IJPSR (2023), Volume 14, Issue 8 **(Research Article)**

INTERNATIONAL JOURNAL OF PHARMACEUTICAL SCIENCES AND RESEARCH

Received on 31 October 2022; received in revised form, 28 July 2023; accepted 29 July 2023; published 01 August 2023

DEVELOPMENT AND ASSESSMENT OF DIFFERENT BALANCING TECHNIQUES AND DEEP LEARNING BASED EARLY MORTALITY PREDICTION MODELS FOR ICU IMBALANCE DATA

Babita Majhi and Aarti Kashyap [*]

Department of CSIT, Guru Ghasidas Vishwavidyalay (Central University), Bilaspur - 495009, Chhattisgarh, India.

Keywords:

Mortality prediction, Intensive care unit, Deep learning based classifiers, missing value imputation, Imbalance data

Correspondence to Author:
Aarti Kashyap

Research Scholar,
Department of CSIT,
Guru Ghasidas Vishwavidyalay (Central University), Bilaspur - 495009, Chhattisgarh, India.

E-mail: aarti.kas2009@gmail.com

ABSTRACT: Predicting ICU patient's mortality is an important area of research to assist the clinical staff in decision-making, and subsequently make more exact strategies in recognizing high mortality risk patients. Handling huge clinical data is still a challenge. These data have different issues such as high dimensions, many missing values, imbalanced data, time-series and data recorded irregularly. This paper mainly focuses on developing four different deep learning models: deep neural network (DNN), deep long-short term memory (DLSTM), deep bidirectional long-short term memory (DBLSTM) and deep gated recurrent unit (DGRU) using two standard datasets, the Physionet challenge 2012 and WiDS datathon 2020 to predict the ICU patients mortality. During the simulation study, missing values are handled using k-NN imputation in Physionet and mean imputation in WiDSdatathon, then balancing of the data is done by employing synthetic minority oversampling technique (SMOTE), cost sensitive learning (CSL) and generative adversarial network (GAN). Feature extraction is done by using discrete wavelet transform (DWT) in WiDS datathon. From the simulation study, it is demonstrated that the (SMOTE+DGRU) has obtained AUC, F1-score and accuracy of 0.8081, 0.7964, 0.8081 respectively in 90 epochs for Physionet challenge 2012. Whereas (SMOTE+DBLSTM) has provided AUC, F1-score and accuracy of 0.8724, 0.8739, 0.8724 respectively for same epochs for WiDSdatathon. In overall, it is observed that the SMOTE balancing technique is performing better in comparison to CSL and GAN.

INTRODUCTION: In the past years, the mortality prediction of ICU patients has been an important area in medical research. The serious consideration units are intensive care units (ICU) that treat highly sick patients, who need consistent care to survive [1].

QUICK RESPONSE CODE

DOI: 10.13040/IJPSR.0975-8232.14(8).4170-92

This article can be accessed online on www.ijpsr.com

DOI link: https://doi.org/10.13040/IJPSR.0975-8232.14(8).4170-92

According to US, 10% to 20% of patients are dying in the hospital and more than five million patients are taking admission in the ICU every year [2].

Various types of medical and bio-signal sensors are utilized in the ICU unit, generating more data [3]. Due to a large amount of data, it is challenging for medical care suppliers to work on the current usage of normal measures, models, results of lab tests, and patient-checking signals [4]. Secondly, ventilators and medical equipment are high in cost. Likewise, countless available ICU beds can cause high monetary expenses [5].

Due to the high cost of the ICU, patients are unable to afford it. Also, ICUs have various staffs in contrast with the quantity of patients conceded for standard checking. Mortality prediction will help in managing the costly equipment, manpower, and other resources. Traditionally, several scoring frameworks have been developed to handle the severity of illness scores for predicting a patient's death or survival in the ICU. These scores are Acute Physiology and Chronic Health Evaluation (APACHE) [6], Mortality Probability Model (MPM) [6], Simplified Acute Physiology score (SAPS) [6] and Sequential Organ Failure Assessment (SOFA) [6].

In the present era, several machine learning, and deep learning models are utilized in real-life examples for solving the problem of classification and prediction. Mortality prediction of the ICU patients is a binary classification problem which can be solved by employing different machine learning approaches. On the view of above, two datasets Physionet challenge 2012 [7] and WiDSdatathon 2020 [8] are used for simulation. The Physionet and Kaggle websites have provided challengesto develop different machine learning models for the ICU patient's mortality prediction for the above datasets respectively. The Physionet challenge has also floated two events to find out the results *i.e*. event I and event II. Event I is the measurement performance of the binary classifier and event II is the measurement performance of the risk estimator. Event I is evaluated using two scoring criteria by finding the minimum value between the precision (+P) and sensitivity (Se) and event II, a challenge is given for the modification in Hosmer Lemeshow statistic [9]. Higher the value is better for score 1, which should be closer to 100, and lower the value is better for the score 2, which should be closer to 0.

WiDSdatathon 2020 has provided 91,713 patient's data with 188 features.

Both the datasets are high dimensional, imbalanced, and a synchronized, hence causes more complex computation to the classifier and degrades the performances. Some of the machine learning algorithms which have been used in the past for the purpose are Support Vector Machine, Linear Regression, Multilayer Perceptron, Naive Bayesian, Random Forest, Decision Tree [10].

Deep learning is the state-of-art, a subfield of machine learning used to handle large and complex data. It automatically extracts the hidden information from data and classifies the model [11]. Being motivated by the advantages of deep learning, it is proposed for the ICU patient's mortality prediction for the first time. The primary approaches of this paper is described as follows:

- ❖ Different pre-processing techniques are applied to mitigate imbalance, irregular, and missing data problems exist in the datasets.
- ❖ Three different data balancing techniques: synthetic minority oversampling technique (SMOTE), cost-sensitive learning (CSL) and generative adversarial network (GAN) are used and results are compared.
- ❖ It has proposed four different deep learning models, DNN [12], DLSTM [13], DBLSTM [14] and DGRU [15] for the ICU patients mortality prediction.

The ICU patient's mortality prediction is a binary classification problem, where 0 represents survival and 1 represents the patient's death. The Physionet and WiDSdatathon datasets are huge in size and imbalanced. Some features have over half of (50%) of missing information, making it more challenging to develop a suitable model for it. Most of the authors adopted different strategies to handle the missing data to improve classification performance. Some authors handle time-series data by converting the irregular data into regular data and then using different statistical measures like first, last, minimum, maximum and mean value of the features. Others used imputation methods such as mean imputation and interpolation to handle missing data problems.

Many authors have used different feature selection methods to choose relevant features out of a large number of features available. Some authors have used only set A [3, 17, 21-29], some used set A, B [4, 18-20, 30-36] both and others used all three datasets (A, B and C) [16] of Physionet for training, testing and validating their proposed machine learning models. The review of the literature published in recent past using Physionet data and WiDS data are given as below:

SAITS, a unique technique for missing value imputation in multivariate time series, is proposed. SAITS is based on the self-attention mechanism. With the use of a joint optimization technique, SAITS is trained to extract missing values from a weighted mixture of two DMSA (diagonal masking of self-attention) blocks. DMSA directly captures the feature correlations and temporal connections across time steps, which boosts training efficiency and imputation precision [16]. For PyTorch, the torchtime Python package offers reproducible implementations of the frequently used PhysioNet and UEA&UCR time series categorization repository data sets. There are tools available for working with time series of unequal length that have irregularly sampled data and partial observations. It seeks to facilitate fair model comparisons in this fascinating field of study and streamline access to PhysioNet data [17].

MultiTime Attention Networks, a brand-new deep learning framework is developed. In order to create a fixed-length representation of a time series having a variable number of observations, Multi-Time Attention Networks develop an embedding of continuous time values. Utilizing various datasets, we examine how well this framework performs interpolation and classification tasks [18]. To represent this non-commutativity, [19]employ the free algebra, a traditional mathematical construct. Also, employ low-rank tensor projection compositions to address the inherent computational complexity of this algebra. As a result, scalable, modular building blocks are produced, providing state of the art performance on common benchmarks including multivariate time series classification, mortality prediction, and generative video models.

The ODE-RNNs family of time series models, which uses neural ordinary differential equations to specify the hidden state dynamics (Neural ODEs) is presented. Firstly, it looked into this model as a solo RNN improvement. Also applied this model to enhance the Latent ODEs variational autoencoder model's recognition networks. Latent ODEs offer explicit uncertainty estimates regarding latent states as well as generally interpretable latent states. Both models are appropriate for the irregularly sampled time series data that are frequent in many applications because neither one calls for discretizing observation times or imputing data as a preprocessing step. Finally, they show that the rates of observations can be modelled by combining continuous-time latent states with Poisson processes [20]. Monteiro, *et al.* [4] have mainly focused on three objectives. Firstly, they reduce the dimensions, decreases the uncontrolled variance, and make the model less dependent on the training set. Also, they have employed feature selection and feature reduction techniques and done multivariate data analysis using spectral clustering, principal component analysis (PCA), factor analysis (FA), and Tukey's HSD test. Then different machine learning classifiers are applied to find out the best score using all three datasets (A, B, and C) and reported that the random forest (RF) classifier performs better in set A.

In [21], the authors have proposed a new combined algorithm, named as just-in-time learning (JITL), and one class extreme learning machine (ELM), which predicts the length of days stays in the hospital using set A. In the combination of JITL and one class ELM, the JITL is utilized to look for customized cases for patient and one-class ELM is used to choose if the patient can be delivered within 10 days. The aim of [22] is to investigate, how early it predicts the ICU patient's mortality. The authors have conducted the time series analysis using various data mining techniques during the initial 48 hours of the admission. The experiment is conducted with different models, traditional scoring system, filter, and imbalanced algorithm, SMOTE. The Filter with SMOTE method performs better in comparison to other models.

The cost-sensitive principle component analysis (CSPCA) and chaos particle swarm optimization (CPSO) are adopted in [3] to find out the best solution for this problem using support vector machine (SVM) as classifier. They focus on highly imbalanced ICU big dataset A and used CPSO, for parameter optimization of the SVM classifier. To resolve the challenges of the data, the authors have applied different analytical tools and techniques during pre-processing, feature extraction and feature selection. The authors in [23] have reported Gated Recurrent Neural Network-D (GRU-D) model for mortality prediction. They try to implement multivariate time series data with missing values using GRU-D. Two representations of GRU-D, masking and time interval are integrated with the deep learning model architecture. Hence, it captures long-term temporal dependencies and achieves better performances. A novel machine-learning algorithm, which combines just-in-time learning with extreme learning machine known as JITL-ELM has been proposed in [24] and tries to optimize variables globally. A new framework is proposed in this paper using clustering for mortality prediction.

Classification of imbalanced data is an important problem in the area of clinical or medical data. Designing a new algorithm to resolve the imbalance issue in the clinical data, using different transformation techniques for feature transformation, and hypothesis testing are proposed in [25, 26] has developed two new approaches known as single task transfer learning and multi-task transfer learning using small size and imbalanced class data. They have used patients ICU types (coronary care unit (CCU), cardiac surgery recovery unit (CSRU), medical ICU(MICU) and surgical ICU(SICU)) for the prediction of mortality. Out of these two approaches, multi-task transfer learning has obtained a better score1. A two-phase hybrid framework using clustering for this purpose is developed in [27].

In the starting phase, clustering is used to get the knowledge and in the next phase, classifiers (namely SVM, ANN, and DT) are constructed out of which the SVM classifier performs better. In the preprocessing, a segmentation-based method is used that divides different variables into various segments. To maintain the statistic of the features, maximal-minimal values are considered. A post-surgical decision support system is developed using multilayer neural network [28]. It focuses on the missing values of the variables and divides them into 4 groups – (1) General descriptor variables - taken at the time of admission, (2) Low-sampling variables - variables that have 50% of missing values, (3) Medium sampling variables - mean for each variable less than 15 for each patient and (4) High sampling - extracting statistics (min, max, mean, median, *etc*.) for every variable [4]. The authors in [29] have compared the various preprocessing methods (missing value imputation, outlier detection, feature extraction, *etc*.) with the Box-Cox outlier rejection technique and employed regularized logistic regression model for mortality prediction. The SAPS-I, which is known as baseline algorithm used in [30]. A classifier based on tree is proposed using Bayesian ensemble learning algorithm in [31]. The proposed algorithm has special features that can detect outliers, handle missing values, and automatically normalise the data. To estimate the model performance, Jack-knifing method is applied.

A logistic regression algorithm that trains the classifier and evaluates the classifier's performance using 10 fold cross validation is reported in [32]. Here, missing values are handled using the mean imputation method. The SVM classifier, using general descriptors and time-series data to predict the ICU patient's mortality is done in [33]. To deal with the missing values, "Imputation" method is used where it replaces missing value by "zero". To train the model, six different SVM classifiers are applied. For every SVM classifier, positive (overall samples) and negative (1/6th samples) are taken as the training set. An artificial neural network(ANN) classifier has been developed using the 48 hours observations from the admission in [34]. The ANN model predicts the risk of patient's mortality in hospitals using various physiological variables.

A two-layered neural network model is proposed for the classification, which consists of 15 neurons in each hidden layer. Hundred voting classifiers are trained, and the result of the model is the mean of all hundred voting classifiers. The training and testing of the models use 5 fold cross-validation. Also, fuzzy threshold value determines the neural network output [35]. focuses on time-series data or motiffs and developed Baseline, L1, L2, L3 and L1,2,3 models for predicting the ICU patient's mortality in hospital. It partitions the symbol sequences into different measurements (low measurement, high measurement and medium measurement). The developed models are shown as better than the traditional scoring systems such as SOFA, SAPS-II and APACHE-II. A model based on hidden markovmodel (HMM) and logistic regression (LR) using different vital signs is presented in [36]. The HMM estimates the process hidden states using experimental variables. A framework is proposed in [37] containing an effective feature extraction by interpolating the data, analysis of histogram and temporal data. A cascaded ad a boost model is proposed, which cascades two feature vectors obtained from histogram and temporal analysis. The fuzzy rule based system or fuzzy inference system (FIS) is used in [38] whose coefficients are optimized by using genetic algorithm (GA). The FIS has two options - (1) fuzzy rules and (2) coefficients. The coefficients, convert all the features into fuzzy values which are further processed by fuzzy rules. They also analyse the errors, and each fuzzy rule produces a fuzzy output which may be 'high' or 'very high', 'medium' and 'low' or 'very low'.

A new technique, known as simple correspondence analysis (SCA) is proposed in [39] for the ICU patients mortality prediction. It combines the medical and laboratory data using two different conventional scoring systems such as SAPS-II and APACHE-II. It selects the most important features from the patient's information and identifies the relationship among different features and target variables (in-hospital death). A linear Bayes classifier is proposed to predict mortality in the ICU [40]. During the pre-processing, 935 features are extracted from original features. Further, the features having more than 200 NAN values are removed and reduce it to 352 features. The remaining NAN values are replaced by mean values. A new binary classifier to predict ICU patients mortality admitted in the hospital is used in [41].

A combination of the feature selection criteria, forward sequential selection and logistic regression (FSS-LRM) model is proposed. 32 variables are selected out of 42 for analysis using 10 fold cross-validation on the training set A. To predict the mortality rate of the ICU patient's [42] has developed SVM model. During the preprocessing, each variable's mean, standard deviation, min and max value are extracted for further analysis. The SVM model faces the problem of over-fitting on the training set A. This issue is circumvented using random patterns. An approach is designed based on ANN in [43] to predict the ICU patient's mortality rate in the hospital. The missing values are handled by imputing the mean value and removing outliers. To train the network, 70% of data is used. A logistic regression model is utilized in [44] to predict mortality rate. Three different approaches used are – (1) selection of derived variables using set A, and calculation of first, last, mean, max, min, total time and first difference of each variable. (2) logistic regression model is employed to predict mortality rate using set A [45]. reports a logistic regression model (risk prediction model) for predicting the probabilities of mortality using 30 selected features from patients who admitted in the ICU. Study is done with the adult patients who admitted in the ICU in following categories - (1) Coronary Care Unit (CCU), (2) Medical ICU (MICU), (3) Cardiac Surgery Recovery Unit (CSRU) and (4) Surgical ICU (SICU). Sequence of steps used are: filtering, feature extraction and prediction.

An ensemble model is developed using test-time augmentation (TTA) for tabular data, merging 42 models. Predictive value imputation, or linear regression, and statistical value imputation, or mean/median are both used to handle missing values. Additionally, a number of functionalities are removed based on various factors. With a higher AUC, this ensemble model won the WiDsDatathon 2020 competition on the kaggle website [46]. The remaining section is coordinated as follows: Section 2 deals with the materials and methods which describes collection of data, pre-processing of data, balancing of data, as well the procedure for the development of prediction models. Simulation study, results and discussions are elaborated in section 3. The comparison with existing models is also dealt in this section. In section 4, conclusions are drawn.

MATERIALS AND METHODS:

Methodology for Mortality Prediction: The development of mortality prediction models undergo several steps as shown in **Fig. 1.**

Step 1: Data collection: At first the problem is defined and data are collected,

Step 2: Pre-processing of raw data: Missing values are handled using k-Nearest Neighbour imputation (k-NN) [47] for Physionetand by mean imputation in WiDSdatathon. Balancing of the datasets are done using synthetic minority oversampling techniques (SMOTE) [48], cost sensitive learning (CSL) [49] and generative adversarial network (GAN) [50]. Then the data are scaled using Min-Max normalization method. Finally the relevant features are extracted using DWT [51] from the WiDS data as the number of features are very large. In case of physionet all features are used as input.

Step 3: The selection of the models: DNN, DLSTM, DBLSTM and DGRU are employed as classifiers.

Step 4: Training and Testing: Training and testing of the models are carried with proper tuning of hyper parameters and using 5 fold cross validation over the entire datasets.

Step 5: Performance measures: Different performance measures such as specificity, sensitivity, precision, F1-score, area under curve (AUC) and accuracy are evaluated.

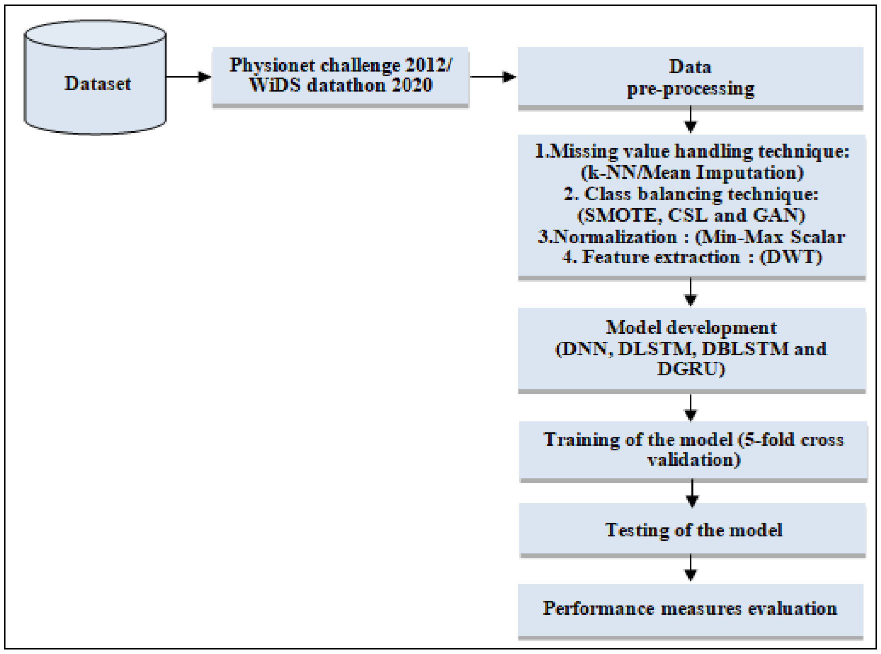

Dataset
Physionet challenge 2012/ WiDS datathon 2020
Data pre-processing
1.Missing value handling technique: (k-NN/Mean Imputation)
2. Class balancing technique: (SMOTE, CSL and GAN)
3.Normalization : (Min-Max Scalar
4. Feature extraction : (DWT)
Model development (DNN, DLSTM, DBLSTM and DGRU)
Training of the model (5-fold cross validation)
Testing of the model
Performance measures evaluation

FIG. 1: STEP BY STEP PROCEDURE FOR THE DEVELOPMENT OF ICU MORTALITY PREDICTION MODELS

ICU Dataset: The datasets are obtained from the Physionet challenge 2012 [7] and WiDSdatathon 2020 [8] websites, which contain data of patients admitted in the ICU for atleast 48 hours and 24 hours respectively. These websites provide all physiological and biomedical information to the authors to develop various machine learning models for mortality prediction, which is helpful for researchers to conduct in-depth research [21]. Physionet challenge 2012 provides three different datasets to the participants, *i.e.* Set A, B, and C. Each dataset consists of 4000 instances of patients. In total 12000 instances are available in set A, B, C and 42 features or variables individually. Out of these 42 features, six of them are general descriptor variables such as Record ID, Age, Height, Weight, Gender (female-0, male-1) and ICU type (Coronary Care Unit (CCU), Medical ICU (MICU), Cardiac Surgery Recovery Unit (CSRU) and Surgical ICU (SICU)), recorded at the time of patient's taking admission in the ICU. The rest 36 variables are time-series variables (multiple times observations are taken) as shown in **Table 1.** Outcome descriptors (in-hospital death) are also provided for each of these three sets A, B, and C. In-hospital death (target variable) comprises of two values (0 and 1), where 0 - indicates the alive of the patients and 1- indicates the deceased of the patients in the hospital during the stay in ICU.

TABLE 1: GENERAL DESCRIPTORS AND TIME-SERIES VARIABLES RECORDED IN THE ICU FOR PHYSIONET DATA [52]

S. no.	Variables	Description	Physical Units
		6-General Descriptor Variables	
1.	Record ID	Unique integer ID	
2.	Age	-	years
3.	Gender	Female(0), Male(1)	-
4.	Height	-	cm

5.	Weight	-	kg
6.	ICU type	CCU, CSRU, MICU and SICU	-
		36-Time-series variables	
1.	Albumin	Albumin	g/Dl
2.	ALP	Alkaline Phosphate	IU/L
3.	ALT	Alanine transaminase	IU/L
4.	AST	Aspartate transaminase	IU/L
5.	Bilirubin	Bilirubin	mg/dL
6.	BUN	Blood urea nitrogen	mg/dL
7.	Cholesterol	Cholesterol	mg/dL
8.	Creatinine	Creatinine	mg/dL
9.	DiasABP	Invasive diastolic arterial blood pressure	mmHg
10.	FiO_2	Fractional inspired oxygen	[0-1]
11.	GCS	Glasgow Coma Score	[3-15]
12.	Glucose	Serum Glucose	mg/dL
13.	HCO_3	Serum Bicarbonate	mmol/L
14.	HCT	Hematocrit	%
15.	HR	Heart Rate	bpm
16.	K	Serum Potassium	mEq/L
17.	Lactate	Lactate	mmol/L
18.	Mg	Serum Magnesium	mmol/L
19.	MAP	Invasive mean arterial blood pressure	mmHg
20.	MechVent	Mechanical Respiration Ventilation	0/1(true/false)
21.	Na	Serum Sodium	mEq/L
22.	NIDiasABP	Non-invasive diastolic arterial blood pressure	mmHg
23.	NIMAP	Non-invasive mean arterial blood pressure	mmHg
24.	NISysABP	Non-invasive systolic arterial blood pressure	mmHg
25.	$PaCO_2$	Partial pressure of arterial carbon dioxide	mmHg
26.	PaO_2	Partial pressure of arterial oxygen	mmHg
27.	pH	Arterial pH	[0-14]
28.	Platelets	Platelets	cells/nL
29.	RespRate	Respiration Rate	bpm
30.	SaO_2	O_2 saturation in haemoglobin	%
31.	SysABP	Invasive systolic arterial blood pressure	mmHg
32.	Temp	Temperature	°C
33.	TropI	Troponin-I	µg/L
34.	TropT	Troponin-T	µg/L
35.	Urine	Urine Output	ml
36.	WBC	White Blood Cells Count	cells/nL

The women in data science (WiDS) datathon 2020 providesa challenge to the participants (especially women's) to develop new machine learning models for the prediction of mortality of ICU patients. The WiDS datathon intends to motivate women's from all over the world to look into data science. The test is to make a model that utilizes information from the initial 24 hours of intensive care unit to predictpatient'smortality. MIT's Global Open Source Severity of Illness Score (GOSSIS) alongside the Harvard Privacy Lab, has gathered the information from various patients of ICU, visiting 1,30,000 clinics during one year [8]. The WiDS dataset consists of 91,713 patient's data with total 188 features and one target variable "hospital_death" which comprises of two binary values 0 and 1 to predict whether the patient will alive or deceased in the hospital. The features are the demographic, vital signs, APACHE covariate, lab blood gas, comorbidity, APACHE prediction and other related information. In this study, sets A, B and Ctogether of Physionetand WiDSdatathon 2020 datasets are used for training and as well as for testing using 5 fold cross validation. Both the datasets are challenging as they have high dimension, imbalanced data and variables with a synchronization of time [4]. In both the datasets, some features have more than 50% missing data, which need to be handled carefully. **Table 2** shows the missing value percentage of time series variables in sets A, B and C of Physionet data and **Table 3** tabulates the missing value percentage in physiological parameters of WiDSdatathon.

TABLE 2: MISSING VALUE PERCENTAGE OF TIME SERIES VARIABLES OF SETS (A,B,C)IN PHYSIONET DATASET

Physiological parameters	Missing value (%)	Physiological parameters	Missing value (%)
Record ID	0.0	pH	24.1
Age	0.0	$PaCO_2$	24.5
ICU Type	0.0	PaO_2	24.6
target	0.0	MAP	30.0
BUN	1.5	SysABP	30.2
Creatinine	1.5	DiasABP	30.2
HR	1.5	FiO_2	32.4
GCS	1.5	MechVent	36.8
Temp	1.5	Gender	43.9
HCT	1.6	Lactate	45.2
Platelets	1.7	Height	47.7
HCO_3	1.7	SaO_2	55.3
Na	1.8	AST	56.6
WBC	1.8	ALT	56.6
K	2.1	Bilirubin	56.6
Mg	2.4	ALP	57.5
Glucose	2.5	Albumin	59.4
Urine	2.8	RespRate	72.3
Weight	7.6	TroponinT	78.0
NISysABP	12.7	Cholesterol	92.1
NIDiasABP	12.8	TroponinI	95.3
NIMAP	12.8		

TABLE 3: MISSING VALUE PERCENTAGE OF PHYSIOLOGICAL PARAMETERS IN WIDSDATATHON 2020 DATASET

Physiological parameters	Missing value (%)	Physiological parameters	Missing value (%)	Physiological parameters	Missing value (%)	Physiological parameters	Missing value (%)
hospital_id	0.0	d1_diasbp_noninvasive_min	1.1	h1_temp_max	23.7	h1_hco3_min	83.0
hospital_death	0.0	d1_heartrate_max	0.2	h1_temp_min	23.7	h1_hemaglobin_max	79.7
age	4.6	d1_heartrate_min	0.2	d1_albumin_max	53.5	h1_hemaglobin_min	79.7
bmi	3.7	d1_mbp_invasive_max	73.9	d1_albumin_min	53.5	h1_hematocrit_max	80.1
elective_surgery	0.0	d1_mbp_invasive_min	73.9	d1_bilirubin_max	58.5	h1_hematocrit_min	80.1
ethnicity	0.0	d1_mbp_max	0.2	d1_bilirubin_min	58.5	h1_inr_max	63.2
gender	0.0	d1_mbp_min	0.2	d1_bun_max	11.5	h1_inr_min	63.2
height	1.5	d1_mbp_noninvasive_max	1.6	d1_bun_min	11.5	h1_lactate_max	92.0
icu_type	0.0	d1_mbp_noninvasive_min	1.6	d1_calcium_max	14.2	h1_lactate_min	92.0
pre_icu_los_days	0.0	d1_resprate_max	0.4	d1_calcium_min	14.2	h1_platelets_max	82.5
readmission_status	0.0	d1_resprate_min	0.4	d1_creatinine_max	11.1	h1_platelets_min	82.5
weight	3.0	d1_spo2_max	0.4	d1_creatinine_min	11.1	h1_potassium_max	78.6
albumin_apache	59.3	d1_spo2_min	0.4	d1_glucose_max	6.3	h1_potassium_min	78.6
apache_2_diagnosis	1.8	d1_sysbp_invasive_max	74.1	d1_glucose_min	6.3	h1_sodium_max	79.2
apache_3j_diagnosis	1.2	d1_sysbp_invasive_min	74.1	d1_hco3_max	16.4	h1_sodium_min	79.2
apache_post_o	0.0	d1_sysbp_max	0.2	d1_hco3_min	16.4	h1_wbc_max	82.8

perative							
arf_apache	0.8	d1_sysbp_min	0.2	d1_hemaglobin_max	13.2	h1_wbc_min	82.8
bilirubin_apache	63.4	d1_sysbp_noninvasive_max	1.1	d1_hemaglobin_min	13.2	d1_arterial_pco2_max	64.6
bun_apache	21.0	d1_sysbp_noninvasive_min	1.1	d1_hematocrit_max	12.7	d1_arterial_pco2_min	64.6
creatinine_apache	20.6	d1_temp_max	2.5	d1_hematocrit_min	12.7	d1_arterial_ph_max	65.6
fio2_apache	77.3	d1_temp_min	2.5	d1_inr_max	63.2	d1_arterial_ph_min	65.6
gcs_eyes_apache	2.1	h1_diasbp_invasive_max	81.7	d1_inr_min	63.2	d1_arterial_po2_max	64.6
gcs_motor_apache	2.1	h1_diasbp_invasive_min	81.7	d1_lactate_max	74.6	d1_arterial_po2_min	64.6
gcs_unable_apache	1.1	h1_diasbp_max	3.9	d1_lactate_min	74.6	d1_pao2fio2ratio_max	72.0
gcs_verbal_apache	2.1	h1_diasbp_min	3.9	d1_platelets_max	14.7	d1_pao2fio2ratio_min	72.0
glucose_apache	12.0	h1_diasbp_noninvasive_max	8.0	d1_platelets_min	14.7	h1_arterial_pco2_max	82.8
heart_rate_apache	1.0	h1_diasbp_noninvasive_min	8.0	d1_potassium_max	10.5	h1_arterial_pco2_min	82.8
hematocrit_apache	21.7	h1_heartrate_max	3.0	d1_potassium_min	10.5	h1_arterial_ph_max	83.3
intubated_apache	0.8	h1_heartrate_min	3.0	d1_sodium_max	11.1	h1_arterial_ph_min	83.3
map_apache	1.1	h1_mbp_invasive_max	81.6	d1_sodium_min	11.1	h1_arterial_po2_max	82.8
paco2_apache	77.3	h1_mbp_invasive_min	81.6	d1_wbc_max	14.4	h1_arterial_po2_min	82.8
paco2_for_ph_apache	77.3	h1_mbp_max	5.1	d1_wbc_min	14.4	h1_pao2fio2ratio_max	87.4
pao2_apache	77.3	h1_mbp_min	5.1	h1_albumin_max	91.4	h1_pao2fio2ratio_min	87.4
ph_apache	77.3	h1_mbp_noninvasive_max	9.9	h1_albumin_min	91.4	apache_4a_hospital_death_prob	8.7
resprate_apache	1.3	h1_mbp_noninvasive_min	9.9	h1_bilirubin_max	92.3	apache_4a_icu_death_prob	8.7
sodium_apache	20.3	h1_resprate_max	4.8	h1_bilirubin_min	92.3	aids	0.8
temp_apache	4.5	h1_resprate_min	4.8	h1_bun_max	81.9	cirrhosis	0.8
urineoutput_apache	53.4	h1_spo2_max	4.6	h1_bun_min	81.9	diabetes_mellitus	0.8
ventilated_apache	0.8	h1_spo2_min	4.6	h1_calcium_max	82.7	hepatic_failure	0.8
wbc_apache	24.0	h1_sysbp_invasive_max	81.7	h1_calcium_min	82.7	immunosuppression	0.8
d1_diasbp_invasive_max	74.1	h1_sysbp_invasive_min	81.7	h1_creatinine_max	81.7	leukemia	0.8
d1_diasbp_invasive_min	74.1	h1_sysbp_max	3.9	h1_creatinine_min	81.7	lymphoma	0.8
d1_diasbp_max	0.2	h1_sysbp_min	3.9	h1_glucose_max	57.4	solid_tumor_with_metastasis	0.8
d1_diasbp_min	0.2	h1_sysbp_noninvasive_max	8.0	h1_glucose_min	57.4		

d1_diasbp_noninvasive_max	1.1	h1_sysbp_noninvasive_min	8.0	h1_hco3_max	83.0

The following section discusses the pre-processing techniques used to deal with the missing data, imbalance and scaling of data.

Pre-processing of Data: Original raw data in Physionetare available in the comma-separated text file format. Each patient's information is recorded in an individual text file. Each text file consists of multiple observations which have been recorded numerous times. For example, the Temperature (C°) of a particular patient is taken 5 times, so five values are recorded. The problem is how to select one value out of these five. Hence, the mean value of temperature is calculated and stored. The same process is repeated for all other variables. **Table 4** presents an example of the variable temperature.

TABLE 4: EXAMPLE OF MEAN VALUE FOR THE VARIABLE TEMPERATURE

Temperature (C°)	
35.2	Mean value of Temperature is **34.88**
35.1	
34.8	
34.5	
34.8	

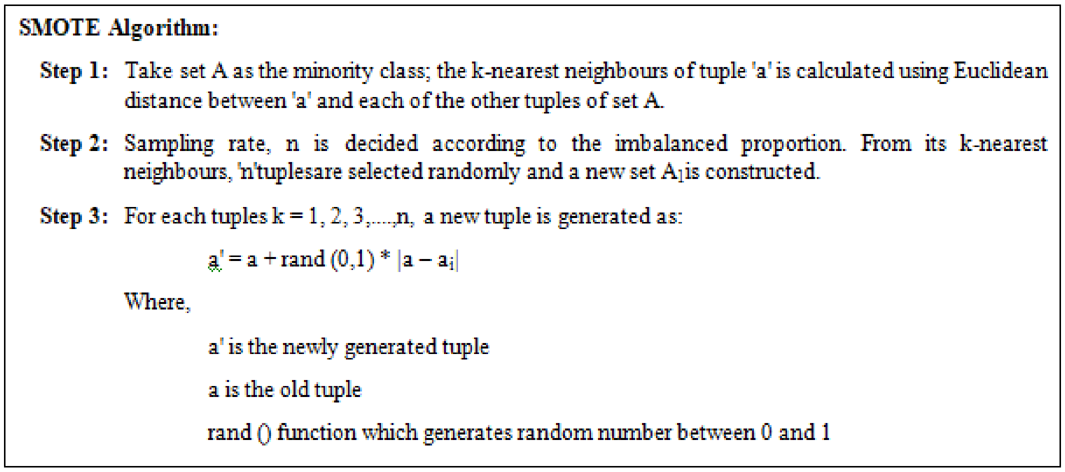
In the same way, all individual text files are converted into comma-separated values in an excel file by calculating the mean value of each feature. The WiDSdatathon is available in. csv file format.

Handling Missing Values: Physionet and WiDSdatathon datasets have many missing values. These missing values can cause problems and may lead to poor performances. Hence, there is a need to identify the missing values and replace them with numeric ones, also known as missing value imputation. One of the popular methods for data imputation is the k-NN imputation algorithm [47] is used in Physionet. The idea behind the k-NN imputation algorithm is to recognize 'k' examples in the dataset that are comparative or nearer in the space. Then, these 'k' examples are used to estimate the missing information of data points. The mean value of the k-points is imputed in each example's missing values. The distance measure used in the calculation is Euclidean distance. In case of WiDSdatathon the missing values are filled using mean imputation as k-NN imputation is very time consuming in this case.

Class Balancing: Class imbalance is a problem where instances of one class are not equal to other classes in the dataset. In the present study, there are two types of classes: majority class (negative class) and minority class (positive class). The ratio between more significant part (majority class) and minor (minority class) indicates the class imbalance ratio. During the model's training, data input should be distributed, *i.e.* minority and majority classe instances should be equal. The problem with the imbalanced dataset is the biasness towards the negative class (majority class) of the dataset and try to neglect minority class, hence degradation in the performance of standard machine learning models [53]. In Physionet dataset, 10,293 (survived patient's) and 1,707 (death patients) and in WiDSdatathon 83,798 (survived patients) and 7,915 (died patient's) are there.

Balancing of imbalanced data can be done using various under sampling and oversampling techniques. In this study, Synthetic Minority Oversampling Technique (SMOTE), Cost-Sensitive Learning (CSL) and generative adversarial network (GAN) are used to balance the datasets, as they are the most widely used effective methods for balancing. The advance version of oversampling technique is the SMOTE. It generates the synthetic data points from the minority class. Its main advantage is that in place of generating duplicate data points, it creates synthetic data points which improves the performance. CSL minimizes the misclassification (false negative) costs and GAN generates the similar data as the original one. All these sampling techniques are explained briefly in the next section.

Synthetic Minority Oversampling Techniques (SMOTE): The SMOTE [48] is widely used in different applications and performs better than the other oversampling techniques. It is one of the oversampling techniques which generates samples from the minority class, artificially. Its primary concept is, framing a new minority class sample by adding a few minority class samples that work together. For each minority sample, k number (taken k=5) of closest neighbors of a similar class is determined. Then, a few tuples are randomly chosen from them as indicated by the oversampling rate. After it, new artificial tuples are produced along with the line in the middle of the minority samples and their closest neighbors are chosen. In this manner, the over fitting issue is mitigated. It causes the limits of the decision for the minority class to increase further into the more significant part of the class region [54]. The algorithm of SMOTE used in the simulation is given below:



SMOTE Algorithm:

Step 1: Take set A as the minority class; the k-nearest neighbours of tuple 'a' is calculated using Euclidean distance between 'a' and each of the other tuples of set A.

Step 2: Sampling rate, n is decided according to the imbalanced proportion. From its k-nearest neighbours, 'n'tuplesare selected randomly and a new set $A_1$is constructed.

Step 3: For each tuples k = 1, 2, 3,.....n, a new tuple is generated as:

$$a' = a + \text{rand}(0,1) * |a - a_i|$$

Where,

a' is the newly generated tuple

a is the old tuple

rand () function which generates random number between 0 and 1

Cost Sensitive Learning (CSL): Cost sensitive learning (CSL) [49] method is used for balancing the data using weighted cost function. Subsequently, misclassifying an example from the smaller class distribution (minority class) will cost the classifiers more than misclassifying an example from the larger class distribution (majority class). Minimizing total cost is the main aim of CSL and ignore the various false negative (misclassification) errors. Most of the sklearn classifiers used class weight as 'balanced'. In the prediction of mortality of ICU patients, afalse negative is of high cost than a false positive since the patient can die because of postponed treatment occasioned by the misclassification.

Generative Adversarial Network (GAN): GAN [50] is a type of machine learning framework which contains two models i.e. generator and discriminator. It is based on two neural networks. The generator is liable for producing new examples from the space (domain), and the discriminator is liable for characterizing whether generated examples are genuine or fake. The center thought of a GAN depends on the 'indirect' preparing through the discriminator, another neural network that can see how muchan information is 'realistic' [55]. This essentially implies that the generator isn't prepared (trained) to limit the distance to a particular data, yet rather it try to fool the discriminator. The working of GAN model is that first input is passed to the generator model which generates the fake data and some training examples are taken from the real data. Both generated data and selected examples are then passes to the discriminator model which predicts whether the data is fake or real based on the probability. The probability of fake data should be near to 0 and real data should be near to 1. When the output of both the data is differ than 0 or 1, there is a need of updating the weight by using back propagation algorithm. It back propagates and trained the discriminator model and updates the weights. In the same way generator model is also back propagates and updates the weights to obtain the real data. The process will continue until the generator and discriminator performs better after every iteration and obtains the real data.

After balancing the data, there is a need to scaled the data between 0 and 1, so the machine can learn and perform easily.

Normalization of Data: Normalization is the process of scaling the data to a specified range as required. All information in the dataset is in various ranges. The min-max algorithm is used to transfer the data to a range of 0-1 using (1).

$$X_{normalized} = (X-X_{min}) / (X_{max}-X_{min}) \times (X_{newmax}X_{newmin}) + (X_{newmin}) \ldots\ldots..(1)$$

Where, $X_{normalized}$ represents the output of the normalization, X is the original input value, X_{min} is the input X's minimum value and X_{max} is the maximum value of the input X [51]. X_{newmin} and X_{newmax} are new minimum and maximum range respectively.

Feature Extraction: After normalizing the data, discrete wavelet transform (DWT) [51], a transform domain technique is used in the WiDSdatathon 2020 dataset to extract features which obtains wavelet coefficients. The DWT decays the signals or information's, and in each degree of decays the energy of various frequencies and time is connected with a particular coefficient. The signal coefficients having huge energy are chosen and those having less level of energy are disposed off. In the WiDS dataset, three levels of DWT are applied on 176 features to extract 22 wavelet coefficients. The main advantage of using DWT is faster processing as comparison to other feature extraction techniques as it splits the data into half of the entire size repeatedly and hence reduced the dimension of the features. Also, the performances of the models are improved and requires less time in execution. In case of Physionet all 40 features are used in simulation without using DWT.

Development of Classifier Models: Deep learning is the state-of-the-art and branch of machine learning, which is motivated by the biological structure of neurons known as artificial neurons. It consists of many layers and handles large and complex data sets. The DNN, DLSTM, DBLSTM and DGRU models are proposed in this study to predict ICU patient's mortality. The explanation of the models are given as follows:

Deep Neural Network (DNN): A simple DNN[12] is made up of three layers – (1) input layer (which receives input), (2) hidden layers (used for extracting the patterns) and (3) output layer (which generates the output). A network with many hidden layers is known as a deep neural network.

In this study, the DNN model consists of 4 hidden layers. There are 60, 50, 50 and 40 neurons in the first to fourth layers, respectively. The input layer focuses on passing the input to the first hidden layer. Then the first hidden layer computes the weighted sum and passes it through the rectified linear unit (ReLu) activation function to the second layer. The same process is repeated for all the rest hidden layers. The last layer is the output layer, which predicts the output (whether the patients will survive (0) or die (1) in the hospital). The activation function in the output layer is the 'sigmoid' used for binary classification. The stochastic gradient descent algorithm is used with momentum for updating the weights. **Fig. 2** shows the proposed deep neural network architecture for mortality prediction.

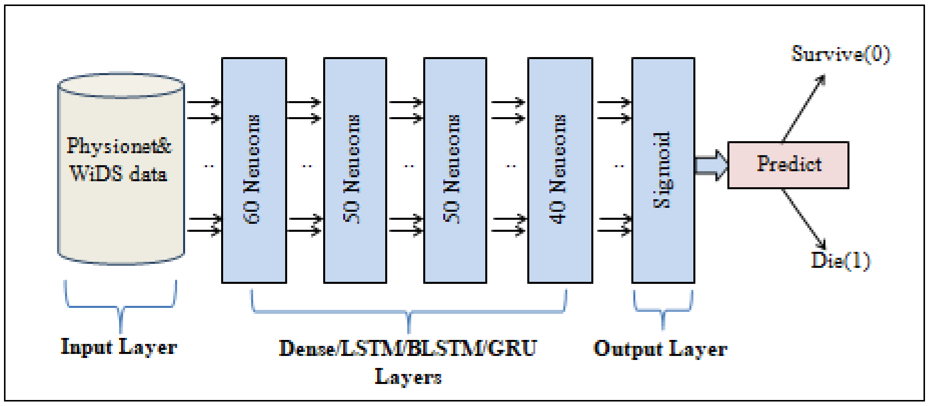


FIG. 2: PROPOSED ARCHITECTURE OF DEEP LEARNING MODELS FOR MORTALITY PREDICTION OF ICU PATIENTS

The following equationis used in DNN for calculating the output:

$$Z = b+w_1\, x_1+w_2\, x_2+\cdots w_n\, x_n \text{......} \quad (2)$$

Where, b = bias weight, w_1.....w_n = weights, n is the number of weights, x_1.....x_n = input, n is the number of inputs, Z = output.

Deep Long-Short Term Memory (DLSTM): Long-short term memory (LSTM) [13] is an advance version of recurrent neural network (RNN) which is used to avoid vanishing gradient problems occurred during the RNN operation. The ability of LSTM is it remembers the information for long terms of time.

In LSTM, the data travels through a framework known as cell states. Thusly, LSTM can explicitly recall or forget to recollect things. The data at a particular cell state has three conditions. These three conditions are the past cell state (the data that was available in the memory after the past time step), past hidden state (this is equivalent to the result of the past cell) and the input of current time step (the new data is entered at that point). LSTM is made up of three gates *i.e.* input gate, forget gate and output gate. Input gate is used to refresh the cell state. For that, it passes the past hidden and current input state in the sigmoid activation function and decides which values will be refreshed by changing the qualities to be somewhere in the range of 0 and 1. Forget gate decides what data ought to be discarded or kept. Data from the past hidden state and current input state is passed using sigmoid activation function. Values emerge somewhere in the range of 0 and 1. Finally, the output gate takes decision of what should be the next hidden state. The LSTM architecture is shown in **Fig. 3**. In this study, the model is made up of six layers *i.e.* one input layer, four hidden layers (LSTM layer) and an output layer.

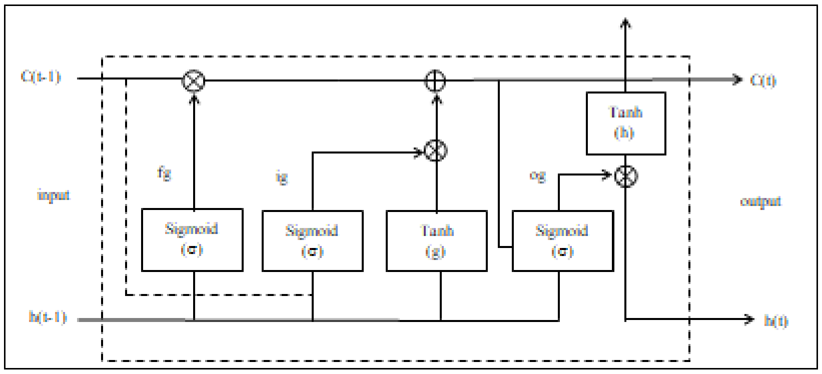


FIG. 3: LSTM NEURON ARCHITECTURE

The following key equations show the working of a LSTM neuron architecture:

Forget gate,

$$fg = \sigma_a (W_{fg} \times x_t + U_{fg} \times h(t-1) + b_{fg}) ...... (3)$$

Input gate,

$$ig = \sigma_a (W_{ig} \times x_t + U_{ig} \times h(t-1) + b_{ig}) ...... (4)$$

Output gate,

$$og = \sigma_a (W_{og} \times x_t + U_{og} \times h(t-1) + b_{og}) ...... (5)$$

$$C'(t) = \sigma_h (W_{ct} \times x_t + U_{ct} \times h(t-1) + b_{ct}) ...... (6)$$

Cell state,

$$C(t) = fg.C(t-1) + ig.C'(t) ...... (7)$$

Hidden state,

$$h(t) = og.\sigma_h (C(t) ...... (8)$$

Where, fg = forgate gate,

ig = input gate,

og = output gate,

C(t) = used to generate C(t) and h(t)

C(t) = cell state,

h(t) = hidden state,

W_{fg}, W_{ig}, W_{og}, U_{fg}, U_{ig}, U_{og} = weight matrices

b_{fg}, b_{ig}, b_{og}, b_{ct} = bias

x_t= input

Deep Bidirectional Long-Short Term Memory (DBLSTM): In bidirectional LSTM [14], data flows in both the directions (forward or backward) to save the previous and the future data. The structure of bidirectional LSTM is same as LSTM (unidirectional) except that it is bidirectional. In this study, model is made up of one input layer, four hidden layer (DBLSTM) and one output layer. The model learns sequentially from both the previous and future values. The output of both the passes (forward and backward) are then combined to produce final results. The final layer is densely connected using sigmoid activation function which gives the output values between 0 and 1. **Fig. 4** shows the architecture of a BLSTM neuron.

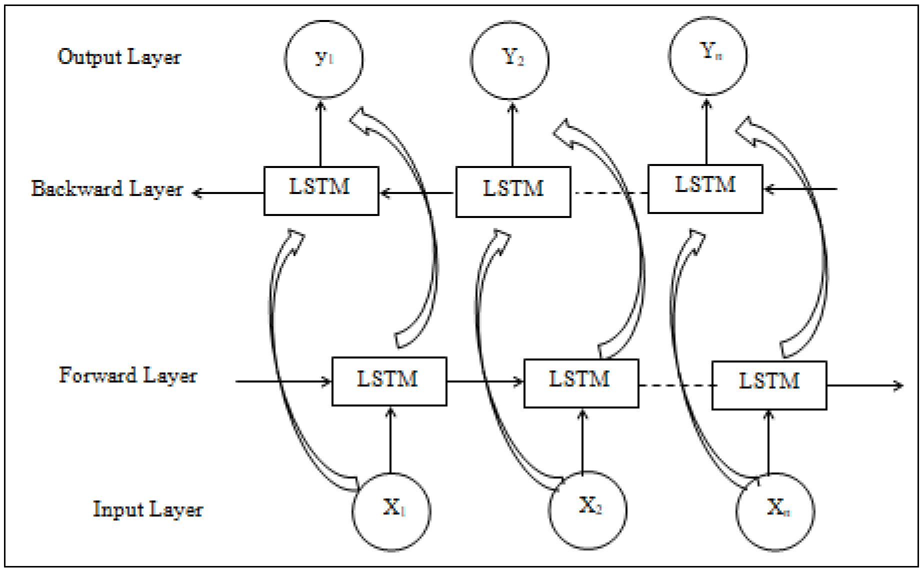


FIG. 4: BLSTM NEURON ARCHITECTURE

The following key equations are used in BLSTM:

Forward hidden layer,

$h_f = \tanh(W_{hf\ xt} + W_{hf}\ h_{f+}b_{f)}$...... (9)

Backward hidden layer,

$h_b = \tanh(W_{hb\ x_t} + W_{hb}\ h_{b+}b_b)$...... (10)

Output,

$y_i = W_{hf}.h_f + W_{hb}\ h_b + b_y$ ...... (11)

Where, h_f = forward hidden layer, h_b = backward hidden layer, y_i = output of both the hidden layer h_f and h_b.

Deep Gated Recurrent Unit (DGRU): The gated recurrent unit (GRU) [15] is a recurrent neural network architecture which is same as LSTM structure. The GRU mainly consists of two gates *i.e* reset gate and update gate in place of the input gate, output gate and the forget gate of the LSTM architecture. The reset gate decides how to join the new contribution with the past memory, and the update gate characterizes the amount of the past memory to keep around. Setting all 1's to reset gate and all 0's to update gate, it behaves like a recurrent neural network architecture. Gated recurrent unit is shown in the **Fig. 5.**

In this study, DGRU based classifier is developed which consists of 6 layers *i.e.* one input layer, four hidden layers (DGRU layers) and one output layer. Here, the number of input is passed to the input layer and then the whole inputs passes to the first hidden layer where weighted sum is calculated and passes through the ReLU activation function to the next hidden layer. The process will repeat until it reaches to the last hidden GRU layer and generate output. The generated output is then passed to the last dense layer (output) which predicts whether the patients will alive or deceased. The final output is compared with the actual output which produces error. Error is minimized using back propagation algorithm.

The advantages of using GRU is, it can be utilized for further developing the memory limit of recurrent neural network as well as it gives the simplicity/ease way of preparing a model. The hidden unit can likewise be utilized for settling the RNN's vanishing gradient problems. It tends to be utilized in different applications, including machine translation, speech signal processing, handwriting recognition *etc.*

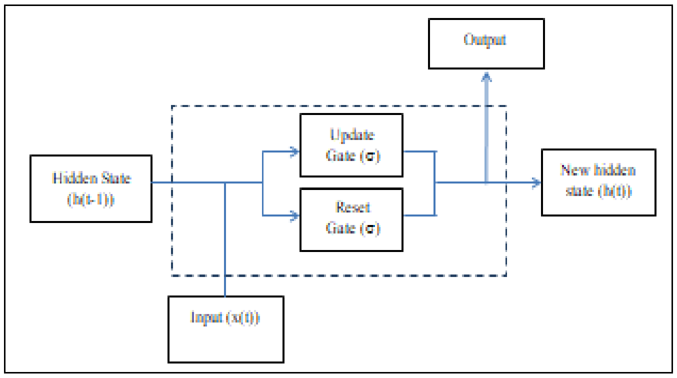


FIG. 5: GRU NEURON ARCHITECTURE

The following key equations are used in GRU architecture:

Reset gate,

$$r(t) = \sigma(x(t)+U(r)+h(t-1)\times W(r)) \ldots\ldots (12)$$

Update gate,

$$u(t)=\sigma(x(t)+U(u)+h(t-1)\times W(u)) \ldots\ldots (13)$$

Where, r(t) = reset gate, u(t) = update gate, U(r), U(u), W(r), W(u) = weight matrices, t= time, h(t-1) = hidden state.

Dropout Layer: When testing accuracy is worse than the training accuracy, then this is known as the problem of over fitting. This problem is resolved using the dropout method. Dropout is a method used to prevent over fitting problems by dropping out the neurons in every hidden layer when training the model [56].

It also reduces the sensitivity of each neuron. These neurons are ignored and assigned to zero value during the training stage, as shown in **Fig. 6**. The dropout rate used in this study is 0.05.

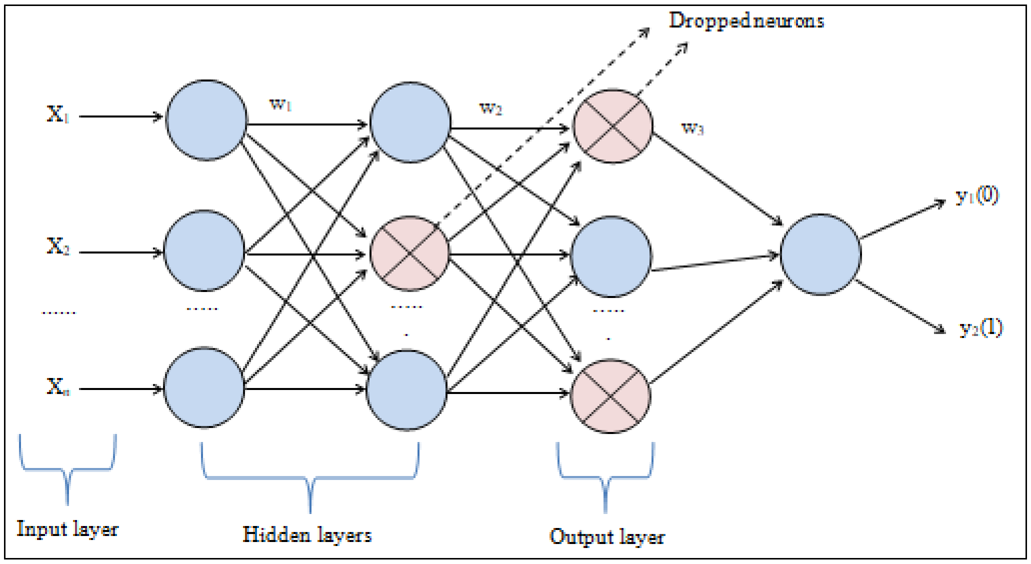


FIG. 6: AN EXAMPLE OF DROPPED OUT NEURONS IN ALL DEEP LEARNING MODELS

Performance Measures: The different performance measures calculated during the testing of the models are discussed below:

Confusion matrix: It is a two-dimensional table consisting of the actual and predicted category of classes as true positive (TP), true negative (TN), false positive (FP) and false negative (FN) [24]. From this, other performance measures such as Log-Loss, AUC, precision, recall, specificity, sensitivity and accuracy are derived.

TABLE 5: CONFUSION MATRIX

		Actual Value	
Predicted Value		1	0
	1	TP	FP
	0	FN	TN

Sensitivity (Se): It is also known as recall or true positive rate (TPR). It is the performance metrics used to evaluate the model's ability to predict truly positive class [51].

$$Se = TP / (TP+FN) \ldots\ldots\ldots (14)$$

Specificity (Sp): Similarly, it is used to evaluate the model's ability to predict truly negative class [50].

$$Sp = TN / (TN+FP) \ldots\ldots\ldots (15)$$

Precision (+P): It is also known as precision. The proportion of truly positive and overall positive is known as positive predictive value [51].

$$+P = TP / (TP+FP) \ldots\ldots\ldots (16)$$

F1-score: It is the weighted average or function of precision (positive predictive value) and recall (sensitivity). Also it is a harmonic mean of precision (positive predictive value) and recall. The value of F1 is near to 1 is the best value [51].

$$\text{F1-score} = 2^* (\text{Precision}^*\text{Recall}) / (\text{Precision}+\text{Recall}) \ldots\ldots (17)$$

Accuracy: It is the ratio of correctly predicted instances to overall testing instances [51].

Acc = (TP+TN) / (TP+FP+FN+TN)…….18)

Area under Curve (AUC): It is the degree of the capacity of a classifier to differentiate among classes which is used as the representation of the receiver operating characteristics curve [54].

ROC: It is an evaluation metric for classification models. It plots a graph between True Positive Rate (TPR) and False Positive Rate (FPR) [51].

Score 1: The minimum value between the sensitivity and the positive predictive value 7.

Score 1 = min (Se, +P).........(19)

Simulation Study, Results and Discussion:

Simulation Study: For mortality prediction of ICU patient's, four different deep neural network models, DNN, DLSTM, DBLSTM and DGRU are simulated, referring to **Fig. 4** and **5** using python.

The proposed models are implemented using Tensor flow 2.6 – an open-source library software [57], Keras – a high-level application programming interface of Tensorflow[58] and Sk-learn (Scikit-learn) – a tool for the data analysis [59].

The proposed models are trained and tested using 5 fold cross validation. The models are made up of one input layer, four hidden layers with 60,50,50,40 number of neurons and one output layer. To train the deep learning models using Physionet challenge 2012 and WiDS datathon 2020 dataset, a value of forty features and twenty two features respectively are given as input to the models. It forwards the weighted input to the next hidden layers, and finally, the output is obtained at the output layer. The obtained output is then compared with the desired value to generate an error. The back propagation algorithm is used to minimize the error and update the weights. Weight regularizer (L2) is used for weight updation. The change in weights is then calculated for every given input. This process completes one iteration.

The same process is repeated for 90 iterations for all deep learning models. The learning rate is fixed at 0.001, as it gives better training performance after several trails. ReLu activation function is used in all hidden layers and 'sigmoid' activation function is used in the output layer. Kernel and bias initializer is set to default "glorot uniform" in the dense layers.

"Adam" optimizer is used with default parameters (learning rate = 0.01) to compile the models. Loss function, 'binary crossentropy' is used. In the networks, layer dropout is set to 0.05. It means 5% of neurons are dropped out from the hidden layers, which helps improving the training performance and overcoming the overfitting problem.

The same process is done for all the models only the working of model is different. Another hyperparameter return sequences = "True" is tuned for all models (DLSTM, DBLSTM and DGRU) except DNN. During cross validation of the models, using the hyperparameters as given in the **Table 6** and **7**, model performance measures are evaluated for both the datasets and exhibited in terms of mean and variance in **Table 8-11.**

TABLE 6: HYPERPARAMETERS TUNED IN DNN AND DLSTM

Hyperparameters	Values used	
	DNN	DLSTM
Quantity of layers	04	04
Quantity of Neurons	60,50,50,40	60,50,50,40
Kernel_initializer	glorot_uniform	-
Kernal_regularize	0.01	-
Learning Rate	0.01	0.01
Momentum Rate	0.9	0.9
Adaptive Learning Rate Method	SGD	SGD
Activation function	ReLU	ReLU
Optimizer	adam	adam
Loss_function	binary_crossentropy	binary_crossentropy
Epochs	90	90
Batch_size	300(for Physionet data)/500 (for WiDS data)	300/500
Dropout Rate	0.05	0.05
Weight regularizer	L2	L2
Return sequences	-	True

TABLE 7: HYPERPARAMETERS TUNED IN DBLSTM AND DGRU

Hyper parameters	Values used	
	DBLSTM	DGRU
Quantity of layers	04	04
Quantity of Neurons	60,50,50,40	60,50,50,40
Kernel_initializer	-	-
Kernal_regularize	-	-
Learning Rate	0.01	0.01
Momentum Rate	0.9	0.9
Adaptive Learning Rate Method	SGD	SGD
Activation function	ReLU	ReLU
Optimizer	adam	adam
Loss_function	binary_crossentropy	binary_crossentropy
Epochs	90	90
Batch_size	300/500	300/500
Dropout Rate	0.05	0.05
Weight regularizer	L2	L2
Return sequences	True	True

RESULTS AND DISCUSSION: The performance measures such as sensitivity, specificity, precision, F1_score, AUC and accuracy of DNN, DLSTM, DBLSTM and DGRU are obtained for Physionet challenge 2012 and presented in the **Table 8** and **9.**

Results using Physionet Challenge 2012 Dataset:

TABLE 8: PERFORMANCE MEASURES OF DNN AND LSTM WITH THREE DIFFERENT BALANCING TECHNIQUES (SMOTE, CSL AND GAN)

Evaluation Criteria	DNN			DLSTM		
	Testing results with 5 fold cross validation			Testing results with 5 fold cross validation		
	SMOTE+DNN (mean±variance)	CSL+DNN (mean±variance)	GAN+DNN (mean±variance)	SMOTE+DLSTM (mean±variance)	CSL+DLSTM (mean±variance)	GAN+DLSTM (mean±variance)
Sensitivity (Se)	0.7790+0.0198	0.7685±0.0490	0.7675±0.0402	0.8280±0.0666	0.7577±0.0047	0.7922±0.0124
Specificity (Sp)	0.7435+0.0137	0.7510±0.0254	0.7412±0.0111	0.7765±0.0184	0.7325±0.0135	0.7877±0.0231
Precision(+P)	0.7524+0.0094	0.7394±0.0136	0.7515±0.0025	0.7868±0.0197	0.7833±0.0142	0.7823±0.0112
F1-Score	0.7654+0.0110	0.7402±0.0141	0.7622±0.0126	0.8060±0.0407	0.7434±0.0074	0.7727±0.0311
Accuracy (Acc)	0.7613+0.0096	0.7536±0.0166	0.7516±0.0135	0.8023± 0.0339	0.7627±0.0056	0.7913±0.0211
AUC	0.7613+0.0096	0.7598±0.0161	0.7514±0.0082	0.8023±0.0335	0.7579±0.0020	0.7913±0.0210
Score1 = min(Se, +P)	0.7524+0.0094	0.7394±0.0136	0.7515±0.0025	0.7868±0.0197	0.7833±0.0142	0.7823±0.0112
Run Time (in second)	171.1885	116.6315	266.7523	24608.2676	22125.2896	22405.2122

TABLE 9: PERFORMANCE MEASURES OF BLSTM AND GRU WITH THREE DIFFERENT BALANCING TECHNIQUES (SMOTE, CSL AND GAN)

Evaluation Criteria	DBLSTM			DGRU		
	Testing results with 5 fold cross validation			Testing results with 5 fold cross validation		
	SMOTE+DBLSTM (mean±variance)	CSL+DBLSTM (mean±variance)	GAN+DBLSTM (mean±variance)	SMOTE+DGRU (mean±variance)	CSL+DGRU (mean±variance)	GAN+DGRU (mean±variance)
Sensitivity (Se)	0.8089±0.0807	0.7608 ± 0.0078	0.7292± 0.0047	0.7759±0.1196	0.763 ± 0.0021	0.7872±0.0881
Specificity (Sp)	0.7877±0.4812	0.7680± 0.0130	0.7804± 0.0077	0.8404±0.0283	0.759 ± 0.0041	0.6881± 0.2420
Precision (+P)	0.7939±0.0284	0.7552 ± 0.0090	0.6775± 0.0115	0.8296±0.0106	0.766 ± 0.0023	0.8852±0.0752
F1-Score	0.7986±0.0403	0.7615 ± 0.0082	0.7532± 0.0053	0.7964±0.0720	0.760 ± 0.0020	0.7701±0.1201
Accuracy (Acc)	0.7983±0.0295	0.7609±0.0000	0.7133 ± 0.0065	0.8081±0.0476	0.763 ± 0.0014	0.8121± 0.0472

AUC	**0.7983±0.0319**	0.7609±0.0000	0.7289±0.0000	**0.8081±0.0455**	0.7621±0.0100	0.7871±0.0020
Score1 = min(Se, +P)	0.7939±0.0284	0.7552 ± 0.0090	0.6775± 0.0115	0.7759±0.1196	0.7630±0.0021	0.7870±0.0882
Run Time (in second)	32418.9047	22188.04459	25110.4225	16134.8110	11470.03846	13445.9614

From the above tables, it is exhibited that SMOTE +DNN, SMOTE+DLSTM, SMOTE+BDLSTM and SMOTE+DGRU have obtained better AUC results as compared to other deep learning models simulated. The performances are measured using same number of layers, neurons and epochs with 5 fold cross validation and runs 10 times independently. Mean and variance of all performance measure values are taken as final value after 10 independent runs. In DNN using physionet challenge 2012 dataset, SMOTE+DNN achieves AUC, F1-score and accuracy of 0.7613±0.0096 (mean± variance), 0.7654±0.0110 and 0.7613±0.0096 respectively in 90epochs using four hidden layers with 60,50,50,40 neurons respectively and performs better in comparison with other models (*i.e.* CSL+DNN, GAN+DNN). Further, the SMOTE+DLSTM results AUC, F1-score and accuracy of 0.8023±0.0335, 0.8060±0.0407 and 0.8023±0.0339 respectively in same epochs using same hidden layers and performs better in comparison with models (*i.e.* CSL+DLSTM, GAN+DLSTM). Similarly, SMOTE+DBLSTM gives AUC, F1-score and accuracy of 0.7983±0.0319, 0.7986±0.0403 and 0.7983±0.0295 respectively in same epochs in comparison with models (*i.e.* CSL+DBLSTM, GAN+DBLSTM). In the same way, SMOTE+ DGRU has resulted the best AUC, F1-score and accuracy values of 0.8081±0.0455, 0.7964±0.0720 and 0.8081±0.0476 respectively. Also, after overall analysis of results on physionet dataset, it is observed that SMOTE+DGRU model outperforms other deep learning models. The ROC curves for all the models that perform best are graphically presented in **Fig. 6** and **7.**

Results using WiDSdatathon 2020:

TABLE 10: PERFORMANCE MEASURES OF DNN AND LSTM MODELS WITH THREE DIFFERENT BALANCING TECHNIQUES (SMOTE, CSL AND GAN)

Evaluation Criteria	DNN			DLSTM		
	Testing results with 5 fold cross validation			Testing results with 5 fold cross validation		
	SMOTE+DNN (mean±variance)	CSL+DNN (mean±variance)	GAN+DNN (mean±variance)	SMOTE+DLSTM (mean±variance)	CSL+DLSTM (mean±variance)	GAN+DLSTM (mean±variance)
Sensitivity (Se)	0.7600±0.0153	0.7795±0.0004	0.7425±0.0111	0.8046±0.0724	0.7826±0.0624	0.7924±0.0232
Specificity (Sp)	0.7292±0.0345	0.7380±0.0117	0.7212±0.0124	0.7516± 0.0757	0.7641±0.0421	0.7624±0.0252
Precision (+P)	0.7382±0.0212	0.7444±0.0028	0.7311±0.0215	0.7694±0.0398	0.7656±0.0411	0.7700±0.0032
F1-Score	0.7486±0.0084	0.7505±0.0025	0.7352±0.0021	0.7829±0.0171	0.7563±0.0141	0.7752±0.0012
Accuracy (Acc)	0.7446±0.0126	0.7588±0.0057	0.7433±0.0122	0.7781±0.0118	0.7639±0.0211	0.7821±0.0023
AUC	0.7446±0.0126	0.7587±0.0057	0.7533±0.0122	0.7781±0.0118	0.7639±0.0211	0.7821±0.0023
Run Time (in second)	757.9428	638.5263	622.4342	34981.0849	32521.5234	31125.2426

TABLE 11: PERFORMANCE MEASURES OF DBLSTM AND DGRU MODELS WITH THREE DIFFERENT BALANCING TECHNIQUES (SMOTE, CSL AND GAN)

Evaluation Criteria	DBLSTM			DGRU		
	Testing results with 5 fold cross validation			Testing results with 5 fold cross validation		
	SMOTE+DBLSTM (mean±variance)	CSL+DBLSTM (mean±variance)	GAN+DBLSTM (mean±variance)	SMOTE+DGRU (mean±variance)	CSL+DGRU (mean±variance)	GAN+DGRU (mean±variance)
Sensitivity (Se)	0.8873± 0.0439	0.7958±0.0152	0.8192±0.0411	0.8786±0.0528	0.7882±0.0312	0.8268±0.0142
Specificity (Sp)	0.8575± 0.0353	0.7825±0.0210	0.7964±0.0312	0.8590±0.0235	0.7721±0.0212	0.8022±0.0031
Precision (+P)	0.8629± 0.0266	0.7846±0.0200	0.8061±0.0221	0.8625±0.0142	0.7613±0.0112	0.8156±0.0312

F1-Score	0.8739± 0.0215	0.7796±0.0154	0.7914±0.0310	0.8693±0.0239	0.7711±0.0231	0.8042±0.0021
Accuracy (Acc)	0.8724± 0.0199	0.7964±0.0162	0.8091±0.0211	0.8689±0.0190	0.7865±0.0221	0.8278±0.0131
AUC	0.8724± 0.0199	0.7964±0.0162	0.8091±0.0211	0.8689±0.0190	0.7865±0.0221	0.8278±0.0131
Run Time (in second)	131687.2815	125631.2254	122314.6623	41645.9075	39625.8170	35361.4526

Similarly, from the above tables, it is demonstrated that CSL+DNN, GAN+DLSTM, SMOTE+ DBLSTM and SMOTE+DGRU have yield better AUC, F1-score and accuracy as compared to other deep learning models in case of WiDsdatathon. The performances are measured using same number of layers, neurons and epochs.

In case of DNN, CSL+DNN achieves AUC, F1-score and accuracy of 0.7587±0.0057 (mean±variance), 0.7505±0.0025 and 0.7588± 0.0057 respectively and performs better in comparison with other models (*i.e.* SMOTE+DNN, GAN+DNN). Further, the GAN+DLSTM results AUC, F1-score and accuracy of 0.7821±0.0023, 0.7752±0.0012 and 0.7821±0.0023 respectively in comparison to other models (*i.e.* SMOTE+DLSTM, CSL+DLSTM). Similarly, SMOTE+DBLSTM achieves better AUC of 0.8724± 0.0199, 0.839±0.0215 and 0.8724±0.0199 respectively in comparison with other models (*i.e.* CSL+DBLSTM, GAN+ DBLSTM).

In the same way, SMOTE+DGRU has obtained the AUC, F1-score and accuracy values of 0.8689±0.0190, 0.8693±0.0239 and 0.8689±0.0190 respectively. Also, after overall analysis of results on WiDSdatathon 2020 dataset, it is observed that SMOTE+DBLSTM model outperforms other deep learning models. The ROC curves for all the proposed deep learning models are graphically presented in **Fig. 7** and **8.**

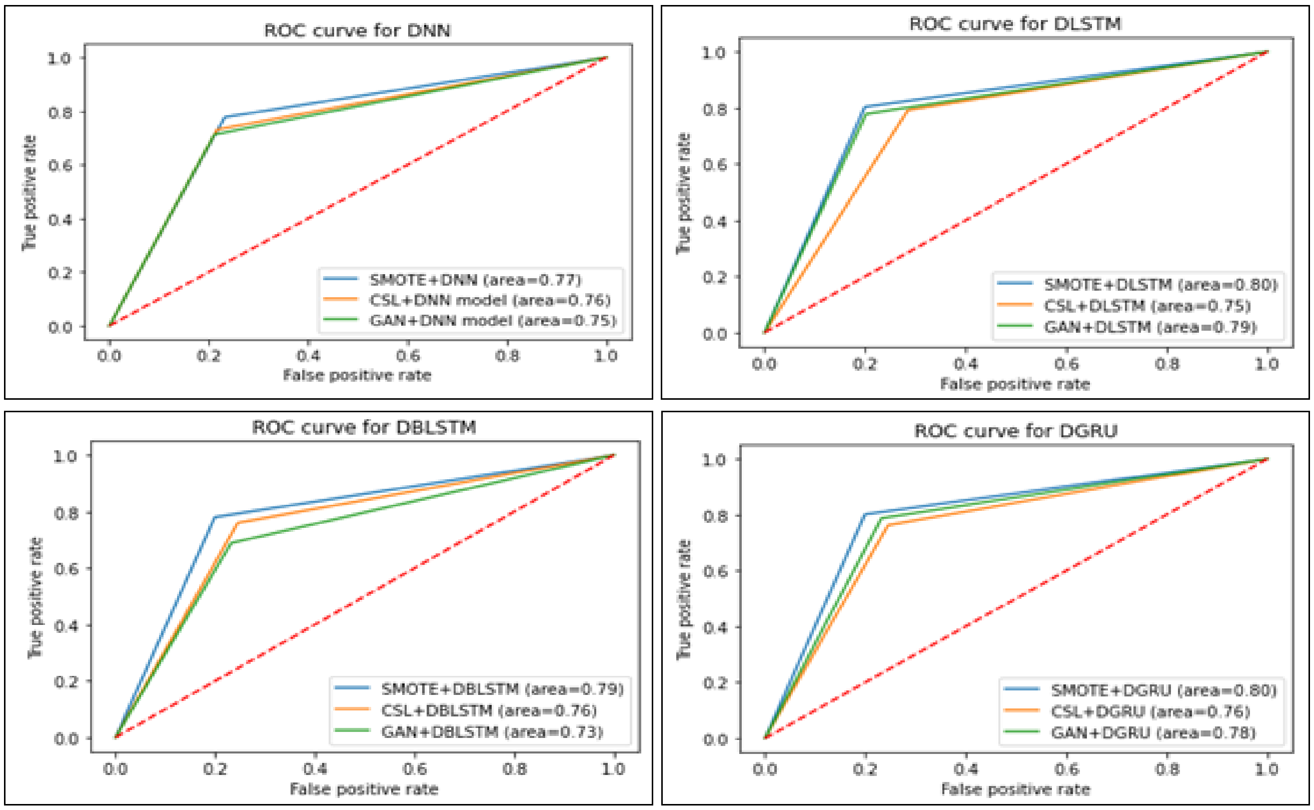


FIG. 7: ROC CURVES FOR DEEP LEARNING MODELS (DNN, DLSTM, DBLSTM AND DGRU) WITH THREE BALANCING TECHNIQUES (SMOTE, CSL AND GAN) AND PHYSIONET CHALLENGE 2012 DATASET

From the above ROC figures, it is observed that (SMOTE+DNN), (SMOTE+DLSTM), (SMOTE+ DBLSTM) and (SMOTE+DGRU) perform better as compared to other models. The (SMOTE+DLSTM) and (SMOTE+DGRU), both perform best with AUC of 80% in case of physionet data.

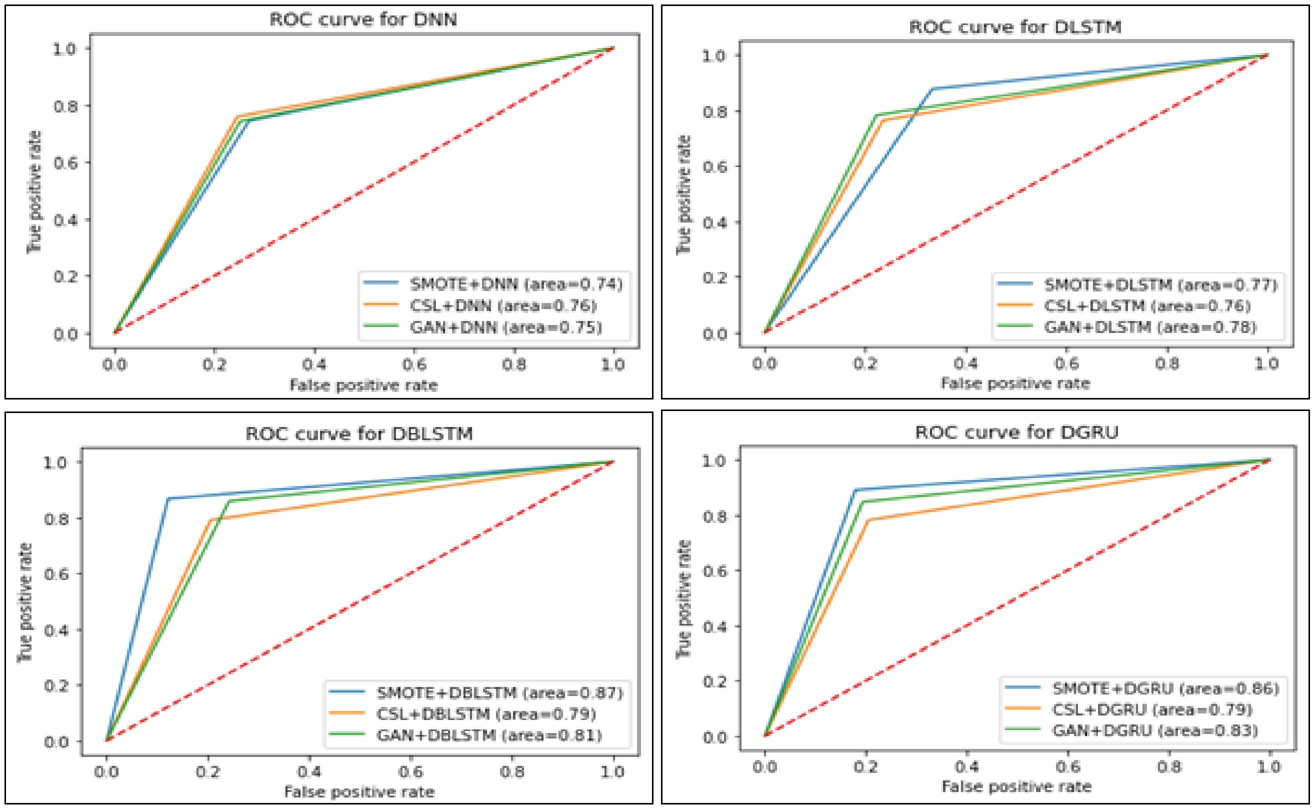


FIG. 8: ROC CURVES FOR DEEP LEARNING MODELS (DNN, DLSTM, DBLSTM AND DGRU) WITH THREE BALANCING TECHNIQUES (SMOTE, CSL AND GAN) AND WIDSDATATHON 2020 DATASET

From the above figures, it is clear that (CSL+DNN), (GAN+DLSTM), (SMOTE+DBLSTM) and (SMOTE+DGRU) perform better as compared to another models. In overall SMOTE+DBLSTM is performing best with AUC of 87% in case of WiDSdatathon.

Comparison with the Previous Existing Models: The results of the proposed deep learning models in terms of AUC are compared with some previous existing results of the physionet challenge 2012 and given in **Table 11**.

Among all the reported developed models of past, [15] has obtained the best AUC score of 86% and [22] has claimed the best score 1 of 81%. The proposed models enhances the performance of AUC and obtained superior results.

TABLE 12: COMPARISON OF AUC VALUES OBTAINED BY PROPOSED MODELS WITH EXISTING MODELS FOR PHYSIONET DATA

Authors	Models	Set A only
		AUC (in %)
Monteiro *et al.*[4]	Random Forest	-
Ma *et al.*[21]	One class JITL	85.10
Liu *et al.* [3]	MCSPCA+CPSO+SVM	77.18
Zhenging *et al.*[23]	GRU-D	83.70
Ding *et al.*[24]	JITL-ELM	85.68
Bhattacharya *et al.* [25]	CHISQ-NEW	86.70
Karmakar *et al.*[26]	Multi-Task Transfer Learning	-
Xu *et al.*[27]	SVM	-
Chen *et al.* [28]	Multilayer Neural Network	-
Johnson *et al.* [29]	Regularized Logistic Regression	84.80
Proposed methodology		(mean±variance)
Using Physionet 2012 dataset (A,B,C sets)	SMOTE+DGRU	80.81

CONCLUSION: Physionet challenge 2012 and WiDSdatathon2020 datasets have provided challenge for developing new machine learning models to predict mortality of ICU patients. In this study, four deep learning models DNN, DLSTM, DBLSTM and DGRU have been proposed for predicting ICU patient's mortality in the hospital. The simulation is done using python programming with different machine learning and deep learning packages. Tensor flow and Keras are used for the deep learning models. In this study, simulation is done using (set A, B and C together) Physionet and WiDS datasets. The simulation study depicts that the proposed models enhance the performances and achieve good results. Out of the four proposed models, SMOTE+DGRU performs bestin comparison with other models in Physionet challenge 2012with AUC, F1-score and accuracy of 0.8081, 0.7964 and 0.8081 respectively. The SMOTE+DBLSTM performs best in comparison with other models in WiDSdatathon 2020 with AUC, F1-score and accuracy of 0.8724, 0.839 and 0.8724 respectively. Still, it is a challenging problem because of imbalance and enormous amounts of missing data in both the datasets. In the future, authors will focus on solving the problem using other deep learning and machine learning techniques for the same, and will try to improve the AUC. Also swarm intelligence algorithms can be used to optimize parameters of proposed deep learning models. An IOT based system can also be designed for this purpose in future work.

ACKNOWLEDGEMENT: The authors would like to thank the department of CSIT, Guru Ghasidas Vishwavidyalaya (Central University), Bilaspur for providing the facilities and support for doing this research.

CONFLICTS OF INTEREST: The authors declared no conflicts of interest.

REFERENCES:

1. XuJ, Yuanjian Z, Mahmood A, Peng Z, Yu Li and Khatoon S: Data mining on ICU mortality prediction using early temporal data. International Journal of Information Technology and Decision Making 2017; 16(01): 117-159.
2. Pronovost PJ, Angus DC, Dorman T, Robinson KA, Dremsizov TT and Young TL: Physician staffing patterns and clinical outcomes in critically ill patients: A systematic review. JAMA 2002; 288(17): 2151–2162.
3. Liu J, Chen X, Fang L and Li JX: Mortality prediction based on imbalanced high-dimensional ICU big data. Computers in industry. Elsevier 2018; 98: 218-225.
4. Monterio F, Meloni F, Baranauskas JA and Macedo AA: Prediction of mortality in Intensive Care Units: a multivariate feature selection. Journal of Biomedical Informatics Elsevier 2020; 107: 103456.
5. Robert R, Coudroy R, Ragot S, Lesieur O, Isabelle R, Souday V, Desachy A, Jean-Paul G, Michel H, Hamrouni M and Jean R: Influence of ICU bed availability on ICU admission decisions. Annals Intensive Care 2015; 5: 55.
6. Awad A, Bader-El-Den M, McNicholas J and Briggs J: Early hospital mortality prediction of intensive care unit patients using an ensemble learning approach. CDATA. International Journal of Medical Informatics 2017; 108: 185-195.
7. Goldberger A, Amaral L, Glass L, Hausdorff J, Ivanov PC, Mark R and Stanley HE: PhysioBank, Physio Toolkit, and PhysioNet : Components of a new research resource for complex physiologic signals 2000; 101: 23.
8. Le Gall JR, Lemeshow S and Saulnier F: A new simplified acute physiology score (saps II) based on a european/north American multicenter study. JAMA 1993; 270(24): 2957-2963.
9. Hosmer D and Lemeshow S: Applied Logistic Regression. A Wiley Interscience publication, third edition 2013.
10. Kim S, Kim W and Park RW: A Comparison of Intensive Care Unit Mortality Prediction Models through the Use of Data Mining Techniques. The Korean society of medical informatics, Healthcare Informatics Research 2011; 17(4): 232-243.
11. Caicedo W and Gutierrez J: ISeeU: Visually interpretable deep learning for mortality prediction inside the ICU. Journal of Biomedical Informatics, Elsevier 2019; 98: 103269.
12. Mahapatra S, Gupta VRR, Sahu SS and Panda G: Deep neural network and extreme gradient boosting based hybrid classifier for improved prediction of Protein-Protein interaction. IEEE/ACM Transactions on Computational Biology and Bioinformatics 2021; 1-1.
13. Majhi B, Naidu D and Mishra AP: Improved prediction of daily pan evaporation using Deep-LSTM model. Neural Computand Applic 2020; 32:7823–7838.
14. Baker S, Xiang W and Atkinson I: Continuous and automatic mortality risk prediction using vital signs in the intensive care unit: a hybrid neural network approach. Scientific Reports 2020; 10: 21282.
15. Dey R and Salem FM: Gate-variants of Gated Recurrent Unit (GRU) neural networks. 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS) 2017; 1597-1600.
16. Du W, Cote D and Liu Y: SAITS: Self-Attention-Based Imputation for Time Series. arXiv 2022; 2: 1-22.
17. Darke P, Missier P and Bacardit J: Benchmark time series data sets for PyTorch – the torchtime package. arXiv 2022; 2: 1-22.
18. Shukla SN and Marlin BM: Multi-Time Attention Networks for Irregularly Sampled Time Series. International Conference on Learning Representations (ICLR) 2021. arXiv 2021; 2: 1-22.
19. Toth C, Bonnier P and Oberhauser H: SEQ2TENS: An efficient representation of sequences by Low-Rank Tensor Projections. International Conference on Learning Representations (ICLR) 2021.arXiv 2021; 2: 1-22.
20. Rubanova Y, Chen RTQ and Duvenaud D: Latent ODEs for Irregularly-Sampled Time Series. arXiv 2019; 2: 1-22.
21. Ma X, Si Y, Wang Z and Wang Y: Length of stay prediction for ICU patients using individualized single classification algorithm. Computer Methods and Programs in Biomedicine 2020; 186: 105224.
22. Awad A, Bader-EI-Den M, McNicholas J, Briggs J and EI-Sonbaty Y: Predicting hospital mortality for intensive care unit patients: Time-series analysis. Health Informatics Journal 2020; 26(2): 1043-1059.

23. Che Z, Purushotham S, Kyunghyun C, Sontag D and Yan L: Recurrent neural networks for multivariate time series with missing values. Scientific Reports 2018; 8(1): 6085.
24. Ding Y, Wang Y and Zhou D: Mortality prediction for ICU patients combining just-in-time learning and extreme learning machine, Neurocomputing, Elsevier 2018; 281: 12-19.
25. Bhattacharya S, Rajan V and Shrivastava H: ICU mortality prediction: A classification algorithm for imbalanced datasets. In proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) 2017; 1288-1294.
26. Karmakar C, Saha B, Palaniswami M and Venkatesh S: Multi-task transfer learning for in-hospital-death prediction of ICU patients. In proc. of 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2016; 3321-3324.
27. Xu J, Li D, Zhang Y, Djulovic A, Li Y and ZengY: CinC Challenge: Cluster analysis of multi-granular time-series data for mortality rate prediction. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 497-500.
28. Chen Y and Hui Y: Heterogenous postsurgical data analytics for predictive modelling of mortality risks in intensive care units. In Proc. of 36th Annual International Conference of IEEE Engineering in Medicine and Biology Society (EMBC) 2014; 4310-4314.
29. Johnson AE, Kramer AA and Clifford GD: Data preprocessing and mortality prediction: the physionet/CinC challenge revisited. In proc. of IEEE conf. on Computing in Cardiology (CinC) 2021; 39: 157-160.
30. Silva I, Moody G, Scott DJ, Celi LA and Mark RG: Predicting in-hospital mortality of ICU patient: The Physionet/Computing in Cardiology Challenge 2012. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 245-248.
31. Johnson AE, Dunkley N, Mayaud L, Tsanas A, Kramer AA and Clifford GD: Patient specific predictions in the intensive care unit using a Bayesian Ensemble. In proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 249-252.
32. Lee CH, Arzeno NM, Ho JC, Vikalo H and Ghosh J: An imputation-enhanced algorithm for ICU mortality prediction. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 253-256.
33. Citi L and Barbieri R: PhysioNet 2012 Challenge: Predicting mortality of ICU Patients using a cascaded SVM-GLM Paradigm. In proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 257-260.
34. Xia H, Daley BJ, Petrie A and Zhao X: A neural network model for mortality prediction in ICU. In Proc. of IEEE conf. on Computing in Cardiolog (CinC) 2012; 39: 261-64.
35. McMillan S, Chia C, Esbroeck AV, Rubinfeld I and Syed Z: ICU mortality prediction using time series motifs. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 265-268.
36. Vairavan S, Eshelman L, Haider S, Flower A and Seiver A: Prediction of mortality in an intensive care unit using logistic regression and a hidden Markov model. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 393-396.
37. Yi C, Sun Y and Tian Y: Cin C Challenge: Predicting in-hospital mortality in the intensive care unit by analyzing histograms of medical variables under cascaded Adaboost model. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 397-400.
38. Krajnak M, Xue J, Kaiser W and Balloni W: Combining machine learning and clinical rules to build an algorithm for predicting ICU mortality risk. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 401-404.
39. Severeyn E, Altuve M, Ng F, Lollett C and Wong S: Towards the prediction of mortality in intensive care units patients: A simple correspondence analysis approach. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 469-472.
40. Macas M, Kuzilek J, Odstrcilik T and Huptych M: Linear Bayes classification for mortality prediction. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 473-476.
41. Marco LYD, Bojarnejad M, King ST, Duan W, Maria CD, Zheng D, Murray A and Langley P: Robust prediction of patient mortality from 48 Hour Intensive Care Unit data. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 477-480.
42. Bosnjak A and Montilla G: Predicting mortality of ICU Patients using statistics of physiological variables and support vector machines. In proc. of IEEE conference on Computing in Cardiology 2012; 39: 481-484.
43. Pollard TJ, Harra L, Williams D, Harris S, Martinez D and Fong K: 2012 PhysioNet Challenge : An artificial neural network to predict mortality in ICU patients and application of solar physics analysis methods. In Proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 485-488.
44. Hamilton SL and Hamilton JR: Predicting in-hospital-death and mortality percentage using logistic regression. In proc. of IEEE conf. on Computing in Cardiology (CinC) 2012; 39: 489-492.
45. Bera D and Nayak MM: Mortality risk for ICU patients using logistic regression, in proceeding of IEEE Conf. On Computing in Cardiology (CinC) 2012; 39: 493-496.
46. Cohen S, Dagan N, Cohen-Inger N, Ofer D and Rokach L: ICU Survival Prediction Incorporating Test-Time Augmentation to Improve the Accuracy of Ensemble-Based Models in IEEE Access 2022; 9: 91584-91592.
47. Zhang S: Nearest neighbor selection for iteratively k-NN imputation. The Journal of Systems and Software 2012; 85: 2541–2552.
48. Chawla NV, Bowyer KW, Hall LO and Kegelmeyer WP: SMOTE: Synthetic Minority Over-Sampling Technique. Journal of Artificial Intelligence Research 2002; 16: 321-357.
49. Mienye ID and Sun Y: Performance analysis of cost-sensitive learning methods with application to imbalanced medical data. Informatics in Medicine Unlocked 2021; 25: 100690.
50. Nazari E and Branco P: On oversampling via generative adversarial networks under different data difficulty factors. Proceedings of the Third International Workshop on Learning with Imbalanced Domains: Theory and Applications, in Proceedings of Machine Learning Research 2021; 154: 76-89.
51. Han J, Pei J and Kamber M: Data mining: Concepts and Techniques. Morgan Kaufmann Publishers, Elsevier, Second Edition 2011.
52. Majhi B, Kashyap A and Majhi R: Mortality prediction of ICU Patients using machine learning techniques, in: S. Dashetal.(Eds.), Biomedical Data Mining for Information Retrieval: Methodologies, Techniques and Applications, Wiley 2021; 1-19.
53. Sonak A and Patankar RA: A survey on methods to handle imbalance dataset, International Journal of Computer Science and Mobile Computing 2015; 4-11: 338-343.

54. Kotsiantis S and Pintelas PE: Handling Imbalanced datasets: A review, GESTS International Transactions on Computer Science and Engineering 2006; 30.
55. Vanilla GAN (GANs in computer vision: Introduction to generative learning)". theaisummer.com. AI Summer. April 10, 2020. Archived from the original on June 3, 2020. Retrieved September 20, 2020.
56. Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R: Dropout: a simple way to prevent neural networks from over fitting. The Journal of Machine Learning Research 2014; 15(1): 1929–1958.
57. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J and Zheng X: Tensorflow: a system for large-scale machine learning. In-proceedings of 12thUSENIX conference on operating systems design and Implementation 2016; 272-283.
58. Chollet F: Keras: Deep learning library for Theano and Tensorflow. GitHub Repository 2015; 1-21.
59. Pedregosa F, Weiss R and Brucher M: Scikit-learn: machine learning in python. J Mach Learn Res 2011; 12: 2825-2830.

How to cite this article:
Majhi B and Kashyap A: Development and assessment of different balancing techniques and deep learning based early mortality prediction models for ICU imbalance data. Int J Pharm Sci & Res 2023; 14(8): 4170-92. doi: 10.13040/IJPSR.0975-8232.14(8).4170-92.