IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 2 August 2023, accepted 11 August 2023, date of publication 15 August 2023, date of current version 22 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3305379

RESEARCH ARTICLE

Efficient Deep Learning Models for Predicting Super-Utilizers in Smart Hospitals

MADIHA JAFFAR[1], SUNDAS SHAFIQ[1], NAZIA SHAHZADI[1], NABIL ALRAJEH[2], MOHSIN JAMIL[3], AND NADEEM JAVAID[1], (Senior Member, IEEE)

[1]Department of Computer Science, COMSATS University Islamabad, Islamabad 44000, Pakistan
[2]Department of Biomedical Technology, College of Applied Medical Sciences, King Saud University, Riyadh 11633, Saudi Arabia
[3]Department of Electrical and Computer Engineering, Memorial University of Newfoundland, St. John's, NL A1C 5S7, Canada

Corresponding author: Nadeem Javaid (nadeemjavaidqau@gmail.com)

This project is funded by Researchers Supporting Project number (RSPD2023R648), King Saud University, Riyadh, Saudi Arabia.

ABSTRACT In healthcare, a huge amount is paid to meet the requirements of High-Need High-Cost (HNHC) patients, also known as super-utilizers. The major aim of the proposed study is to predict HNHC patients. This paper proposes hybrid Deep Learning (DL) models that identify HNHC patients, and help the management to manage their resources and budgets accordingly. We use the strategy of the Interquartile Range (IQR) that specifies the range of HNHC. The dataset used in this work has a lot of inconsistencies that are tackled by different techniques. Using three DL models, Gated Recurrent Unit (GRU), Fully Convolutional Network (FCN) and Vanilla Recurrent Neural Network (VRNN), we proposed 2 hybrid DL models, FCN-VRNN and GRU-FCN, for the prediction of HNHC patients. The performance of these models is evaluated based on different performance metrics, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), R-Squared (R^2) and execution time. The results depict that the proposed hybrid models show more competitive and outperforming results than the individual models. Proposed hybrid system model 1 FCN-VRNN achieves MSE of 0.4%, RMSE of 7%, MAPE of 29.8%, MAE of 3%, and R^2 of 99.4%. Proposed hybrid system model 2 GRU-FCN achieves MSE of 0.4%, RMSE of 6.5%, MAPE of 23.55%, MAE of 2.5%, and R^2 of 99.5%.

INDEX TERMS Deep learning, expenditure, FCN, GRU, high-cost, healthcare, super-utilizers, VRNN, smart hospitals.

I. INTRODUCTION

Healthcare is one of the significant and essential aspects of our society. It promotes the well-being of individuals and communities. As the World's population is rapidly increasing, the need for healthcare services is also increasing. A large number of healthcare activities are performed including treatments, diagnosis, etc. Providing quality healthcare is the most critical and expensive task for care providers. The cost associated with healthcare services is becoming exponentially high because of the need for more tools and equipment due to the highly complicated and costly nature of healthcare services. This is true given that healthcare professionals require substantial amounts of skills, education, and expertise to deliver quality care. Healthcare cost is increasing with the passage of time affecting financial management badly in many circumstances. Different events such as earthquakes and COVID-19 also badly affect even the best financial management plans. So, efficient prediction plays an important role to handle these types of situations. Using Artificial Intelligence (AI), risks can be tackled, for example predicting the High Need High Cost (HNHC) patients. High-risk patients having critical situations can be identified using predictive models. Besides, providing interventions can lead to high-quality healthcare. Due to the intricate and diverse structure of healthcare data, predicting high-utilizers presents particular difficulties.

The associate editor coordinating the review of this manuscript and approving it for publication was Junhua Li.

Traditional statistical methods frequently fall inadequate in capturing the complex relationships and patterns identified in the data, which reduces their capacity for prediction.

The rapid growth in Machine Learning (ML) due to advanced algorithms and high computing power helps in the prediction of healthcare data. Different patterns in data can be identified and classified, large amounts of extracted information can be processed, and accurate results can be obtained using ML. However, in traditional ML models, a lot of extensive feature engineering is required [4]. Deep Learning (DL) predictive models help in making better management decisions. In healthcare, decision-making is the most critical step. Efficient management of hospital resources and cutting down on hospital costs are the main aspects of healthcare. In healthcare, the decision-making process is quite intricate. Using the predictive models, the decision-making process can be improved that helps in managing healthcare expenditure [1]. Efficient management of resources is essential for decision-making. In the field of healthcare, AI can accurately detect diseases, injuries and illnesses. Also, the best medical interventions and therapy are proposed to the patients. The use of DL in healthcare is expanding quickly and is assisting in the early diagnosis and forecasting of infectious diseases. To analyze and evaluate the trend of DL approaches in pandemic detection and prediction, DL approaches are proving to be quite useful. Low performance, incorrect labeling, and a lack of high-quality datasets are some of the issues that are efficiently dealt with using the DL approaches. Also, DL approaches are scalable and are continuously improving [2], [3].

A Neural Network (NN) is a solution for dealing with large datasets like healthcare ''inpatient charges'' dataset. ANN basically tries to mimic the brain of a human having the ability to process and learn complex tasks based on input. In structural modeling, Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Multi-Layer Perceptron (MLP), etc., can be applied [5]. The paper's main objective is to identify HNHC patients, who are referred to as super-utilizers and account for a sizable portion of healthcare expenditure. Another objective is to help hospital administration allocate resources and funds efficiently by identifying these patients.

II. PROBLEM STATEMENT

Predicting the healthcare cost is one of the main issues addressed by the authors in [9], which is tackled by CNN predictive model. The model gives good results. However, it works well only on image data and not on textual data. Predicting super-utilizers is one of the most challenging tasks in healthcare. Also, providing patients with adequate care is a significant challenge for healthcare providers and hospitals. Such patients often have complex medical needs and require a huge amount of resources to manage their care leading to high healthcare costs. In [8], authors used Pearson product-moment correlation for predicting the super-utilizers which are sensitive to outliers. This technique is limited to measuring linear relationships and also has assumptions of normality.

TABLE 1. List of abbreviations.

Abbreviation	Definition
ANN	Artificial Neural Network
AI	Artificial Intelligence
Bi-LSTM	Bi-Directional Long Short Term Memory
CNN	Convolutional Neural Network
DNN	Deep Neural Network
EHR	Electronic Health Record
FCN	Fully Covolutional Network
GBM	Gradiant Boosting Machine
GA	Genetic Algorithm
GRU	Gated Recurrent unit
HNHC	High-Need High-Cost
ICU	Intensive Care Unit
LASSO	Regularized Regression
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
MAE	Mean Absolute Error
MLP	Multi Layer Perceptron
VRNN	Vanilla RNN
z	Update Gate
r	Reset Gate
$htbar$	Current Memory Unit
h_t'	Candidate Hidden States
h_t	Hidden State

A. MOTIVATION AND CONTRIBUTIONS

The major contributions of the proposed work are as follows.

- Two hybrid DL models are suggested as a means of predicting HNHC patients. GRU, FCN, and VRNN are combine dto form these models. These three are used both individually and in combined forms for experimental analysis. The predictive accuracy of these hybrid models is increased by utilizing the advantages of several architectures.
- Lot of inconsistencies exist in the dataset, ''Hospital Inpatient Cost Transparency: Beginning 2009'', which must be tackled. Data inconsistencies are addressed using different techniques such as Label Encoding and Interquartile Range (IQR). These methods help to transform inconsistent data, ensuring its reliability and compatibility for analysis.
- The performance of the proposed models is evaluated using Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), R^2, and execution time.

The remaining paper exhibits the following structure. In section II, problem statement is elaborated. In section III, related work is discussed, and the proposed system framework is elaborated in section IV. Explanation of single DL models is provided in section V. The description of algorithms is given in section VI. The dataset details are given in section VII and simulation results are provided in

section VIII, Finally, the conclusion is stated in section IX. In Table 1, all the abbreviations are given.

III. RELATED WORK

In [6], the authors aimed to predict Resource Intensive Healthcare (RIHC) using the US government data, which proves to be a challenging task because the high-utilizers use RIHC in large amounts. Concurrent analysis from five different sources was utilized for prediction. By IQR, a threshold is set for finding super-utilizers among the US counties. Four ML techniques run in a parallel way to check the performance.

Due to the targeted interventions, the healthcare cost is increasing with the passage of time. In [7], the authors proposed a quantized evaluation metric to check whether the solution was cost-effective or not. After evaluation of traditional healthcare and readmission on the impact of ML using the proposed metric, the net healthcare saving was estimated at over $1 million, successfully preventing the rate of readmission by 50%. To forecast the expense of HNHC patients, the ML predictive models are used like Regularized Regression (LASSO), Linear Regression (LR), Gradient Boosting Machine (GBM), and Recurrent Neural Networks (RNN). RNN is a DL sequential model. The results show that the RNN model performs the best of all ML models. The authors looked at the temporal association between healthcare spending throughout a number of time steps. In [8], the authors predicted the cost of a patient at the individual level by an effective method that automatically learned the patterns from different types of time series data using a CNN, in insurance claim data of patients.

In [9], the authors used the CNN model. The architecture consisted of 3 convolutional layers and pooling layers with a Leaky ReLU (LReLU) activation function. The proposed model was compared with the benchmark techniques and the variants like Visual Geometry Group Network (VGGNET), Residual Network (ResNet) and AlexNet. Also, the model was compared with spikes temporal pattern and symbolic temporal pattern. Results showed that the proposed CNN model performed the best of all models. The performance of model was evaluated by MAPE.

In [10], DL was used, which is a prominent innovation that has been applied to time series data. The data is used to forecast the patient's survival chances. For significant indicators of decline, doctors use Electronic Health Records (EHR) data. In recent studies, the authors used RNN models like LSTM, Bi-LSTM, etc [11]. However, they were unable to find the long-term temporal dependencies. To handle this problem, the authors proposed Multi-Head Transformer (MHT). The MHT model was built on a transformer that performed well on time series data and used vital signs to forecast the patient's future variables. To check the learning capability of the model it was trained on different tasks such as the Intensive Care Unit (ICU) to check the remaining length of stay. The model was evaluated on the basis of Area Under the Receiver Operating Characteristic Curve (AUC-ROC), precision, and recall. With several applications, such as care management of accounting, and capitated medical payment adjustment, accurate prediction of patient spending on healthcare is a crucial task. Models based on LR mostly depend on manually created features and require a lot of medical expertise. Poor prediction performance was observed using the proposed approach. In [12], the authors proposed a multi-view DL model using historical claims data. This model was designed to predict the healthcare cost at the individual level. Different types of data were handled by this model like patient information, medical record, drug information, etc.

Predicting the expenditure of medicines using historical claims data using ML techniques is done by a lot of researchers. Generative Adversarial Network (GAN) is a DL model that needs to be explored for time series data. So, in [13], the authors proposed Variance Generative Adversarial Network (VGAN) to minimize the variance (deviation from actual data) during the model training. In the proposed model, the authors used an LSTM as a generator network and CNN or MLP was used as a discriminator network. Results showed that the proposed model performed the best when compared to other GANs and ML techniques like LR, Gradient Boosting Regressor (GBR), LSTM, and MLP.

An ANN was used to predict the inpatient charges, Length Of Stay (LOS), and discharges. Reverse Total Shoulder Arthroplasty (rTSA), and Hemi Shoulder Arthroplasty Hemi (HSA) were put forward to check the internal validity of the patient's required metric values. Overall, the models proposed by authors in [14], performed very well. This study can be used to help doctors and surgeons in finding the associated risks before surgery. Models were evaluated by ROC and AUC curves [15]. This study examined the patterns of how healthcare costs were utilized by super-utilizers. In the US, top 5% cost represents the total of 55% cost of healthcare. So for finding those patients, IQR used a threshold and clustering to predict those patients. With the type of cluster, the outcome of patients also changes with 90-day death rates ranging from 1.0% to 3.2% and comorbidities occurring in a range of 6.9% to 16.5% of cases. Other studies showed that the super-utilizers were of the same type of patients (homogeneous). The study in [16] showed that different types of super-utilizers (heterogeneous) also exist.

Patient's temporal data was used for forecasting the healthcare cost. The data was used in a fine-grain form in [17], and features were extracted by a method called spike detection. It helped in extracting important features from data and learning the patterns so that the prediction performance becomes more accurate. 3 years of data of medical and pharmacy claims were used. MAPE was used for the proposed model's performace evaluation. So enhancing the features like visit patterns and temporal cost improve the performance, meanwhile, the features of medical did not have much impact on the prediction.

In [18], the authors used DL and ML for predicting healthcare costs. They employed DNN and ridge regression. Both models were compared with other techniques, and the results showed that the DNN model outperformed all the other models in term of cost prediction. This superiority is attributed to the DNN's ability to learn more patterns from the data due to its deep structure and higher generalizability as compared to the other models. The authors proposed an efficient system for predicting and preventing dengue. The system proposed in [19] helped in identifying and alerting the individuals affected by the dengue virus with the help of IoT, cloud computing, and fog computing. Naive Bayesian Network (NBN) was used for the prediction of individuals. On the cloud system, social network analysis was used for the Global Positioning System (GPS). Using the past dengue cases, new cases were predicted by the autoregressive method.

One of the most serious and critical diseases is the heart disease. In [20], the authors proposed a system that detected heart disease correctly. The system was structured on the basis of ML techniques like Support Vector Machine (SVM), Logistic Regression, ANN, K-Nearest Neighbor (KNN), Naïve Bayes (NB), and Decision Tree (DT). For the least important features or redundant features, the authors used different techniques like minimal redundancy and maximal relevance. For the validation, they used the 1-leaf cross-validation technique. The least absolute shrinkage selection operator was used for relevant feature selection. Additionally, the method proposed for detecting heart problems in the field of medicine was found to be practical in real-time environments.

Mental health is a very important health factor especially for the young people. Thus, accounting for a large portion of overall healthcare costs. In [21], the authors predicted the high cost of patients by examining different factors of a patient like a diagnosis, demographic factors, etc. The results of the study have consequences for categorizing high-cost service users and determining crucial areas for improvement in the outcomes for the young people. The classification algorithm was used for the prediction of high-cost medical expenditures.

In [22], the authors used different ML algorithms like RF, LR, and XGBoost. Findings showed that XGBoost had excellent performance with 77.7% accuracy. The analysis was performed using national Health Insurance Service (NHIS) of Korea's data from 2010 to 2017. When compared, the outcomes of this study were found more significant than the existing studies.

The authors proposed a DL model in [23] for efficient resource management for medical systems and resource utilization prediction. State-of-the-art techniques usually select the features manually, which requires a lot of domain knowledge. It could handle huge amounts of heterogeneous data like codes, operations, IDs, etc. Hierarchical attention-based neural network transformer captured the hidden patterns of data for accurate prediction.

ML techniques were applied to healthcare claims data. Different algorithms like Random Forest (RF), GBM, logistic regression, and ANN were applied for prediction in healthcare. Data of 3 years, 2016, 2017 and 2018, was used for training the algorithms. The authors claimed that RF performed the best among all the above-mentioned algorithms [24]. AUC curve was used for the evaluation of the model using the validation data. The results showed that the models based on trees like ANN and LR performed the best.

In [25], the authors proposed a system called Health Cloud. In this model, cloud computing and ML algorithms were used that helped heart patients to predict their health status. The patients could get help from the system at home by giving the details to health system. The authors used different ML algorithms like SVM, K-NN, LR, and GB. To evaluate the performance of these models, different metrics were used like accuracy, precision, recall, specificity, execution time, and memory usage. The simulation results showed that LR gave outstanding results with an accuracy of 85.96%. The model was validated using 5-fold cross-validation.

The authors performed classification on heart disease by GA and a feature selection algorithm. A hybrid approach was proposed in [26] for detecting heart disease at an early stage without involving the heart specialists. Different pre-processing techniques were used on the data before passing it to the model like the chained algorithm was used to handle the missing values. Synthetic Minority Oversampling Technique (SMOTE) was used to balance the class distribution. The simulation results showed that the proposed model achieved the highest accuracy of 86.6%.

In [27], for the purpose of predicting diabetes, the authors presented a stacking model. A correlation technique was utilized to choose the features. The stacking model was made of MLP, SVM, and LR, respectively. On the PIMA Indian diabetes dataset, the proposed model performed better than Adaptive Boosting (AdaBoost). The proposed model achieved the highest accuracy of 78.2%.

IV. PROPOSED SYSTEM MODELS

The working flow of proposed model is dicussed in this section. We proposed 2 hybrid DL models in the underlying work, as shown in Figure 1. Before passing the data to the DL model, it needs to be pre-processed properly. In healthcare data, there is multi-variate time series data like patient's medicine codes, medical and surgical codes, etc. Label encoder is a technique used on string features to extract important information, which can be used for prediction. DL model accepts data in the same format so we need to transform string features into numerical data. The outliers are the data points that lie outside of a specific range. Basically, these points are different from majority of the points and are found by IQR. In the proposed model, this technique is used for finding high-cost patients on the basis of median cost.

In the proposed hybrid system model 1, FCN-VRNN, both FCN and VRNN are combined to predict the high-utilizers. The FCN model is very useful in extracting the

spatial features and VRNN captures the temporal dependencies. By combining them both, we can extract spatial features and capture the hidden patterns from the data. FCN uses a mechanism of parallel processing that provides efficient execution time. VRNN has the ability to handle different variable-length sequential input data due to its architecture. This model gives promising results and is adaptable in the healthcare field.

In the proposed hybrid system model 2, GRU-FCN, the data is given to both models who deeply learn the hidden patterns from the data. Both models are complex and take advantage of each other. GRU uses a gating mechanism to control the flow of information, improve the gradient flow and ensure efficient memory management, which enables improving the generalizability capability of the model, whereas, FCN helps in feature extraction. The results of both models are combined with each other to give the final prediction using the output layer.

In the third model, VRNN-GRU, VRNN and GRU are combined. Both sequential models have flexible architectures and can effectively handle sequential data. Both models strengthen each other efficiently, handle the data and improve long-term dependencies. This hybrid model is used for an experimental analysis.

V. TECHNIQUES

Single DL models are individual DL models that are utilized in the proposed hybrid models.

A. GATED RECURRENT UNIT

Gated Recurrent Unit (GRU) is an advanced version of RNN. GRU works well on sequential data, time series data, and textual data. In [28], GRU consists of two gates, reset gate and update gate. These gated mechanisms are useful for controlling the flow of information. At each time step, to update the hidden state deliberately, this gated mechanism is used, which helps in effectively modeling the sequential data. There are different variations of RNN like LSTM, Bi-LSTM, etc. LSTM [29], which consists of three gates, is also designed to address the problem of vanishing gradients. The hidden state of the GRU contains the information that is stored in the internal cell state. The flow of data is controlled by the following gates of GRU. The description of each gate is given below. The gates are determined using Equations 1 and 2.

1. Update Gate (z): it determines how much historical data must be forwarded to the future. It is equivalent to the output gate of an LSTM recurrent unit.

$$z\_t = sigmoid(W\_z * [h\_\{t-1\}, x\_t]) \tag{1}$$

2. Reset Gate (r): this gate selects how much of the previous data to remove similar to the way the input gate and forget gate operate in an LSTM recurrent unit.

$$r\_t = sigmoid(W\_r * [h\_\{t-1\}, x\_t]) \tag{2}$$

3. Current Memory Unit ($\bar{h}_t$): it is used to make the closest to the input zero-mean. It means that closest to variance and adds some non-linearity into the reset gate. To reduce the influence of historical knowledge on the data that is to be forwarded, this unit is made the subset of the reset gate. The basic architectures of both GRU and RNN are comparable. The only difference is that GRU consists of gates meanwhile RNN has no gating mechanism.

GRU's hidden state is determined using Equations 3 and 4.

Candidate Hidden State

$$h\_t' = tanh(W\_h * [r\_t * h\_\{t-1\}, x\_t]) \tag{3}$$

Hidden State

$$h\_t = (1 - z\_t) * h\_\{t-1\} + z\_t * h\_t' \tag{4}$$

where W_r, W_z, and W_h are learnable weight matrices, x_t is the input at time step t, $h\_\{t-1\}$ is the previous hidden state, and h_t is the current hidden state [28].

Algorithm 1 Gated Recurrent Unit

Require: Input at time step t: x_t Previous hidden state at time step $t-1$: $\mathbf{h}\_\{t-1\}$
Ensure: Hidden state at time step t: $\mathbf{h}t$
1: Initialize parameters: W_r, W_z, W_h
2: Compute the reset gate: $r\_t = \sigma(W\_r \cdot [\mathbf{h}t-1, \mathbf{x}t])$
3: Compute the update gate: $z\_t = \sigma(W\_z \cdot [\mathbf{h}t-1, \mathbf{x}\_t])$
4: Compute the candidate hidden state: $\mathbf{h}t' = \tanh(W\_h \cdot [r\_t \odot \mathbf{h}t-1, \mathbf{x}\_t])$
5: Update the hidden state: $\mathbf{h}t = (1 - z\_t) \odot \mathbf{h}t - 1 + z\_t \odot \mathbf{h}\_t'$
6: **return** $\mathbf{h}\_t$

B. FULLY CONVOLUTIONAL NETWORK

It is a variant of CNN [30] in which every neuron is connected to every other neuron in the other layer. Traditional CNN frequently includes fully connected layers that do not perform convolutional operations, thus CNN is not fully convolutional [31]. The completely connected layers can also be thought of as convolutions with kernels that cover the entire input area, which is the main idea. These layers have more parameters than their comparable convolution layers.

FCN is a DL model that is developed for image data segmentation. However, we are using it for sequential data analysis, where the input is arranged chronologically. The traditional CNN is used to extract meaningful features and capture temporal dependencies from the sequence of inputs.

The main concept behind the FCN model is that it directly uses 1D convolutional layers on sequential data. The sequence of inputs moved through these convolutional layers produces a feature vector or feature map to identify regional patterns. In this way, the model can learn hierarchical representations of data by applying multiple convolutional layers and work well on unseen data. If the number of layers increases, the model captures global patterns in the data, and helps in better cost prediction while less layers capture local patterns. FCN uses a global pooling operation to sum up all the features learned across the whole sequence. It is the opposite of CNN, which uses fully connected layers at

Algorithm 2 Gated Recurrent Unit for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
function GRU_model(input_shapeof_data)
    model = Sequentialmodel()
    model.add(GRU(Layers))
    model.add(Dense(Layer))
    model. compile
    return model
end function
start_time = current_time()
model = model(X_{train}.input shape of data)
history = model.fit_func(X_{train}, y_{train}, epochs = 20, batch_size = 256)
    Fit the model for training
y_{predict} = model.predict(X_{test})
end_time = current_time()
MSE = mean_squared_error(y_{test}, y_{pred})
Performance Evaluation Metrics:
RMSE = $\sqrt{\text{Mean Square Error}}$
MAPE = mean_absolute_percentage_error(y_{test}, y_{pred})×100%
MAE = mean_absolute_error(y_{test}, y_{predict})
R-Squared = R^2 (y_{test}, y_{predict})
Code for finding high-utilizers
count_high_utilizers = 0
count_low_utilizers = 0
for i in y_{predict} **do**
    if i >= upper_range **then**
        count_high_utilizers = count_high_utilizers
    else
        count_low_utilizers = count_low_utilizers
    end if
end for
total_no_utilizers = length(y_{predict})
Print

the end to produce a fixed-size output. While predicting high-cost patients, FCN learns from all the historical values enabling it to easily predict future costs for future time steps.

1) CONVOLUTION OPERATION

The convolution operation, in [32], for sequential data is similar to the spatial convolution in images, except that it operates in one dimension. The 1D filter is applied to the sequential input, and element-wise multiplications are performed between the filter and the corresponding input sequence elements. The output value at each position is obtained by summing these multiplications. The output of a convolutional layer can be calculated using the convolution operation. The convolution operation is defined in Equation 5, given an input sequence or feature map X and a collection of filters W.

$$Y = X * W \tag{5}$$

Here, Y represents the output feature map or vector, and $*$ denotes the convolution operation.

2) ACTIVATION FUNCTION

The activation function introduces non-linearity into the model to capture complex relationships. In the FCN model, the Rectified Linear Unit (ReLU) activation function is commonly used and can be defined using Equation 6.

$$\text{ReLU}(x) = \max(0, x) \tag{6}$$

3) GLOBAL AVERAGE POOLING

After applying the ReLU activation function, the results are obtained using Equation 7.

$$\text{Y} = (1/N) * sum(X) \tag{7}$$

Here, N is the representation of the length of the sequence or the number of feature maps [32].

One benefit of using an FCN over other traditional neural network topologies, like a fully connected network [33], is that it has the ability to handle inputs of various shapes due to FCN's convolutional layers, which are applied to local portions of the input and can handle inputs of any size. Semantic segmentation is the main application of FCNs. Upsampling, pooling, and convolution are just a few examples of locally connected layers. Avoiding dense layers means using fewer parameters, which speeds up network training. Furthermore, an FCN can handle various input sizes provided all the local connections. The primary distinction between the two models is that FCN contains fully convolutional layers, which means that the network's convolutional layers can receive varying-sized inputs and generate outputs of the same size. CNNs, on the other hand, demand constant input and output sizes. Additionally, FCN employs skip connections and 1×1 convolutions to decrease the number of parameters and improve network accuracy [33], [34]. A complete neural network is developed from several layers that are completely interconnected, connecting every neuron in one layer to every neuron in the other layer.

- FCN has the major advantage of being "structure agnostic", which means that it does not make any special assumptions.
- While being structure agnostic makes fully linked networks generally applicable, such networks do tend to perform worse than specialized networks suited to the structure of a problem space.

C. VANILLA RECURRENT NEURAL NETWORK

It is a type of RNN that accepts different inputs (variables) and gives different outputs (variables) [35], contrast to the vanilla feed-forward NN.

Different types of inputs expect different types of outputs:

One-to-one: a typical feed-forward NN architecture where we expect one output from a single input.

One-to-many: this is comparable to image captioning. One fixed-sized image serves as the input, while the output can be either words or phrases with various lengths.

Many-to-one: this is used to categorize sentiments. Words or even pages of words are anticipated as the input. The result

Algorithm 3 Fully Convolutional Network

Input: Input data, S = time step, b = bias, and W = weight.
Step 1: Initialize loop variables
for $I = 0; I < E; I++$ **do (Loop of output feature map)**
  for $h = 0; h < E; h++$ **do**
    for $m = 0,\ m < M; m++$ **do**
      for $c = 0,\ c < C; c++$ **do**
        for $i = 0,\ i < R; i++$ **do**
          for $j = 0,\ j < R; j++$ **do**
            $O[I][h][m]$ += $W[m][c][i][j]$ * $I[c][I * S + i][h * S + j] + b[m]$
          end for
        end for
      end for
    end for
  end for
end for

Algorithm 4 Fully Convolutional Network for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
function FCN_model(input_shape_data)
  model = Sequential
  model.add(Convolutional layers(Layer))
  model.add(Global average pooling layer(Layer))
  model.add(Dense(Layer))
  return model
end function
start = current_time()
model = model(X_{train}.input shape of data)
history = model.fit_func($X_{\text{train}}, y_{\text{train}}$, epochs = 20, batch_size = 256)
  Fit the model for training
y_{predict} = model.predict(X_{test})
end_time ← current_time()
MSE = Mean_Squared_Error($y_{\text{test}}, y_{\text{pred}}$)
Performance Evaluation Metrics:
RMSE = $\sqrt{\text{Mean Square Error}}$
MAPE = Mean_Absolute_Percentage_Error($y_{\text{test}}, y_{\text{pred}}$) × 100%
MAE = Mean_Absolute_Error($y_{\text{test}}, y_{\text{predict}}$)
R-Squared = R^2 ($y_{\text{test}}, y_{\text{predict}}$)
  Steps for predicting high-utilizers
count_high_utilizers = 0
count_low_utilizers = 0
for i in y_{predict} **do**
  if i >= upper_range **then**
    count_high_utilizers = count_high_utilizers
  else
    count_low_utilizers = count_low_utilizers
  end if
end for
total_no_utilizers = length(y_{Predict})
Print

can be a regression result with continuous values that show the likelihood of feeling happy.

Many-to-many: this paradigm works perfectly for automated translation like Google Translate. The output will be the same statement in English as the input with a modifiable length.

Mathematical Formulation: vector x can be shown as a time step for each method.

$$h_t = f_w\,(h_{t-1}, x_t) \tag{8}$$

The time step of an input vector is represented by x_t, and the time step of a hidden state is represented by $h[t]$. Therefore, we can consider h_{t-1} to be the previous hidden state. Simple matrix multiplication of the input and hidden state by certain weights W results in the formation of the hidden state.

1) FEED FORWARD PROPAGATION

This is an example of a simple one-to-many RNN. If we were to produce $h[t]$, we need some weight matrices, $h[t-1]$, $x[t]$ and a non-linearity *tanh*

$$h_t = tanh\,(W_{hh}h_{t-1} + W_{xh}x_t + B_h) \tag{9}$$

At each time step, we need to produce an output $y[t]$. As it is a one-to-many network, another weight matrix is required that accepts a hidden state [35], and projects it onto the output as given in Equation 10.

$$y_t = W_{hy}h_t + B_y \tag{10}$$

VI. DESCRIPTION OF ALGORITHMS

In algorithm 1 [36], we present the basic steps of GRU. It tells the flow of information and how it is handled using the gated mechanism. As new information comes, the previous hidden state is updated using candidate hidden states. The reset gate ensures how much past information needs to be ignored or discarded and the update gate ensures how much information from the past should be taken along with new input for the prediction. In algorithm 2, we present the steps that how we predict the super-utilizers using the GRU DL model. Steps explain how we use GRU in our research for finding the super-utilizers. Splitting of data into testing and training is done in such a manner that 70% data is used for training and 30% data is used for testing. Then, the GRU module is implemented, and results of every model are validated using MSE, RMSE, MAPE, MAE, R2, and execution time. The same steps are repeated for all models. For finding the high-utilizers, the threshold is set on the median cost using the IQR. If cost is more than $41534.2675, then the patient is regarded as a high-utilizer and vice versa.

In Algorithm 3 [37], we present generic steps of using FCN. In Algorithm 4, data splitting is initially performed followed by the usage of convolution layers for extracting features at each layer. Mechanisms of convolution layers are used to extract important features. In Algorithm 5 [5], simple working of VRNN is provided. In Algorithm 6, simple RNN is used for the prediction. It has a memory that stores previous output. In Algorithm 7, the hybridization of VRNN and GRU models is done using the output of both models and

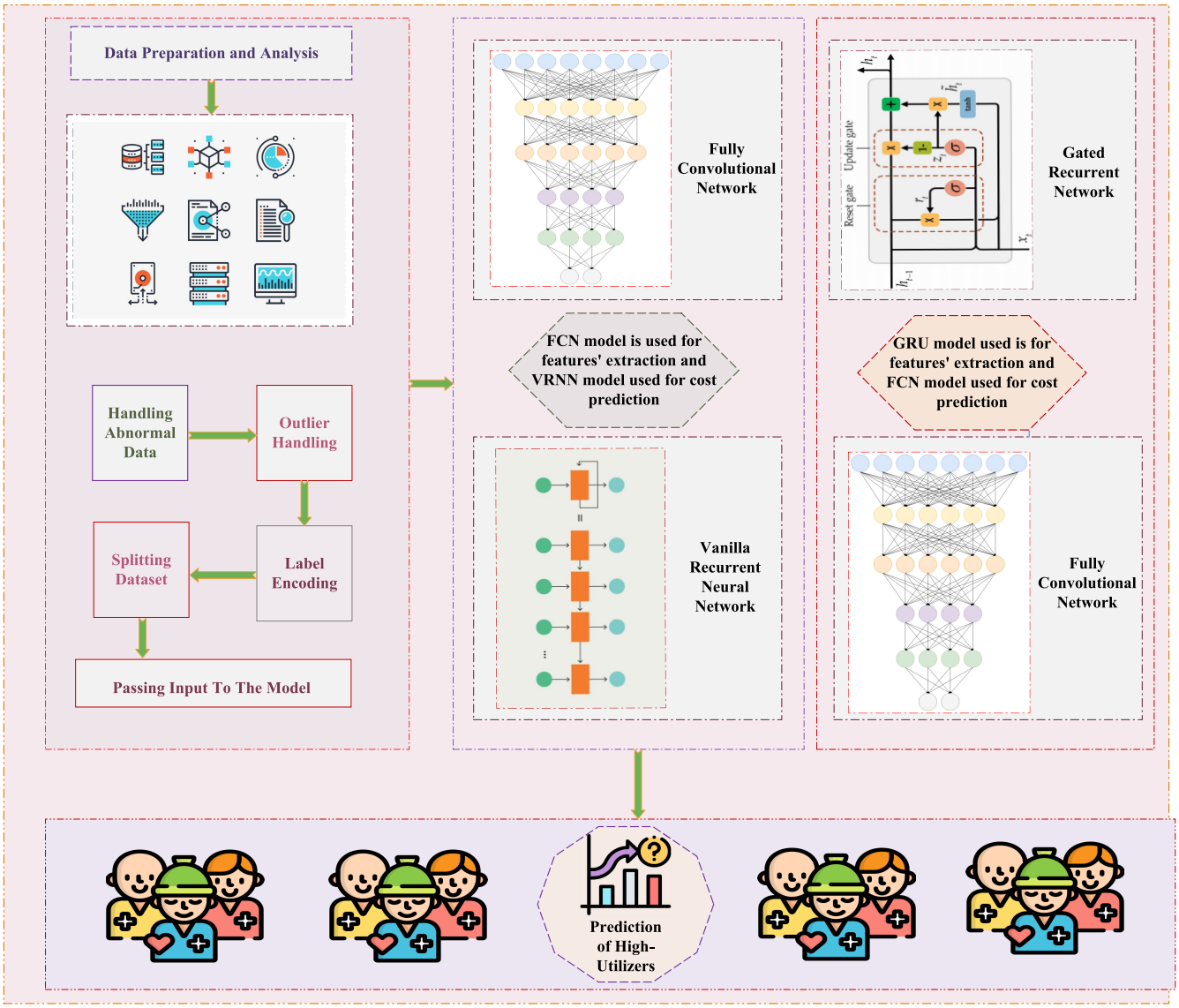


FIGURE 1. Main hybrid system model.

then combining them to get the final output. These 2 models are defined and then concatenated using the merged layer. In Algorithm 9, GRU and FCN are defined and merged to produce the hybrid model. Similarly, the hybrid model of FCN and VRNN is presented in Algorithm 8.

VII. DATASET DESCRIPTION

A detailed description of the dataset used in our research is provided in this section. The dataset plays a crucial role in achieving the objectives of our study.

A. DATA SOURCE

The dataset is obtained from Hospital Inpatient Cost Transparency: Beginning 2009 [39]. This dataset contains information provided by State of New York Article 28 Hospitals as part of the Institutional Cost Report (ICR), and Statewide Planning and Research Cooperative (SPARCS) data submissions. The file of the dataset includes data regarding the Severity Of Illness (SOI) level, hospital discharges, medical or surgical categorization, the median charge, the median cost, the average charge, and the average cost per discharge.

B. DATASET CHARACTERISTICS

Dataset is sequential. It consists of 14 features and is structured in CSV format in which each row represents a specific instance, and each column represents a specific feature. The dataset comprises 1048575 instances collected over a period of 2009-2017. However, we use the data of 2 years only.

- 2015 and 2016 were chosen to capture a representative sample of hospital inpatient cost transparency trends. At the time of our investigation, this represents the most recent years which were accessible in the dataset, as more extensive data can bring inconsistencies in findings.

Algorithm 5 Vanilla Recurrent Neural Network

Ino = No. of input layers
Hno = No. of hidden layers
Ono = No. of output layers
S = No. of data instances
For i = 1 to Hno do
For j = 1 to S do
Calculate the forward pass for the forward hidden layer activation function using h_{tf}
EndFor
EndFor
For j= S to 1 do
Calculate the backward pass for the backward hidden layer activation function using h_{tb}
EndFor
For j= 1 to Ono do
Calculate the forward pass for the output layer using the previous stored output
EndFor

Algorithm 6 Vanilla Recurrent Neural Network Model for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
model = model(X_{train}.input shape of data)
history = model.fit_func(X_{train}, y_{train}, epochs = 20, batch_size = 256)
Fit the model for training
y_{predict} = model.predict(X_{test})
end_time ← current_time()
Performance evaluation metrics
MSE = Mean_Squared_Error(y_{test}, y_{pred})
$RMSE = \sqrt{\text{Mean\_Squared\_Error}}$
MAPE = Mean_Absolute_Percentage_Error(y_{test}, y_{pred}) × 100%
MAE = Mean_Absolute_Error(y_{test}, y_{predict})
R-Squared = R^2 (y_{test}, y_{predict})
Steps for predicting high-utilizers
count_high_utilizers = 0
count_low_utilizers = 0
for i in y_{predict} **do**
  if i >= upper_range **then**
    count_high_utilizers = count_high_utilizers
  else
    count_low_utilizers = count_low_utilizers
  end if
end for
total_no_utilizers = length(y_{Predict})
Print

Algorithm 7 Experimental Model: Vanilla Recurrent Neural Network-Gated Recurrent Unit for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
Define VRNN
model = Sequential()
model.add(simple RNN(Layer))
Define GRU
model = Sequential()
model.add(GRU(Layer))
For hybrid define VRNN and GRU model
model.add(input(Layer))
model.add(merged(Layer))
model.add(output(Layer))
model.compile
return model
start = current_time()
model = model(X_{train}.input shape of data)
history = model.fit_func(X_{train}, y_{train}, epochs = 20, batch_size = 256)
Fit the model for training
y_{predict} = model.predict(X_{test})
end_time ← current_time()
Performance evaluation metrics
MSE = Mean_Squared_Error(y_{test}, y_{pred})
$RMSE = \sqrt{\text{Mean\_Squared\_Error}}$
MAPE = Mean_Absolute_Percentage_Error(y_{test}, y_{pred}) × 100%
MAE = Mean_Absolute_Error(y_{test}, y_{predict})
R-Squared = R^2 (y_{test}, y_{predict})
Steps for predicting high-utilizers
count_high_utilizers = 0
count_low_utilizers = 0
for i in y_{predict} **do**
  if i >= upper_range **then**
    count_high_utilizers = count_high_utilizers
  else
    count_low_utilizers = count_low_utilizers
  end if
end for
total_no_utilizers = length(y_{Predict})
Print

- We have analyzed data within a stable context by considering data of small-time frames and providing a more concentrated analysis and finding's interpretation.
- To provide a baseline for future study and evaluate changes and improvements, high-utilizer patterns, and features for the years 2015–2016 were examined.

C. TARGET VARIABLE AND FEATURES

Our research focuses on the prediction of healthcare costs and high-cost patients. So, the target variable is "Median Cost". It represents the middle value in the dataset. Different sets of features or predictors are included in the dataset like Year, Facility Id, Facility Name, APR DRG Code, APR Severity of Illness Code, APR DRG Description, APR Severity of Illness Description, APR Medical Surgical Code, APR Medical Surgical Description, Discharges, Mean Charge, Median Charge, and Mean Cost. Predicting the median cost can provide useful information about the expected cost. It is very helpful for finding the expenditure, trends, and patterns in the dataset. Figure 2 shows the closest relation between the features and tells which features are the most important for prediction. In this dataset, the model LACE stands for the length of stay, acuity, comorbidities, and emergency department visits. This model helps in providing scoring to data and tells the severity of the disease as used in [6] by different authors. Our dataset is already pre-processed by this model

Algorithm 8 Proposed Hybrid System Model 1: Fully Convolutional Network-Vanilla Recurrent Neural Network for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
Define FCN
model = Sequential()
model.add(3 conv(Layer))
model.add(3 global average pooling(Layer))
Define VRNN
model = Sequential()
model.add(Simple RNN(Layer))
Define For hybrid define FCN and VRNN model
model.add(merged(Layer))
model.add(output(Layer))
model.compile
return model
start = current_time()
model = model(X_{train}.input shape of data)
history = model.fit_func(X_{train}, y_{train}, epochs = 20, batch_size = 256)
Fit the model for training
y_{predict} = model.predict(X_{test})
end_time = current_time()
Performance evaluation metrics
MSE = Mean_Squared_Error(y_{test}, y_{pred})
$RMSE = \sqrt{\text{Mean\_Squared\_Error}}$
MAPE = Mean_Absolute_Percentage_Error(y_{test}, y_{pred}) × 100%
MAE = Mean_Absolute_Error(y_{test}, y_{predict})
R-Squared = R^2 (y_{test}, y_{predict})
Steps for predicting high-utilizers
count_high_utilizers = 0
count_low_ utilizers = 0
for i in y_{predict} **do**
  if i >= upper_range **then**
    count_high_utilizers = count_high_utilizers
  else
    count_low_utilizers = count_low_utilizers
  end if
end for
total_no_utilizers = length(y_{Predict})
Print

Algorithm 9 Proposed Hybrid System Model 2: Gated Recurrent Unit-Fully Convolutional Network for Predicting Super-Utilizers

Require: X_{train}: Training the features (independent features)
Require: y_{train}: Training the target feature (dependent feature)
Require: X_{test}: Testing data independent features
Require: y_{test}: Testing data dependent feature
Define GRU
model = Sequential()
model.add(GRU(Layer))
Define FCN
model = Sequential()
model.add(3 conv(Layer))
model.add(3 global acerage pooling(Layer))
For hybrid define GRU and FCN model
model.add(concatenate(Layer))
model.add(output(Layer))
model.compile
return model
start = current_time() execution_time
model = model(X_{train}.input shape of data)
history = model.fit_func(X_{train}, y_{train}, epochs = 20, batch_size = 256)
Fit the model for training
y_{predict} = model.predict(X_{test})
end_time ← current_time()
Performance evaluation metrics
MSE = Mean_Squared_Error(y_{test}, y_{pred})
$RMSE = \sqrt{\text{Mean\_Squared\_Error}}$
MAPE = Mean_Absolute_Percentage_Error(y_{test}, y_{pred}) × 100%
MAE = Mean_Absolute_Error(y_{test}, y_{predict})
R-Squared = R^2 (y_{test}, y_{predict})
Steps for predicting high-utilizers
count_high_utilizers = 0
count_low_utilizers = 0
for i in y_{predict} **do**
  if i >= upper_range **then**
    count_high_utilizers = count_high_utilizers
  else
    count_low_utilizers = count_low_utilizers
  end if
end for
total_no_utilizers = length(y_{Predict})
Print

and the 'APR Severity of Illness Code' feature is already present. So this helps in efficiently predicting healthcare cost.

D. DATA PREPROCESSING

Prior to conducting the analysis or passing the data into the DL model, we performed numerous preprocessing steps to ensure the quality and integrity of the data. Data preprocessing consists of a set of procedures and methods intended to improve, enhance, and clean up the raw data to make it appropriate for analysis. Data cleansing is the initial stage of data preprocessing.

Step 1: Data Extraction: In order to extract data from the dataset, a threshold year is chosen in this phase. The threshold year is set as 2015, hence two years of data, 2015 and 2016, are taken into consideration. From the total no. of instances of the dataset, i.e., 1048575, 229183 instances are used in the proposed work due to analyzing the data within a stable context. In order to fairly assess a model's capability to generalize new data, the study uses data splitting. 30% of the data is used for testing and 70% of the data is used for training.

Step 2: Columns that are categories or strings must be transformed into a numerical format in order to undertake analysis and modeling tasks. Specifically, the LabelEncoder() function gives each distinct category in the string column a special numerical label.

Step 3: IQR is used in Step 3 to identify outliers. Data points that differ from the rest of the data points in a dataset are known as outliers. They must be properly identified and

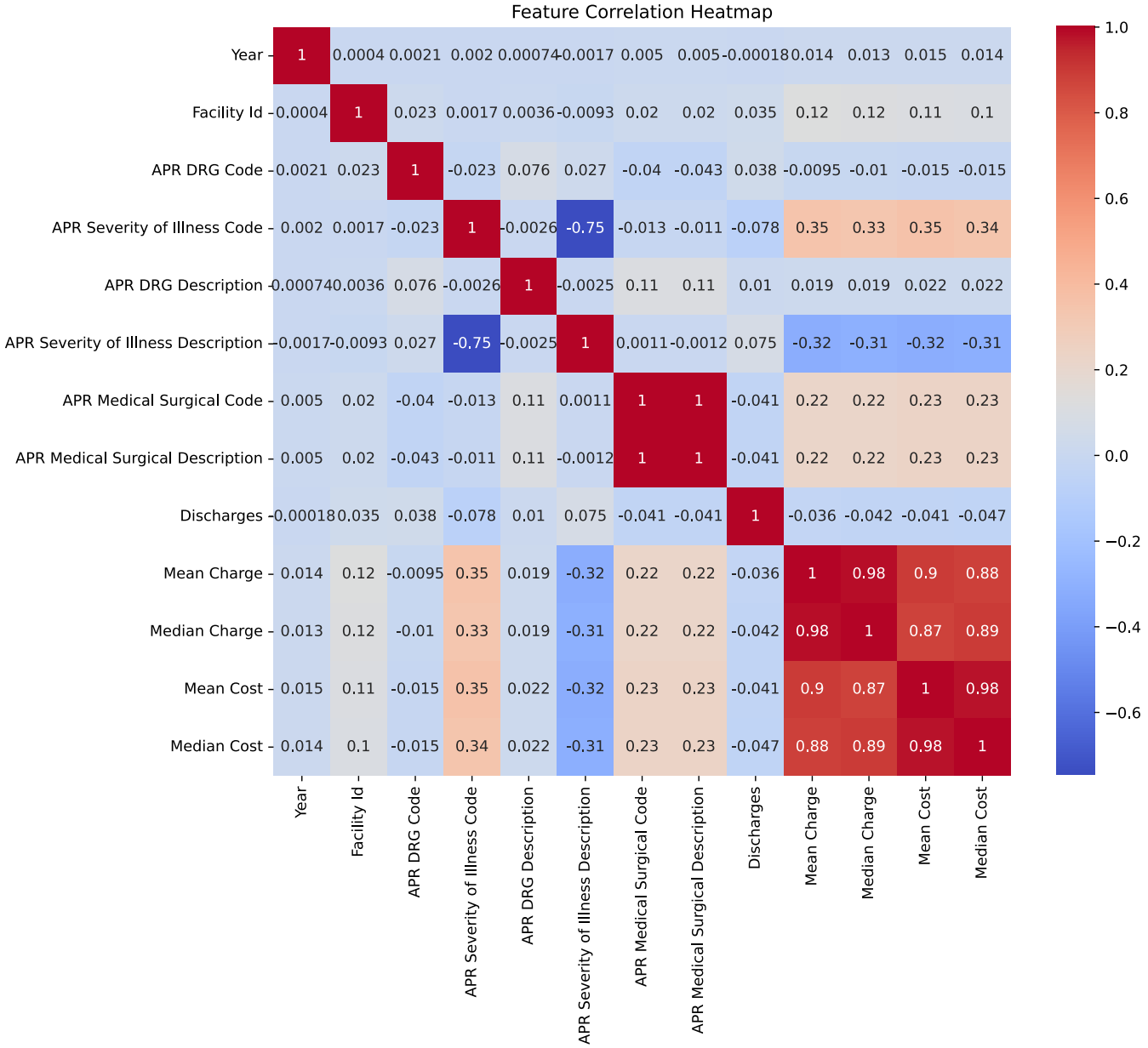


FIGURE 2. **Correlation heatmap.**

dealt with since they have the potential to affect the outcomes of statistical analysis and modeling. The IQR is determined as the spread between a feature's 75th percentile (Q3) and its 25th percentile (Q1). Outliers are data points that are outside of the normal distribution and can be further examined or treated as such. These data points lie outside the range of Q1 - 1.5 * IQR or above the range of Q3 + 1.5 * IQR. This strategy is used for finding high-cost patients. IQR method is used for finding the lower and upper limits for detecting outliers, which are basically the super-utilizers. These limits can be computed as follows:

- Lower Threshold = (IQR*k)/25th Percentile
- Upper Threshold: 75th Percentile + (IQR * k)

Find the data points that exceed the upper threshold to identify the high-cost patients. These patients fall under the category of high-cost patients and are regarded as outliers in terms of their expenses.

The method in [6] identifies patients whose expenditures are significantly higher than those of the dataset's average by the IQR and establishing proper criteria. It provides a statistical way to discriminate high-cost cases and can be helpful in a number of healthcare applications, including cost analysis, resource allocation, and identifying patients in need of specialized care or therapies [6].

As shown in Figure 2, the correlation heatmap explains the relation between different features. There is a strong

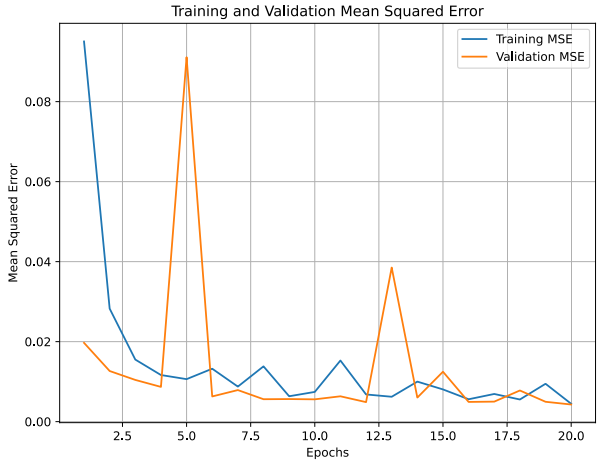


FIGURE 3. Training and validation loss of FCN-VRNN.

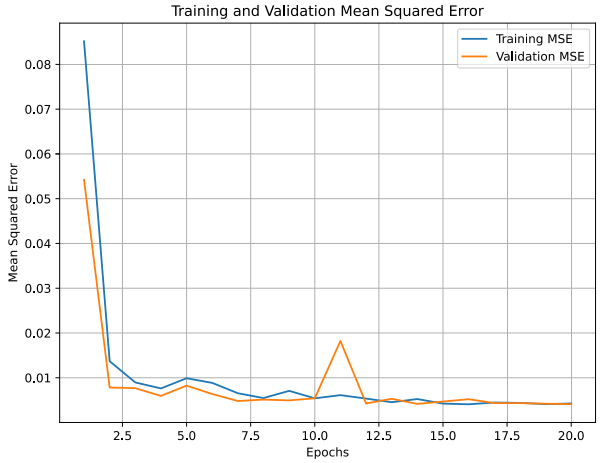


FIGURE 4. Training and validation loss of GRU-FCN.

correlation between the target feature and the input feature. We did not reduce the number of features in our study, and instead of using the Facility Name feature, we used Feature ID. A basic Linear Regression (LR) model might be helpful when there is a strong correlation between the target feature and the input feature. However, there are some limitations of LR that must be considered. LR cannot capture complicated or complex relationships and does not have the generalization ability to provide accurate prediction. It may work well during training but is unable to accurately predict new unseen data. Even though the median cost is highly correlated with the mean cost, real-world datasets contain complex and non-linear relationships, making LR insufficient. DL models can uncover intricate hidden patterns and can capture long-term dependencies, overcoming this limitation. With convolutional layers in the FCN-VRNN and GRU-FCN architectures, these models have two prominent parameters, weights and biases, and flexible architecture of NN. DL models can improve prediction performance without the use of any explicit feature engineering technique.

E. ETHICAL CONSIDERATIONS AND DATA PRIVACY

The dataset is publically available at [39]. Confidentiality agreements must be signed by everyone who has access to hospital inpatient cost transparency data. These agreements set forth specific instructions for data handlers, emphasizing the potential consequences of unauthorized disclosure or misuse of information, and serve as a reminder for maintaining data confidentiality.

VIII. RESULTS AND DISCUSSION

Results of all hybrid DL models are discussed using MAE, MSE, RMSE, R^2, MAPE, and Execution time.

A. PERFORMACE EVALUATION

In this section, we provide a complete overview of the results obtained using DL models.

The performance metrics used are MSE, MAE, RMSE, R^2, and execution time [40]. The formulas are given below.

RMSE is given using Equation 11.

$$RMSE = \sqrt{\sum_{i=1}^{N} (predicted_i - actual_i)^2 / N} \tag{11}$$

The predicted values are those values that we use for testing and actual values are those that we use for training.

MAE is given using Equation 13.

$$MAE = \frac{\sum_{i=1}^{N} |predicted_i - actual_i|}{N} \tag{12}$$

MAE is also the error that computes the difference between predicted values and actual values, given in Equation 13.

$$MAE = \frac{\sum_{i=1}^{N} |e_i|}{N} \tag{13}$$

MSE is given using Equation 14.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (predicted_i - actual_i)^2 \tag{14}$$

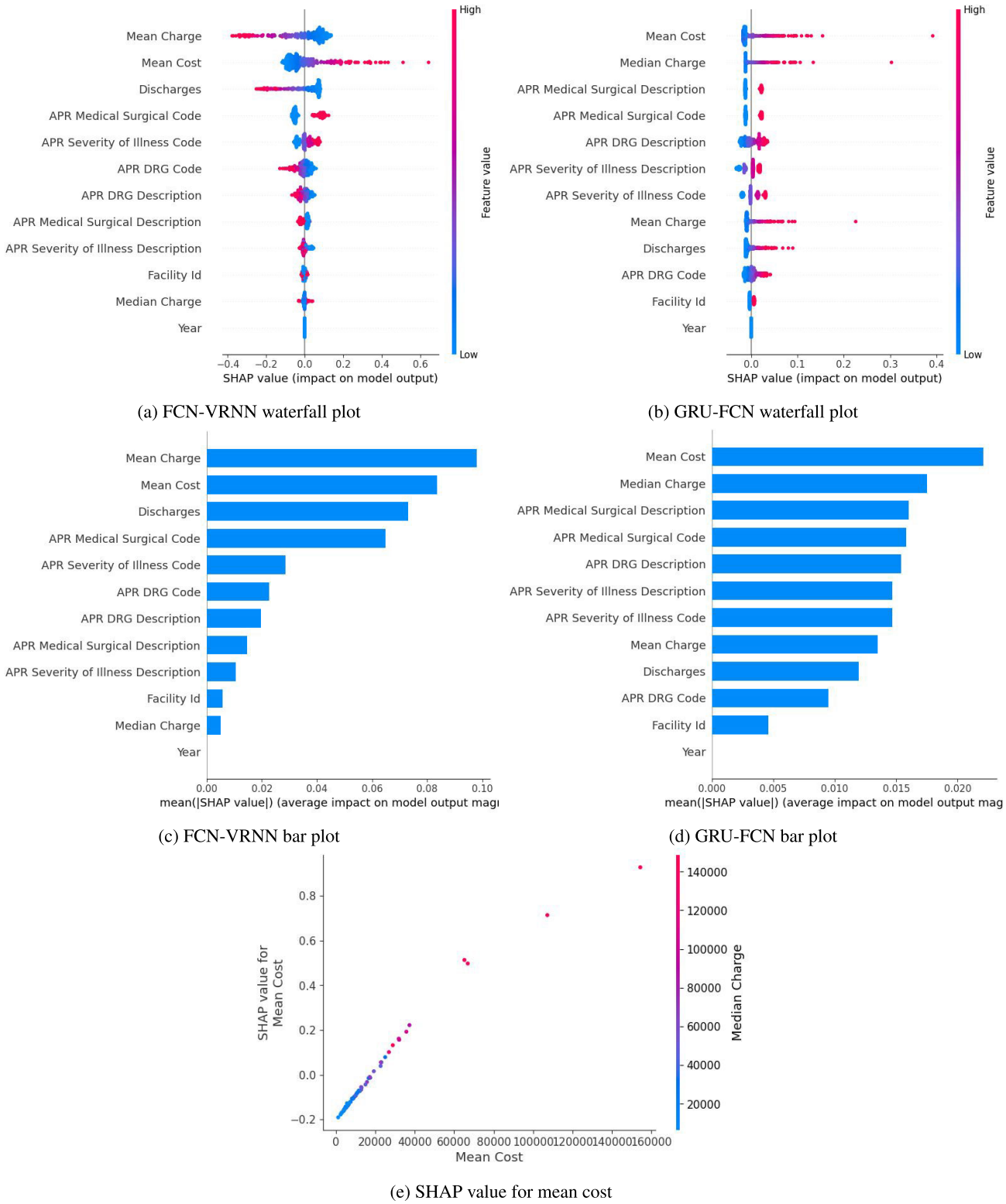
where N = total no. of observations.

R^2 = coefficient of determination is given using Equation 15.

$$R^2 = \text{cor}(actuali, predictedi)^2 \tag{15}$$

Table 2 shows the results of single DL models: VRNN, FCN, and GRU. Performance metrics used for the model's performance evaluation are MAE, MAPE, MSE, RMSE, R^2, and execution time. Table 2 shows that GRU outperformed VRNN and FCN in terms of MAE, MSE, RMSE, MAPE, and R^2. GRU takes more time for execution. Also, VRNN is executed more quickly than FCN. For several runs of simulations, we have plotted the best values.

In Table 3, the results are shown by merging two DL models. These models integrated 2 architectures for achieving the results. These are the combinations of hybrid DL models namely VRNN-GRU, FCN-VRNN, and GRU-FCN.



(a) FCN-VRNN waterfall plot

(b) GRU-FCN waterfall plot

(c) FCN-VRNN bar plot

(d) GRU-FCN bar plot

(e) SHAP value for mean cost

FIGURE 5. Implementation of SHapley in the proposed models.

Figure 3 shows the loss of the training and validation phase, and the way our model minimizes loss through an efficient training approach. The loss curve presents how the training phase progresses over subsequent epochs. The hybrid FCN-VRNN model is helpful for tasks involving both spatial feature extraction and temporal information due to the combination of FCN for important feature extraction and RNN for sequence modeling. By combining losses of both

TABLE 2. Prediction results with single deep learning models.

Performance metrics	VRNN	FCN	GRU
MSE (%)	0.9	0.9	0.4
RMSE (%)	9.9	9.5	6.9
MAPE (%)	60.03	44.94	33.16
MAE (%)	4.4	6.9	3.2
R^2 (%)	97	98.2	96
Execution Time (s)	69.61	158.2	393.7

TABLE 3. Prediction results with hybrid deep learning models.

Performance metrics	VRNN-GRU	FCN-VRNN	GRU-FCN
MSE (%)	0.6	0.4	0.4
RMSE (%)	7.4	7	6.5
MAPE (%)	33.95	29.82	23.55
MAE (%)	3.3	3	2.5
R^2 (%)	99.3	99.4	99.5
Execution Time(s)	212.2	211.2	393.8

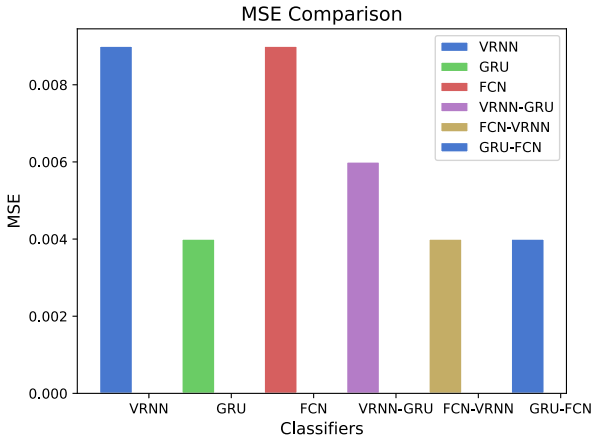


FIGURE 6. Mean squared error.

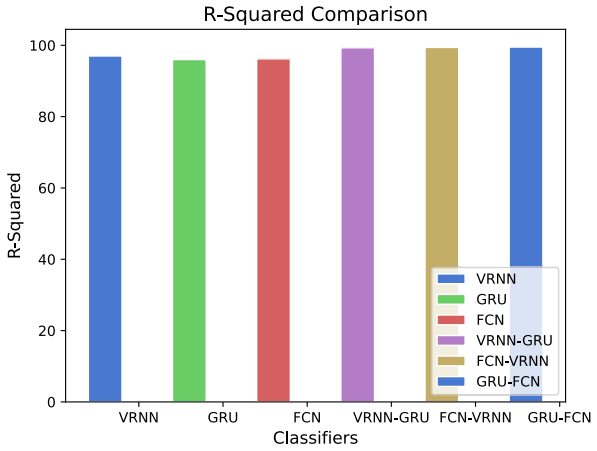


FIGURE 8. R-Squared.

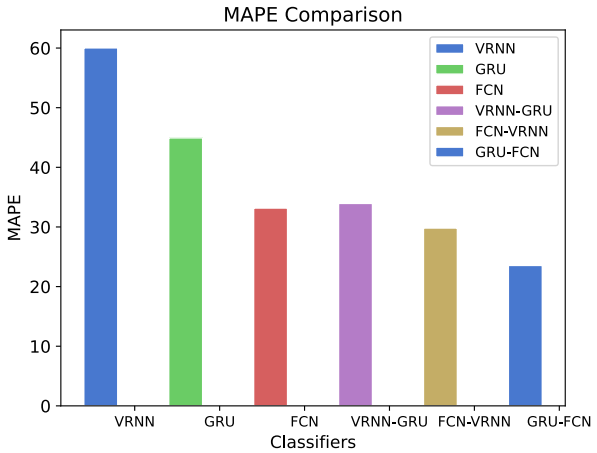


FIGURE 7. Mean absolute percentage error.

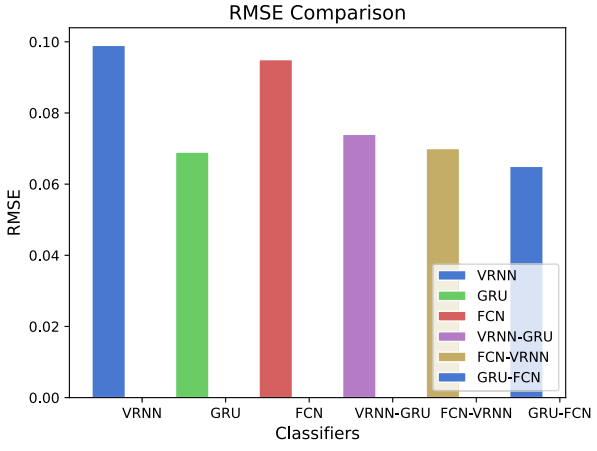


FIGURE 9. Root mean squared error.

components, training loss is calculated, and parameters like weights and biases are changed using backpropagation and optimization methods. The spikes on epochs 9 and 13 are high in the training phase, which means the model is not learning the relevant features. After the 13th epoch, the model's loss is less and model learned the important features. The testing loss of the model assesses performance on unseen data to check the model's generalizability. It also helps to prevent the model from overfitting or underfitting. Similarly, Figure 4 shows the training and validation loss of the hybrid GRU-FCN model, which consistently the difference between the actual and predicted values. Furthermore, we have performed the SHapley Additive Explanations (SHAP) method to show the impact of the features on the output. Figure 5a presents the FCN-VRNN

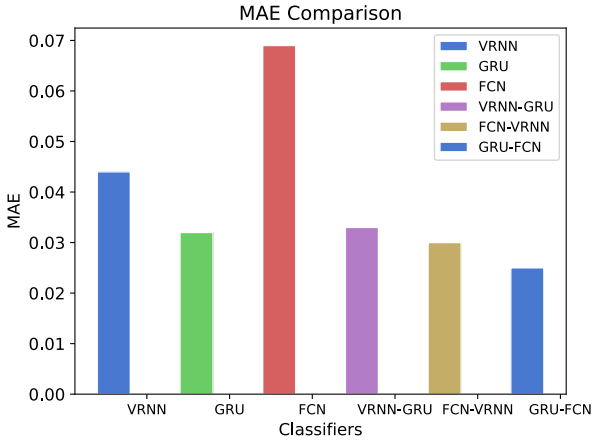


FIGURE 10. Mean absolute error.

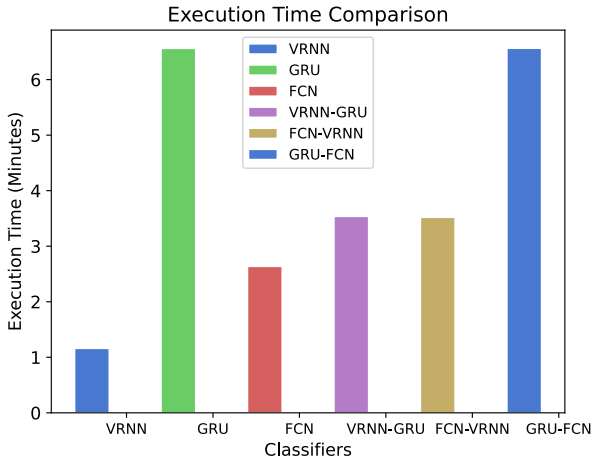


FIGURE 11. Execution time.

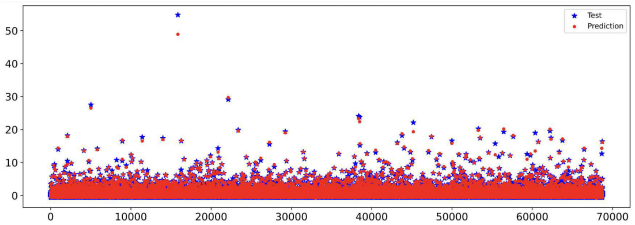


FIGURE 12. Scatter plot of FCN-VRNN.

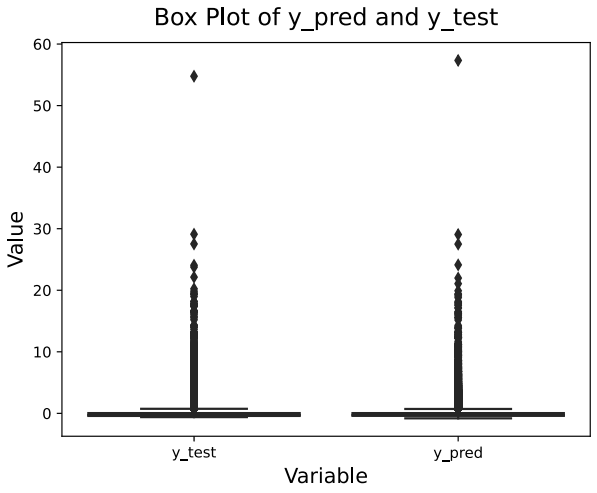


FIGURE 13. Box plot of GRU-FCN.

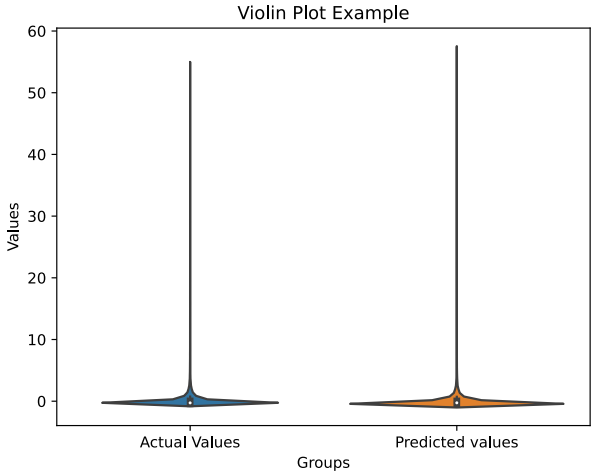


FIGURE 14. Violin plot of GRU-FCN.

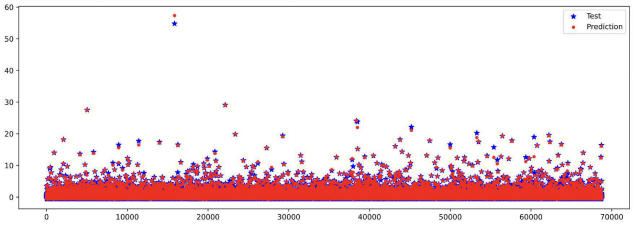


FIGURE 15. Scatter plot of GRU-FCN.

hybrid waterfall plot which helps in the visualization of the internal working of the model. It helps to visualize the contribution of each feature and explain the impact on the final prediction of high-utilizers. These plots are basically associated with local interpretability, which represents the positive and negative effects of each contributor on the prediction. Similarly, Figure 5b demonstrates the GRU-FCN hybrid waterfall plot, which helps to understand the validation and behavior of the hybrid model, and can easily interpret the contribution of every feature. Figure 5c presents the mean importance of the features using a bar plot of FCN-VRNN and determines the average SHAP values for every feature used for predictions. Ranking features in descending order of mean SHAP values highlights the significant impact of features on the output. These plots are associated with global interpretability, and help to find the least important and most relevant features that contribute to the prediction of high utilizers. Similarly, Figure 5d demonstrates the mean importance of the features of GRU-FCN using a bar plot.

As shown in Figures 6, 7, 8, 9, and 10. the hybrid models performed well when as compared to single DL models. GRU-FCN has low values of MSE, MAE, RMSE, R^2, and MAPE followed by FCN-VRNN. If we see the execution time in Figure 11, GRU-FCN takes the largest execution time. However, the error in prediction is very low. GRU-FCN outperforms all other models. Overall, the performance of

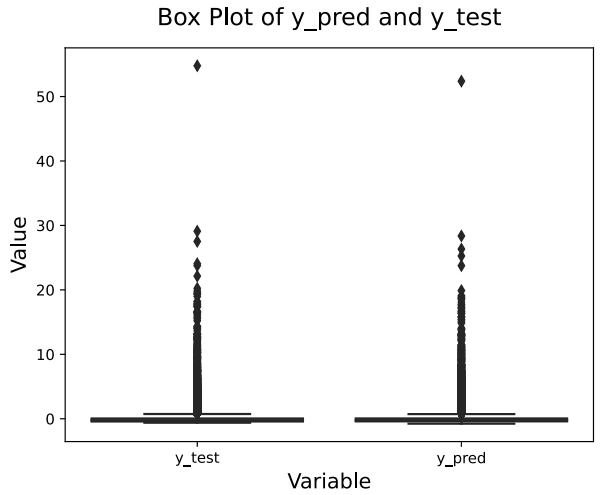


FIGURE 16. Box plot of FCN-VRNN.

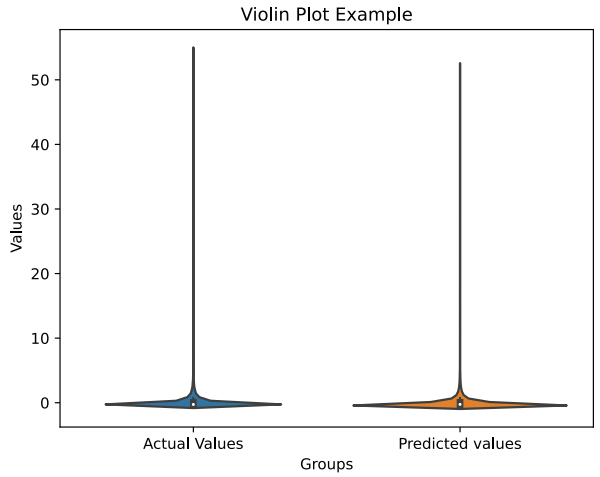


FIGURE 17. Violin plot of FCN-VRNN.

hybrid models is found quite impressive as compared to the performance of the individual models.

Figures 12, 13 and 14 present the scatter plot, box plot and violin plot of GRU-FCN hybrid model. These plots show a minimum difference between actual and predicted values. Similarly, Figures 15, 16, and 17 present the scatter plot, box plot and violin plot of FCN-VRNN hybrid model. We can measure the performance of the proposed models by analyzing these plots.

B. HYPERPARAMETERS

In GRU, epochs are taken to be 20 while the batch size is set as 256. Moreover, the GRU layer, dense layer, and ReLU activation function are used. In FCN, epochs are taken to be 20 while the batch size is set as 256. Moreover, the convolutional layer, average pooling layer, dense layer and ReLU activation function are used. In VRNN, epochs are taken to be 20 while the batch size is set as 256. Moreover, dense layer, simple RNN, and tanh activation function are used. In hybrid FCN-VRNN, filter size 128, Kernel size is 2, concatenate average pooling layer, dense layer, input layers, three Conv1D layers, ReLU activation function, merged layer, and output layer are used. In hybrid VRNN-GRU, no. of epochs is taken to be 20 while the batch size is set as 256. Moreover, merged layers, GRU layers, and input layers are used. In hybrid GRU-FCN, kernel size 2, no. of epochs are taken to be 20 while the batch size is set as 256. Moreover, three convolutional layers, GRU layer, average pooling layer, and the ReLU activation function are used.

IX. CONCLUSION

In this paper, two hybrid deep models, FCN-VRNN and GRU-FCN, are proposed for the prediction of high-utilizers. Before passing the data to the proposed models, we perform different pre-processing steps like converting string columns into numeric using a label encoder and setting a threshold by IQR for predicting high-utilizers. In FCN-VRNN, FCN is used as a feature extractor and VRNN is used for prediction. While in GRU-FCN, GRU is used as a feature extractor and the data is passed to FCN, which extracts the features for prediction. Moreover, simulations are performed and the results show that the proposed models outperform the baseline models in terms of MSE, RMSE, MAE, MAPE, and R^2.

Proposed hybrid system model 1 FCN-VRNN achieves MSE of 0.4%, RMSE of 7%, MAPE of 29.8%, MAE of 3%, R^2 of 99.4% and execution time is 211.2 seconds. The diference between the actal and predicted values is minimum. However, due to the integration of two models, execution time is large. Proposed hybrid system model 2 GRU-FCN achieves MSE of 0.4%, RMSE of 6.5%, MAPE of 23.55%, MAE of 2.5%, R^2 of 99.5% and execution time is 393.8. Similarly, the prediction is closer to the actual prediction and execution time is large due to hybridization of two models.

ACKNOWLEDGMENT

The authors extend their appreciation to the Researchers Supporting Program for Project number (RSPD2023R648), at King Saud University for supporting this research project.

REFERENCES

[1] R. L. Sanz and P. Leon-Sanz, "Modeling health data using machine learning techniques applied to financial management predictions," *Appl. Sci.*, vol. 12, no. 23, p. 12148, Nov. 2022.

[2] S. A. Ajagbe, K. A. Amuda, M. A. Oladipupo, O. F. Afe, and K. I. Okesola, "Multi-classification of Alzheimer disease on magnetic resonance images (MRI) using deep convolutional neural network (DCNN) approaches," *Int. J. Adv. Comput. Res.*, vol. 11, no. 53, pp. 51–60, Mar. 2021.

[3] S. A. Ajagbe and M. O. Adigun, "Deep learning techniques for detection and prediction of pandemic diseases: A systematic literature review," *Multimedia Tools Appl.*, pp. 1–35, May 2023.

[4] Z. A. Khan, M. Adil, N. Javaid, M. N. Saqib, M. Shafiq, and J.-G. Choi, "Electricity theft detection using supervised learning techniques on smart meter data," *Sustainability*, vol. 12, no. 19, p. 8023, Sep. 2020.

[5] K. Kaushik, A. Bhardwaj, A. D. Dwivedi, and R. Singh, "Machine learning-based regression framework to predict health insurance premiums," *Int. J. Environ. Res. Public Health*, vol. 19, no. 13, p. 7898, Jun. 2022.

[6] I. M. Ricket, M. E. Matheny, T. A. MacKenzie, J. A. Emond, K. L. Ailawadi, and J. R. Brown, "Novel integration of governmental data sources using machine learning to identify super-utilization among U.S. counties," *Intell.-Based Med.*, vol. 7, Jan. 2023, Art. no. 100093.

[7] K. Teo, C. W. Yong, F. Muhamad, H. Mohafez, K. Hasikin, K. Xia, P. Qian, S. Dhanalakshmi, N. P. Utama, and K. W. Lai, "The promise for reducing healthcare cost with predictive model: An analysis with quantized evaluation metric on readmission," *J. Healthcare Eng.*, vol. 2021, pp. 1–10, Nov. 2021.
[8] C. Yang, C. Delcher, E. Shenkman, and S. Ranka, "Machine learning approaches for predicting high cost high need patient expenditures in health care," *Biomed. Eng. OnLine*, vol. 17, no. S1, pp. 1–20, Nov. 2018.
[9] M. A. Morid, O. R. L. Sheng, K. Kawamoto, and S. Abdelrahman, "Learning hidden patterns from patient multivariate time series data using convolutional neural networks: A case study of healthcare cost prediction," *J. Biomed. Informat.*, vol. 111, Nov. 2020, Art. no. 103565.
[10] G. Harerimana, J. W. Kim, and B. Jang, "A multi-headed transformer approach for predicting the patient's clinical time-series variables from charted vital signs," *IEEE Access*, vol. 10, pp. 105993–106004, 2022.
[11] R. Khalid, N. Javaid, F. A. Al-Zahrani, K. Aurangzeb, E.-U.-H. Qazi, and T. Ashfaq, "Electricity load and price forecasting using jaya-long short term memory (JLSTM) in smart grids," *Entropy*, vol. 22, no. 1, p. 10, Dec. 2019.
[12] X. Zeng, S. Lin, and C. Liu, "Multi-view deep learning framework for predicting patient expenditure in healthcare," *IEEE Open J. Comput. Soc.*, vol. 2, pp. 62–71, 2021.
[13] S. Kaushik, A. Choudhury, S. Natarajan, L. A. Pickett, and V. Dutt, "Medicine expenditure prediction via a variance-based generative adversarial network," *IEEE Access*, vol. 8, pp. 110947–110958, 2020.
[14] J. M. Karnuta, J. L. Churchill, H. S. Haeberle, B. U. Nwachukwu, S. A. Taylor, E. T. Ricchetti, and P. N. Ramkumar, "The value of artificial neural networks for predicting length of stay, discharge disposition, and inpatient costs after anatomic and reverse shoulder arthroplasty," *J. Shoulder Elbow Surg.*, vol. 29, no. 11, pp. 2385–2394, Nov. 2020.
[15] A. Ahmad, N. Javaid, N. Alrajeh, Z. Khan, U. Qasim, and A. Khan, "A modified feature selection and artificial neural network-based day-ahead load forecasting model for a smart grid," *Appl. Sci.*, vol. 5, no. 4, pp. 1756–1772, Dec. 2015.
[16] J. M. Hyer, A. Z. Paredes, S. White, A. Ejaz, and T. M. Pawlik, "Assessment of utilization efficiency using machine learning techniques: A study of heterogeneity in preoperative healthcare utilization among super-utilizers," *Amer. J. Surg.*, vol. 220, no. 3, pp. 714–720, Sep. 2020.
[17] M. A. Morid, O. R. L. Sheng, K. Kawamoto, T. Ault, J. Dorius, and S. Abdelrahman, "Healthcare cost prediction: Leveraging fine-grain temporal patterns," *J. Biomed. Informat.*, vol. 91, Mar. 2019, Art. no. 103113.
[18] P. Drewe-Boss, D. Enders, J. Walker, and U. Ohler, "Deep learning for prediction of population health costs," *BMC Med. Informat. Decis. Making*, vol. 22, no. 1, pp. 1–10, Dec. 2022.
[19] S. K. Sood, V. Sood, I. Mahajan, and Sahil, "An intelligent healthcare system for predicting and preventing dengue virus infection," *Computing*, vol. 105, pp. 617–655, Jan. 2021.
[20] J. P. Li, A. U. Haq, S. U. Din, J. Khan, A. Khan, and A. Saboor, "Heart disease identification method using machine learning classification in E-healthcare," *IEEE Access*, vol. 8, pp. 107562–107582, 2020.
[21] K. S. Dickson, N. A. Stadnick, T. Lind, and E. V. Trask, "Defining and predicting high cost utilization in children's outpatient mental health services," *Admin. Policy Mental Health Mental Health Services Res.*, vol. 47, no. 5, pp. 655–664, Sep. 2020.
[22] Y. Choi, J. An, S. Ryu, and J. Kim, "Development and evaluation of machine learning-based high-cost prediction model using health check-up data by the national health insurance service of Korea," *Int. J. Environ. Res. Public Health*, vol. 19, no. 20, p. 13672, Oct. 2022.
[23] K. Yu, Z. Yang, C. Wu, Y. Huang, and X. Xie, "In-hospital resource utilization prediction from electronic medical records with deep learning," *Knowl.-Based Syst.*, vol. 223, Jul. 2021, Art. no. 107052.
[24] B. Langenberger, T. Schulte, and O. Groene, "The application of machine learning to predict high-cost patients: A performance-comparison of different models using healthcare claims data," *PLoS ONE*, vol. 18, no. 1, Jan. 2023, Art. no. e0279540.
[25] F. Desai, D. Chowdhury, R. Kaur, M. Peeters, R. C. Arya, G. S. Wander, S. S. Gill, and R. Buyya, "HealthCloud: A system for monitoring health status of heart patients using machine learning and cloud computing," *Internet Things*, vol. 17, Mar. 2022, Art. no. 100485.
[26] P. Rani, R. Kumar, N. M. O. S. Ahmed, and A. Jain, "A decision support system for heart disease prediction based upon machine learning," *J. Reliable Intell. Environ.*, vol. 7, no. 3, pp. 263–275, Sep. 2021.
[27] S. K. Kalagotla, S. V. Gangashetty, and K. Giridhar, "A novel stacking technique for prediction of diabetes," *Comput. Biol. Med.*, vol. 135, Aug. 2021, Art. no. 104554.
[28] GeeksforGeeks. (2023). *Gated Recurrent Unit Networks*. Accessed: May 2, 2023. [Online]. Available: https://www.geeksforgeeks.org/gated-recurrent-unit-networks/
[29] M. Adil, N. Javaid, U. Qasim, I. Ullah, M. Shafiq, and J.-G. Choi, "LSTM and bat-based RUSBoost approach for electricity theft detection," *Appl. Sci.*, vol. 10, no. 12, p. 4378, Jun. 2020.
[30] *What is a Fully Convolutional Network (FCN)? (no Date) AI Stack Exchange*. Accessed: May 2, 2023. [Online]. Available: https://ai.stackexchange.com/questions/21810/what-is-a-fully-convolution-network
[31] S. Mujeeb, T. A. Alghamdi, S. Ullah, A. Fatima, N. Javaid, and T. Saba, "Exploiting deep learning for wind power forecasting based on big data analytics," *Appl. Sci.*, vol. 9, no. 20, p. 4417, Oct. 2019.
[32] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 640–651, Apr. 2017.
[33] *FCN—Papers With Code (no Date) Papers With Code*. Accessed: May 2, 2023. [Online]. Available: https://paperswithcode.com/method/fcn
[34] N. Javaid, A. Ahmed, S. Iqbal, and M. Ashraf, "Day ahead real time pricing and critical peak pricing based power scheduling for smart homes with different duty cycles," *Energies*, vol. 11, no. 6, p. 1464, Jun. 2018.
[35] *Recurrent Neural Networks*. Accessed: May 2, 2023. [Online]. Available: https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/recurrent-neural-network/recurrent_neural_networks
[36] A. Darmawahyuni, S. Nurmaini, Sukemi, W. Caesarendra, V. Bhayyu, M. N. Rachmatullah, and Firdaus, "Deep learning with a recurrent network structure in the sequence modeling of imbalanced data for ECG-rhythm classifier," *Algorithms*, vol. 12, no. 6, p. 118, Jun. 2019.
[37] *An Efficient Streaming Accelerator for Low Bit-Width Convolutional Neural Networks—Scientific Figure on ResearchGate*. Accessed: May 11, 2023. [Online]. Available: https://www.researchgate.net/figure/Illustration-of-CONV-or-FCNlayersa-Pseudo-code-of-CONV-or-FCN-layers-b_fig1_332076796
[38] *WOA + BRNN: An Imbalanced Big Data Classification Framework Using Whale Optimization and Deep Neural Network—Scientific Figure on ResearchGate*. Accessed: May 11, 2023. [Online]. Available: https://www.researchgate.net/figure/Bidirectional-recurrent-neuralnetwork-pseudocode_fig7_331657959
[39] *New York State Department of Health: Health Data NY (no Date) Hospital Inpatient Cost Transparency: Beginning 2009 | State of New York*. Accessed: May 6, 2023. [Online]. Available: https://health.data.ny.gov/w/7dtz-qxmr/fbc6-cypp?cur=asMEC0Vbaqh&from=root
[40] J. U. Ibeji, T. Zewotir, D. North, and L. Amusa, "Modelling children ever born using performance evaluation metrics: A dataset," *Data Brief*, vol. 36, Jun. 2021, Art. no. 107077.

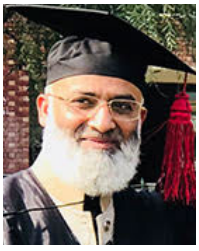
MADIHA JAFFAR received the bachelor's degree in software engineering from COMSATS University Islamabad, Islamabad, Abbottabad, in 2020, and the master's degree in software engineering from the Communications Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, under the supervision of Prof. Dr. Nadeem Javaid. Her research interests include deep learning, healthcare, and artificial intelligence.

SUNDAS SHAFIQ received the M.S. degree in software engineering from the Communication Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad, Pakistan, under the supervision of Prof. Nadeem Javaid, in 2018. Her research interests include fuzzy logic, smart grids, microgrids, and optimal power flow.

NAZIA SHAHZADI received the bachelor's degree in computer science from the University of Central Punjab Lahore, Pakistan, in 2021. She is currently pursuing the M.S. degree in computer science with the Communication Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad, Pakistan, under the supervision of Prof. Nadeem Javaid. Her research interests include artificial intelligence, smart grid, and data science.

NABIL ALRAJEH received the Ph.D. degree in biomedical informatics engineering from Vanderbilt University, USA. Currently, he is a Professor in health informatics with King Saud University. He was a Senior Advisor for the Ministry of Higher Education, his role was implementing development programs, including educational affairs, strategic planning, and research and innovation. He served as a member of the boards of trustees for five private universities in Saudi Arabia. His research interests include E-health applications, hospital information systems, telemedicine, the healthcare applications of smart cities, and wireless sensor networks.

MOHSIN JAMIL is currently an Associate Professor with the Department of Electrical and Computer Engineering, Memorial University of Newfoundland, Canada. He serves as an Associate Editor for IEEE Access, IEEE Canadian Journal of Electrical and Computer Engineering, and *Energies*. His research interests include control system techniques for different applications, primarily focusing on power electronic converters for smart grids, hybrid renewable energy systems and mechatronic systems. He likes to work in other multidisciplinary areas, such as system identification, biomedical engineering, and communication systems using artificial intelligence-based control techniques.



NADEEM JAVAID (Senior Member, IEEE) received the bachelor's degree in computer science and physics from Gomal University, Dera Ismail Khan, Pakistan, in 1995, the master's degree in electronics from Quaid-i-Azam University, Islamabad, Pakistan, in 1999, and the Ph.D. degree from the University of Paris-Est, France, in 2010. He has teaching and research experience of 25 years. He was a Visiting Professor with the University of Technology Sydney, Australia. He is currently a tenured Professor and the Founding Director of the Communications Over Sensors (ComSens) Research Laboratory, Department of Computer Science, COMSATS University Islamabad, Islamabad Campus. He has supervised 187 master's and 30 Ph.D. theses. He has authored over 950 papers in technical journals and international conferences. His research interests include energy optimization in smart/microgrids and wireless sensor networks using data analytics and blockchain. He was a recipient of the Best University Teacher Award (BUTA'16) from the Higher Education Commission (HEC) of Pakistan, in 2016, and the Research Productivity Award (RPA'17) from the Pakistan Council for Science and Technology (PCST), in 2017. He is an Editor of *Sustainable Cities and Society* journal.