0.0012]	0.004
CNN	Optimizer	[rmsprop, adam, sgd]	Adam
	epochs	[50, 100, 150, 200]	150
	batch_size	16, 32, 64	64

TABLE II. PERFORMANCE MATRIX FOR (IDD-HPO) IN A PERSONAL COMPUTER

Classifier	Accuracy for (IDD-HPO)	Time delay for (IDD-HPO)	Energy consumed for (IDD-HPO)
LR	0.671	5	25
KNN	0.883	7	32
SVM	0.901	20	48
RNN	0.854	25	89
CNN	0.963	660	895

Table II shows a comparison performance for ML and DL algorithms when used (IDD-HPO) model with test size 10. As shown in Table II the performance of DL algorithms is more accurate in predicting the use of ML. ML algorithms always need structured data, while DL networks rely on ANN (Artificial Neural Networks) layers. Therefore, the performance of DL was better as the data is not structured but it is multi-dimensional data. Also, time delay and energy consumed are calculated for training and testing both ML and DL algorithms.

Fig. 6, Fig. 7, and Fig. 8 show the accuracy, time delay, and energy consumed respectively for training and testing for Dl with (IDD-HPO) is very high compared to ML, where accuracy, time delay, and energy consumed to training and testing DL is (0.963%, 660 minutes, and 895 joules), respectively.

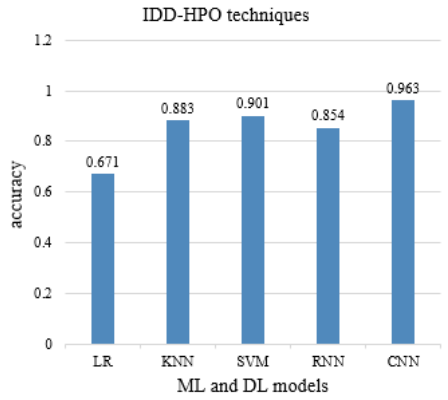


Fig. 6. Accuracy for (IDD-HPO) in Personal Computer.

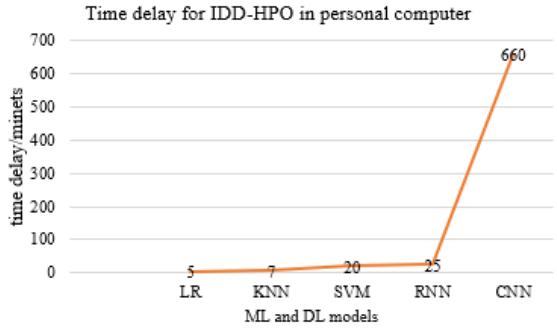


Fig. 7. Time Delay for (IDD-HPO) in Personal Computer.

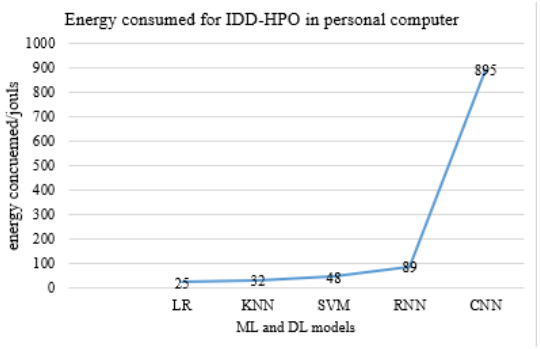


Fig. 8. Energy Consumed for (IDD-HPO) in Personal Computer.

B. Experiments for (IDD-HPO) on MCC System

The experiment is applying in a cloud computing environment so, at the first calculate the time delay and energy consumed for (IDD-HPO) model if data is store at the file and block cloud server.

Table III, shows the accuracy, time delay, and energy consumed for ML algorithms and the DL algorithms when used (IDD-HPO) model in fcs and bcs with test size 10. Also, time delay and energy consumed are calculated for training and testing both ML and DL algorithms if data is store in the cloud as fcs and bcs.

Fig. 9, and Fig. 10 show the comparing of the time delay and energy consumed respectively of using (IDD-HPO) model which is calculated by Equations 4, 7, 11, and 14 when training and testing are played in personal computer and MCC system using fcs and bcs. As shown in Fig. 10, and Fig. 11 The time delay and energy consumed of using (IDD-HPO) are decreasing if the data set is stored in the cloud as fcs against stored data in the cloud as bcs, and use (IDD-HPO) in a personal computer.

Fig. 11 shows the comparing total cost for (IDD-HPO) in the personal computer against (IDD-HPO) in cloud environment based on fcs and bcs which can be calculated by Equations 8, 15 the total cost for (IDD-HPO) in fcs is less than the total cost for (IDD-HPO) in personal computer and total cost for (IDD-HPO) in bcs which satisfies the required conditions for the proposed method in case 2.

TABLE III. PERFORMANCE MATRIX FOR (IDD-HPO) BY USING FCS AND BCS

Classifier	Accuracy for (IDD-HPO)	Time delay for (IDD-HPO) (fcs)	Time delay for (IDD-HPO) (bcs)	Energy consumed for (IDD-HPO) (fcs)	Energy consumed for (IDD-HPO) (bcs)
LR	0.671	2	3	8	6
KNN	0.883	3	5	9	7
SVM	0.901	8	10	11	8
RNN	0.854	15	18	20	16
CNN	0.963	220	290	301	252

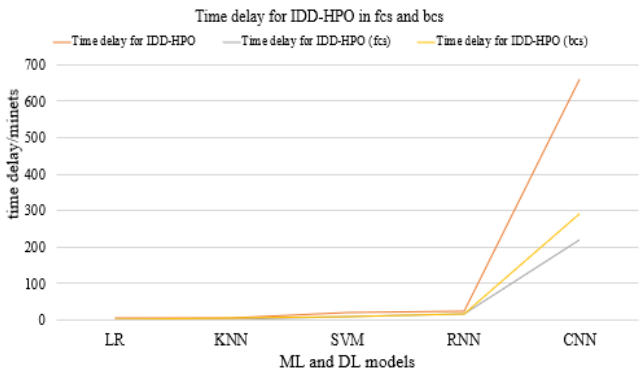


Fig. 9. Time Delay for (IDD-HPO) in Personal Computer and Cloud Server.

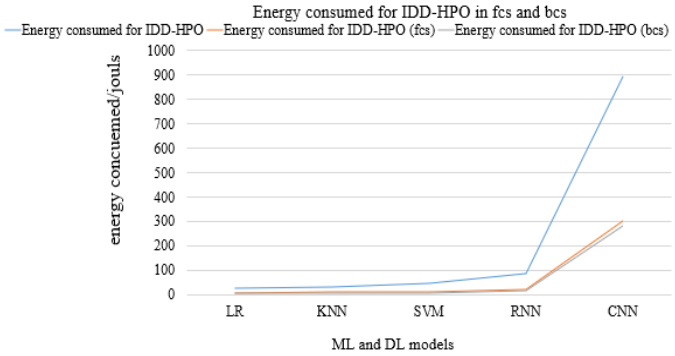


Fig. 10. Energy Consumed for (IDD-HPO) in Personal Computer and cloud Server.

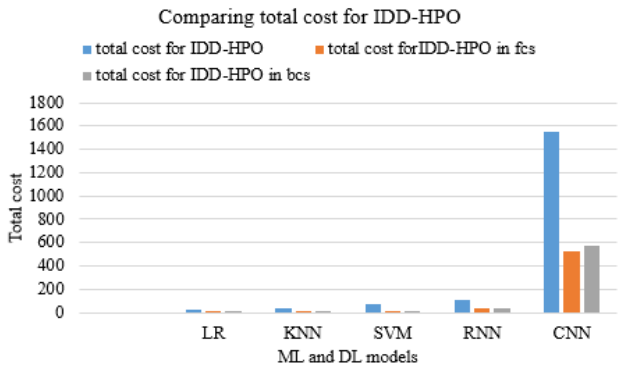


Fig. 11. Comparing Total Cost for (IDD-HPO) with a Personal Computer, FCS, and BCS.

C. Comparative Analysis against State-of-the-Art Models with our Results

In this section, a comparison between the proposed model with the highest accuracy measures ((IDD-HPO)) and other state-of-the-art models illustrated before in the related work section is discussed. Table IV provides a complete analysis of such a comparison. It compares (IDD-HPO) model with other models reported in [7], [11], [2], [8], [3], [4], and [6]. Each one of these models used different ML and DL algorithms with different preprocessing methods (e.g. [7] used ML and DL algorithms, [11] used Categorical Embeddings and Neural Networks, [2] used Data Mining technique with Random Forest, [8] used Multilayer Perceptron, [3] used Multilayer Perceptron (MLP) and Deep Neural Network, [4] used ML with Ensemble Technique, and [6] used CNN with Normalization technique).

TABLE IV. COMPARATIVE ANALYSIS AGAINST STATE-OF-THE-ART MODELS WITH OUR RESULTS

Year	**LR**	**KNN**	**SVM**	**Simple Neural Network**	**RNN**	**CNN**	**Computing environment**	**Model basis**	**Ref.**
	Readmission Prediction Accuracy								
Proposed model	**0.671%**	**0.883%**	**0.901%**	-	**0.854%**	**0.963%**	A personal computer system (anaconda3)	(IDD-HPO) model	-
(2021)	0.642%	0.872%	0.886%	0.873%	0.837%	0.924%	A personal computer system (anaconda3)	ML, RNN, and CNN with non-normalization Technique	[7]
(2021)	---------					0.952%	Personal computer system	Categorical Embeddings and Neural Networks	[11]
(2021)	0.898%						Personal computer system	data mining techniques with random forest	[2]
(2019)	---------					0.95%	Personal computer system	Multilayer Perceptron	[8]
(2019)	0.840%						Personal computer system	Multilayer Perceptron (MLP) and Deep Neural Network	[3]
(2019)	0.635%	-	0.2946%	0.7999%	-	-	Personal computer system	ML with Ensemble Technique	[4]
(2018)	---------					0.92%	Personal computer system	CNN with Normalization Technique	[6]

The proposed model (IDD-HPO) model used hyperparameter optimization which achieves high accuracy for ML algorithms as follows: LR=0.671%, KNN=0.883%, SVM=0.901%), and for reported high accuracy for DL algorithms as follow: RNN=0.854%, and CNN=0.963%. It also comes with the advantage of hyperparameter which is used to control the learning process and evaluates the problem to finding a set of optimal hyperparameters and return the best performance with the smallest time delay and energy consumed.

VI. Results and Discussion

In the first experiment, DL and ML performed higher when ((IDD-HPO)) was used in a personal computer compared to state-of-the-art models. DL algorithms especially CNN reported an overall accuracy of 0.963% using (IDD-HPO) model. But according to performance metrics, time delay and energy consumed for (IDD-HPO) model in DL (CNN) was very high, 660 minutes for time delay and 895 joules for energy consumed.

In the second experiment, (IDD-HPO) model was used for ML and DL algorithms in the MCC system to improve performance matrices by decreasing time delay and energy consumed, at the first data sets must be stored in the cloud according to how to map mobile data item taking intoaccount the limited resources of a mobile device by selecting the most suitable storage level for each data item to decreasing the time of training band testing for hyperparameter tuning. The result was that time delay and energy consumed for (IDD-HPO) model in a cloud environment for DL (CNN) was 220 minutes and 201 joules respectively if data store in the cloud as fcs and 290 minutes and 252 joules, respectively if data store in the cloud as bcs.

Based on these results, (IDD-HPO) model can be used to improve the prediction of hospital readmission in two cases (personal computer and cloud environment) with an accuracy of 0.963% with the smallest time delay and energy consumed if the process of training and testing will be done in a cloud environment based on storing data as fcs.

VII. Conclusion and Future Work

In this paper, the proposed model ((IDD-HPO)) using hyperparameter optimization with ML and DL to improve prediction of hospital readmission over a clinical data set, after applying some preprocessing on the input data then ((IDD-HPO)) is using with ML classifiers (e.g., LR, KNN, and SVM) and DL algorithms (e.g., CNN, RNN) to improve the performance of models. The performance of model was tested and evaluated under two different environments.

The proposed (IDD-HPO) model is successful to improve the accuracy of prediction of hospital readmission in a personal computer compared to state-of-the-art models. Then improve time delay and energy consumed by decreasing them if (IDD-HPO) model performed in cloud based on storing data at fcs. That will have a strong effect on the health care costs and the hospital's efficiency and reputation.

As future work, the (IDD-HPO) will be improved to not only be limited to numbers and text data but also to apply to dynamic audio and video data with the use of hybrid storage in the cloud according to the importance of the data used.

References

[1] UCI Machine Learning Repository: Diabetes 130-US Hospitals For Years 1999-2008 Data Set. https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+ years+1999-2008. [Online] [accessed: 5 - 7 - 2020].

[2] C. Neto, F. Senra, J. Leite, N. Rei, R. Rodrigues, D. Ferreira, and J. Machado. Different Scenarios for the Prediction of Hospital Readmission of Diabetic Patients. Journal of Medical Systems, 45(1): 1-9, 2021.

[3] G. S. Shankar, and K. Manikandan. Predicting the risk of readmission of diabetic patients using deep neural networks. In Innovations in Computer Science and Engineering, Springer, Singapore, pages 385-392, 2019.

[4] H. N. Pham, A. Chatterjee, B. Narasimhan, C. W. Lee, D. K. Jha, E. Y. F. Wong, and M. C. Chua. Predicting hospital readmission patterns of diabetic patients using ensemble model and cluster analysis. In International Conference on System Science and Engineering (ICSSE), IEEE, pages 273-278, 2019.

[5] L. X. Li, and S. S. Abdul Rahman. Students learning style detection using tree augmented naive Bayes. Royal Society open science, 5(7), 2018.

[6] A. Hammoudeh, G. Al-Naymat, I. Ghannam, and N. Obied. Predicting Hospital Readmission among Diabetics using Deep Learning. Procedia Computer Science, vol. 141, no. November: 484-489, 2018.

[7] H. Zaky. Eman, M. Soliman. Mona, K. Elkholy. A., and I. Ghali. Neveen. Enhanced Predictive Modelling for 30-Day Readmission Diabetes Patients Based on Data Normalization Analysis. International Journal of Intelligent Engineering and Systems, 14(4): 204-216, 2021.

[8] T. Goudjerkan, and M. Jayabalan. Predicting 30-day hospital readmission for diabetes patients using multilayer perceptron. International Journal of Advanced Computer Science and Applications, 10(2), 2019.

[9] K. W. Bowyer, N. V. Chawla, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research, 16: 321-357, 2002.

[10] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1): 1929-1958, 2014.

[11] S. Shukla, and S. P. Tripathi. EmbPred30: Assessing 30-Days Readmission for Diabetic Patients Using Categorical Embeddings. In Smart Innovations in Communication and Computational Sciences, Springer, Singapore pages 81-90, 2021.

[12] J. Jordan. Hyperparameter tuning for machine learning models. Retrieved from: Jeremy Jordan: https://www. jeremyjordan. me/hyperparameter-tuning, 2017.

[13] J. Brownlee. How to grid search hyperparameters for deep learning models in python with keras. línea]. Disponible en: https://machinelearningmastery. com/grid-search-hyperparameters-deep-learning-models-python-keras, 2016.

[14] C. Chopra, S. Sinha, S. Jaroli, A. Shukla, and S. Maheshwari. Recurrent neural networks with non-sequential data to predict hospital readmission of diabetic patients. In Proc. of International Conerence on Computational Biology and Bioinformatics, Newark, NJ, USA, pages 18–23, 2017.

[15] A. Hegde, and A. Festag. Artery-C: An OMNeT++ Based Discrete Event Simulation Framework for Cellular V2X. In Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pages 47-51, 2020.

[16] X. Zhang, L. Yao, Q. Z. Sheng, S. S. Kanhere, T. Gu, D. Zhang. Converting your thoughts to texts: Enabling brain typing via deep feature learning of eeg signals. In international conference on pervasive computing and communications (PerCom), IEEE, pages 1-10, 2018.

[17] X. Zhang, L. Yao, C. Huang, Q. Z. Sheng, and X. Wang. Intent recognition in smart living through deep recurrent neural networks.

In International Conference on Neural Information Processing, Springer, Cham, pages 748-758, 2017.

[18] World Health Organization, Global report on diabetes. World Health Organization, 2016.

[19] J. Andreu-Perez, C. C. Y. Poon, R. D. Merrifield, S. T. C. Wong, and G. Z. Yang. Big data for health. IEEE journal of biomedical and health informatics, 19(4): 1193-1208, 2015.

[20] D. Mingle. A Discriminative feature space for detecting and recognizing Pathologies of the vertebral column. International Journal of Biomedical Data Mining, 4(114):2, 2015.

[21] Medicare Payment Advisory Commission. Report to the Congress promoting greater efficiency in Medicare. Washington, DC, 2007.

[22] S. Y. Lim, M. M. Kiah, and T. F. Ang. Security Issues and Future Challenges of Cloud Service Authentication. Polytech. Hung. 14: 69–89, 2017.

[23] P. Borylo, M. Tornatore, P. Jaglarz, N. Shahriar, P. Cholda, and R. Boutaba. Latency and energy-aware provisioning of network slices in cloud networks. Computer. Communications, 157: 1–19, 2020.

[24] M. Carmo, F. S. Dantas Silva, A.V. Neto, D. Corujo, and R. Aguiar. Network-Cloud Slicing Definitions for Wi-Fi Sharing Systems to Enhance 5G Ultra-Dense Network Capabilities. Wireless Communications and Mobile Computing, 1–17, 2019.

[25] B. Bhattacharjee, S. Boag, C. Doshi, P. Dube, B. Herta, V. Ishakian, and L. Zhang. IBM deep learning service. IBM Journal of Research and Development, 61(4/5): 1-10, 2017.

[26] A. A. Gad-Elrab, Zaky. E, and Ghali. N. An Adaptive Data Mapping Storage Selection Algorithm in Mobile Cloud Computing. International Journal of Computer Applications 143: 41-47, 2016.

[27] J. Jordan. Hyperparameter tuning for machine learning models. Retrieved from: Jeremy Jordan: https://www. jeremyjordan. me/hyperparameter-tuning. 2017.