Rohit Verma*, Saumil Maheshwari and Anupam Shukla

Feature engineering combined with 1-D convolutional neural network for improved mortality prediction

https://doi.org/10.1515/bams-2020-0056
Received September 24, 2020; accepted November 2, 2020; published online November 23, 2020

Abstract

Objectives: The appropriate care for patients admitted in Intensive care units (ICUs) is becoming increasingly prominent, thus recognizing the use of machine learning models. The real-time prediction of mortality of patients admitted in ICU has the potential for providing the physician with the interpretable results. With the growing crisis including soaring cost, unsafe care, misdirected care, fragmented care, chronic diseases and evolution of epidemic diseases in the domain of healthcare demands the application of automated and real-time data processing for assuring the improved quality of life. The intensive care units (ICUs) are responsible for generating a wealth of useful data in the form of Electronic Health Record (EHR). This data allows for the development of a prediction tool with perfect knowledge backing.

Method: We aimed to build the mortality prediction model on 2012 Physionet Challenge mortality prediction database of 4,000 patients admitted in ICU. The challenges in the dataset, such as high dimensionality, imbalanced distribution and missing values, were tackled with analytical methods and tools via feature engineering and new variable construction. The objective of the research is to utilize the relations among the clinical variables and construct new variables which would establish the effectiveness of 1-Dimensional Convolutional Neural Network (1-D CNN) with constructed features.

Results: Its performance with the traditional machine learning algorithms like XGBoost classifier, Light Gradient Boosting Machine (LGBM) classifier, Support Vector Machine (SVM), Decision Tree (DT), K-Neighbours Classifier (K-NN), and Random Forest Classifier (RF) and recurrent models like Long Short-Term Memory (LSTM) and LSTM-attention is compared for Area Under Curve (AUC). The investigation reveals the best AUC of 0.848 using 1-D CNN model.

Conclusion: The relationship between the various features were recognized. Also, constructed new features using existing ones. Multiple models were tested and compared on different metrics.

Keywords: 1-D CNN; feature engineering; LSTM; mortality prediction; random forest; XGBoost.

Introduction and related work

Quality of life is the right of every citizen, and health is a crucial determinant of quality of life. Physicians all around the globe are working hard for updating their knowledge with the latest research. They are well trained and experienced too for providing treatments to patients specifically to patients in Intensive Care Units (ICU). ICUs are responsible for constant monitoring of such patients using the medical devices under the supervision of a specialist.

Consequently, we have the electronic health record of patients with baseline characteristics of patients. As ICU accounts for the majority of healthcare expenses for the patients anguish of critical health issues, evaluating the severity of illness becomes critical [1, 2]. The prediction of mortality in patients admitted to ICU helps in evaluating the severity of illness by analysing the raw data that could have been overlooked by busy physicians. Subsequently, this will aid in providing preferable treatments and better health-care policies for patients [3].

Many severity scores since the early 1980s like Simplified Acute Physiology Score (SAPS) [4] and Acute Physiology and Chronic Health Evolution (APACHE) [5] were proposed. These have been defined for predicting the mortality of patients by utilizing patient characteristics recorded during their ICU visits. The prediction using the
scores was limited to a few biological and clinical variables selected by domain experts. Since first published, these scores have been modified time and again to improve their performance [6–9]. However, SAPS-II [6] and APACHE-II [7] scores remain the most widely applied for predicting critical illness in intensive care. Comparison between the two in single ICU has been carried out by Katsaragakis et al. [10]. In spite of many extensions proposed for severity scores, these generally remained overestimated for mortality prediction [11, 12].

Nonetheless, the development of acceptable prediction tools in the field of critical care is not an easy task because of the complexity and inconsistency associated with the data collection. Most of the prediction tool well adopted the logistic regression model and have received weighty importance by critical care professionals [13–15].

Presently, new approaches using machine learning algorithms like decision tree (DTs), artificial neural networks (ANNs), support vector machine (SVM) and an ensemble of machine learning techniques have resulted in various mortality prediction models [16–20]. These approaches have been adopted in different critical care settings or locally customized settings like considering specific disease or specific age of the population etc. Undoubtedly, therefore, machine learning approaches possess great importance for the mortality prediction model.

The ICU data associated with patients are prone to many challenges, including missing and imbalanced data, high dimensionality, irregular recordings, etc. With such challenges, it is difficult to build an effective tool for mortality prediction. Although, many of machine learning algorithms like Logistic Regression (LR), Random Forest (RF), K-Nearest Neighbour (KNN), Neural Networks (NNs) and Deep Neural Networks (DNNs) are of great concern in the field of mortality prediction. Hence, in this paper, efforts were made to assess the capabilities and application of 1-Dimensional Convolutional Neural Network (1-D CNN) model for mortality prediction. Also, recurrent models like LSTM (Long Short-Term Memory) and LSTM-AT (Long Short-Term Memory-Attention) were applied on multivariate time series data [21]. In the study by Awad et al. [22], the authors have achieved 0.83 as AUC (lower than ours) using random forest and Bayesian networks on the full attributes set. Besides this, they have begged a higher AUC but on a significantly reduced attributes set of the original set which means though it has worked on this particular dataset but might not show the same tendency to perform on other datasets of similar type.

In the present work, we focus on dealing with the dimensionality reduction and generation of new variables for adequate assessment of mortality in patients admitted in ICUs. The paper is organized as follows: Section 2 describes the data used in the study; Section 3 states the methodology adopted. Methods and implementation are discussed in Section 4. Next, Section 5 demonstrates the achieved state of the art results. Finally, Section 6 concludes the work with discussion on the present research.

Dataset description

The dataset is publicly available and it is selected from PhysioNet Challenge 2012 [23]. It consists of multivariate clinical time-series data of 8000 ICU patients. The multivariate time series record of each patient consists of 36 variables (Glucose, heart-rate, albumin, etc.) recorded in the first 48 h after the ICU admission. We have used the Training Set A because only for this subset many of the research are available (in-hospital mortality labels). This Set A contains data of 4000 ICU patients. We have chosen only the first task of the challenge and it is about predicting whether the patient dies in the hospital or not. There are 554 patients who died in ICU, and they are represented as positive samples. We cast this as a binary classification problem. Table 1 contains the statistics of the dataset.

Pre-processing and study design

The proposed approach of pre-processing for predicting mortality is achieved in two phases. During the first phase, the pre-processing of data is done. Thereafter, during the second phase new features are constructed from the available pool for efficient development of the prediction model.

Pre-processing

If there is much irrelevant or redundant information present, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can

| Statistics                  | Frequency |
|-----------------------------|-----------|
| Number of samples           | 4000      |
| Number of variables         | 41        |
| Mean of number of time stamps| 68.91     |
| Maximum of number of time stamps| 155     |
| Mean of variable missing rate| 0.8225   |
take considerable amount of processing time. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set.

As shown in Figure 1, we start off things by looking at some features and visualizing them graphically by plotting each of them against the target and other selected features to have a look of the entire range of each of them at once. Through this, we studied the variation of features against the target and their relationship with other features as well. We witnessed some general trends which supported our findings and results later. Also, it helped us to get the intuition for the significant feature engineering that we did and is justified with the research outcome.

Meanwhile, we discovered that there was one feature named “MechVent” standing for mechanical ventilation in ICU. This feature is having a constant value, i.e. zero variance. So, we removed it.

We extended our research by calculating the Pearson correlation coefficient between every feature, including the target. No feature was there in the original dataset which was very highly correlated with the target. Few are listed next which came to top relatively, GCS with −0.254, BUN with 0.223 and Bilirubin with 0.174.

For a given pair of variables $(X, Y)$, Pearson correlation Eq. (1) is calculated as:

$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}$$

where Cov is the covariance and $\sigma$ is standard deviation with respect to $X$ and $Y$. We figured that there were some pair of features which were highly correlated with each other. Some pair of features with very high mutual correlation coefficients were NIDiasABP and NIMAP with 0.884 as correlation while the others were ALT and AST with 0.858, NIMAP and NISysABP with 0.789, Lactate and ICU Type with 0.7312, Creatinine and BUN with 0.68. These values gave us a further way to explore them. Since these were highly correlated features, we wish to keep one from each of the pairs. High correlated features, in general, do not improve models (although it depends on the specifics

Figure 1: Flowchart of the pre-processing step.
of the problem like the number of variables and the degree of correlation). However, here we tested our models later and found that it would be good to remove one from each highly correlated pair.

The possible explanation for it is that highly correlated features are expected to have similar information being delivered to the model, which might cause our model to learn and increase the chances of making the model overfit.

![Figure 2: Features importance by random forest regressor before feature engineering.](image1)

![Figure 3: Features importance by XGBoost before feature engineering.](image2)
Figure 4: Features importance by random forest after feature engineering.

Figure 5: Features importance by XGBoost after feature engineering.
Also, since we are dealing with 41 features in the dataset, we must be careful with the curse of dimensionality. To remove one feature from a pair, we looked at the feature importance curves by fitting the data to the random forest classifier and XGBoost (extreme gradient boosting) classifier.

The feature importance considered here is by weight in XGBoost classifier. By weight, we meant the number of times a feature is used to split the data across all trees. The feature importance we considered using random forest classifier was mean decrease in impurity mechanism; we rather considered mutation importance. Since the permutation mechanism is much more computationally expensive than the mean decrease in impurity mechanism; we rather considered calculating gini importance and also side by side validating it with the importance curves calculated by XGBoost. Importance from each model had their say in the net importance we considered for a feature to be considered as significant. Thus, we removed AST, ICU Type and NIMAP as the one selected feature from each pair. The initial feature importance plots as a result of both the models without any feature engineering are shown in Figure 2 and Figure 3. In both figures, the most important features are GCS, Urine, and BUN.

The other more accurate method for calculating feature importance using random forest classifier is permutation importance. Since the permutation mechanism is much more computationally expensive than the mean decrease in impurity mechanism; we rather considered calculating gini importance and also side by side validating it with the importance curves calculated by XGBoost. Importance from each model had their say in the net importance we considered for a feature to be considered as significant. Thus, we removed AST, ICU Type and NIMAP as the one selected feature from each pair. The initial feature importance plots as a result of both the models without any feature engineering are shown in Figure 2 and Figure 3. In both figures, the most important features are GCS, Urine, and BUN.

### Newly proposed features

As a result, the outcome of discussed pre-processing technique, certain new features have been identified. These new features are constructed by combining the former variables. The four new features constructed from the set of available features (BUN, age, HCO₃, Lactate, pH and White blood cells) are:
- new1 = \( \sum (\text{BUN}, \text{Age}, \text{HCO}_3, \text{Lactate}, \text{pH}, \text{WBC}) \)

Figure 4 and Figure 5 shows the feature importance graph after the identified new features have been constructed. The first and the most significant feature is the cumulative sum of BUN, Age, HCO₃, Lactate, pH and WBC. Though the units are entirely different, the current focus is just on the magnitude after summation. We made several combinations using the outcomes from the correlation map and initial feature importance plots by random forests and XGBoost. Using these outcomes, we made a hypothesis to several combinations of features, most of them failed apart from the few very promising combinations which includes the formerly stated combination as well. As shown in Figure 5, it showed an F score (XGBoost importance) of 625 which is remarkable and is more than double the F score of 32 features out of 39 original features, as shown in Figure 5 (This accounts for a little higher than 82% features). The variables combined are related to each other and shown with Eq. (2).

Under normal conditions

\[
pH = 6.1 + \log_{10}\left(\frac{\text{HCO}_3}{0.03 \times \text{PaCO}_2}\right) \tag{2}
\]

where

### Table 2: BUN:Cr ratio clinical significance [24].

| BUN:Cr | Location | Mechanism |
|--------|----------|-----------|
| >20:1  | Prerenal (before the kidney) | BUN reabsorption is increased. BUN is disproportionately elevated relative to Creatinine in serum. Dehydration or hypoperfusion is suspected. |
| 10–20:1| Normal or postrenal (after the kidney) | Normal range. Can also be postrenal disease. BUN reabsorption is within normal limits. |
| <10:1  | Intrarenal (within kidney) | Renal damage causes reduced reabsorption of BUN, therefore lowering the BUN:Cr ratio. |

### Table 3: Architecture of 1D-CNN used for implementation.

| Layers                  | Output size | Kernel size |
|-------------------------|-------------|-------------|
| Convolution 1D          | (None, 40, 32) | 2*1         |
| ReLU activation         | (None, 40, 32) | 2*1         |
| Convolution 1D          | (None, 40, 32) | 2*1         |
| ReLU activation         | (None, 40, 32) | 2*1         |
| Batch normalization     | (None, 40, 32) |             |
| Max pooling             | (None, 20, 32) |             |
| Dropout                 | (None, 20, 32) |             |
| Convolution 1D          | (None, 20, 64) | 2*1         |
| ReLU activation         | (None, 20, 64) |             |
| Convolution 1D          | (None, 20, 64) | 2*1         |
| ReLU activation         | (None, 20, 64) |             |
| Batch normalization     | (None, 20, 64) |             |
| Max pooling             | (None, 10, 64) |             |
| Dropout                 | (None, 10, 64) |             |
| Flatten                 | (None, 640)   |             |
| Dense                   | (None, 100)   |             |
| Batch normalization     | (None, 100)   |             |
| Dropout                 | (None, 100)   |             |
| Dense                   | (None, 2)     |             |

Total Parameters: 79,678, Trainable Parameters: 79,286, Non-trainable Parameters: 392.
6.1 is the acid dissociation constant ($pK_a$) of carbonic acid ($H_2CO_3$) at normal body temperature.

- HCO$_3^-$ is the concentration of bicarbonate in the blood in mEq/L.

- PaCO$_2$ is the partial pressure of carbon dioxide in the arterial blood.

The next and the second proposed feature for mortality prediction in ICU is the ratio of BUN to Creatinine for each individual.

- $new2 = \frac{\text{BUN}}{\text{Creatinine}}$

As shown in Figure 5, this feature has the second-highest $F$ score (XGBoost importance) of 525. The correlation, as well as the plots, made us suspicious of the possible relationship between these features. For instance, as we said previously, that BUN (blood urea nitrogen) and Creatinine are having a correlation of 0.683 (quite high) and finally together they make a powerful feature as their ratio (BUN/Creatinine). The ratio of BUN to Creatinine is usually between 10:1 and 20:1. An increased ratio may be due to a condition that causes a decrease in the flow of blood to the kidneys, such as congestive heart failure or dehydration. It may also be seen from gastrointestinal bleeding or increased protein in the diet. The ratio may be decreased with liver disease (due to a decrease in the formation of urea) and malnutrition. This indeed provides us with a powerful feature. Table 2 provides the BUN:Cr ratio and its clinical significance.

- $new3 = \sum (\text{GCS}, \text{Albumin})$

- $new4 = \sum (\text{Urine}, \text{GCS}, \text{PaCO}_2, \text{Weight}, \text{Temp})$

The next and the third proposed feature for mortality prediction in ICU is the cumulative sum of GCS and Albumin. This is the highest-scoring feature in random forests importance, as shown in Figure 3. The next and the fourth feature proposed for mortality prediction in ICU is the cumulative sum of Urine, GCS, PaCO$_2$, Weight, and Temp. This feature has the third-highest importance in random forest’s importance curve as well as the third-highest $F$ score, as shown in Figure 4 and Figure 5. The evidence for constructing new feature 3 and feature 4 is well explained by the researchers [25–27].

Further, we deleted some features by looking at the two scenarios. So now we are dealing with a total of 40 features.

### Normalization

Then we normalized the value of each feature by subtracting the mean of each feature from its value and dividing it further by the standard deviation (Eq. (3)). The mean of new distribution becomes 0, and the variance becomes 1.

$$\text{Normalize}(X') = \frac{X - \mu}{\sigma}$$ (3)

where $X'$ is new normalized value, $X$ is the original value, $\mu$ is the mean of the feature and $\sigma$ is the standard deviation.

### Methods and implementations

#### Traditional machine learning methods

Various machine learning models have been developed in the field of healthcare. These models are generated based on the traditional machine learning methods. We used Six non-recurrent models for this clinical task of ICU mortality prediction: XGBoost classifier, light gradient boosting machine (LGBM) classifier, Support Vector Machine Classifier (SVM), Decision Tree (DT), K-Neighbours Classifier (K-NN) and Random Forest Classifier (RF).

These methods have been effectively applied for non-complex, non-sequential data set in the field of healthcare. Therefore, for each method, we have used descriptive statistics like mean, median and mode of the data for transforming 48 h of time-series knowledge as the fixed value attribute. These statistical techniques are applied before the clinical variables are normalized.

![Diagram of vanilla LSTM model](image)
Recurrent models

This research uses the vanilla LSTM (shown in Figure 6) and the LSTM with an attention mechanism are the two recurrent models that we tested. For an LSTM unit, the elementary equations are as follows:

\[
f_t = \sigma(W_{ft}x_t + U_{ft}h_{t-1} + b_f)
\]

\[
i_t = \sigma(W_{it}x_t + U_{it}h_{t-1} + b_i)
\]

\[
o_t = \sigma(W_{ot}x_t + U_{ot}h_{t-1} + b_o)
\]

\[
c_t = f_t \odot c_{t-1} + i_t \odot c_t(W_{ct}x_t + U_{ct}h_{t-1} + b_c)
\]

\[
h_t = o_t \odot \sigma(c_t)
\]

where \(\sigma\) is an activation function, \(f_t\) is a forget gate, \(o_t\) is an output gate, \(i_t\) is an input gate, \(h_t\) is a hidden state, \(c_t\) is a cell state and the operator \(\odot\) denotes the Hadamard product.

The obtained encodings are then fed as an input to decoder RNN in each step. “Hard” and “Soft” attention are the two common types of attention. Hard attention needs more sophisticated methods such as reinforcement learning to train as the model is non-differentiable, but they have an advantage of low computation cost when compared to soft attentions on large-sized inputs. The model becomes differentiable and smooth by virtue of soft attention.

1-D convolutional neural network method

Convolutional neural networks were inspired by biological processes. In these, the connectivity pattern between neurons of a network resembles the organization of the animal visual cortex. The response of an individual cortical neuron in a restricted region of the visual field is known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

In the following paragraph, we explain 1D CNN using 2D-CNN model.

In a 2D-CNN model, we have Convolutional layer after the input layer to convolve around the image. The 3-dimensional image (3rd dimension is for RGB channel) is passed to the 2D-CNN for forward propagation. The convolutions are made by 2D filter kernels to produce the 2D feature maps of the input image. These 2D filter kernels are the weights of the neurons which are optimized using the back-propagation algorithm in each epoch.

The Convolution layers are placed alternately to sub-sampling layer. Here, we are using a sub-sampling layer as a max-pooling layer. The 2D feature maps are slowly and gradually decimated or annihilated due to repeated sub-sampling as we go deeper into the network. The convolution layers are followed by a single layer to flatten the 2D feature maps into a single dimension to feed them to the MLP or fully connected layers which are placed after the convolution layers. The size after the flattening is equal to the product of two dimensions of the 2D feature maps. In between, we have used dropouts and batch normalization layers. Dropout refers to dropping out units. It is a regularization technique to reduce overfitting by preventing complex co-adaptations on training data. Batch normalization layer refers to normalizing the input layer by adjusting and scaling the activations. It also helps in reducing the amount by what the hidden unit values shift around (covariance shift). Since here we are using 1D-CNN to solve our problem, we will be using 1D arrays for both kernels as well as the feature maps, unlike 2D matrices which are used in 2D-CNN. In our experiment, we have used kernel size as 2 and the sub-sampling factor as 2. The parameters for sub-sampling as well as the kernel size, have now become scalar. However, the fully connected layers are identical to 2D-CNN and thus having the same back-propagation algorithm.

In 1D-CNNs [28], the forward propagation for the input of a neuron in the current layer \(l\) from the previous convolution layer, \(l-1\), can be expressed as,

\[
x^{l'}_k = b^{l'}_k + \sum_{i=1}^{N_{l-1}} \text{Conv1D}(w^{l'1}_{ik}, a^{l-1}_i)
\]

where \(b^{l'}_k\) is the bias and \(x^{l'}_k\) is the input of \(k\)th neuron at layer \(l\). \(a^{l-1}_i\) is the output of the \(i\)th neuron at layer \(l-1\) and \(w^{l'1}_{ik}\) is the weight (kernel) from the \(i\)th neuron at layer \(l-1\) to \(k\)th neuron at layer \(l\). The intermediate output of the neuron, \(y^{l'}_k\) from the input \(x^{l'}_k\) can be expressed as:

\[
y^{l'}_k = f(x^{l'}_k) \text{ and } a^{l'}_k = y^{l'}_k \downarrow \text{ss}
\]
where \( ss \downarrow \) represents the down-sampling operation with the factor \( ss \) and \( ol_k \) is the output of the neuron.

Next, we define the back-propagation (BP) steps for our 1D-CNN. The output layer is sourced from where the BP starts and gradually it is backpropagated to the layers before it. Let us define \( l=L \) as the output layer, \( l=1 \) as the input layer and \( N_l \) as the number of classes.

Minimization of the contributions of network parameters to the loss function in Eq. (16) is the sole aim of BP. To achieve the discussed aim, BP consists of steps computing the derivative of the loss function concerning the bias of the neuron \( k \), \( b^l_k \) and the individual weight \( w^l_{ki} \) which is connected to that neuron. To reduce their contributions and the overall error in an iterative fashion, the gradient descent method is used. The derivatives are shown in Eq. (11), where \( \Delta^l_k \) is the delta of the \( k \)th neuron at layer \( l \). It is used to update all the weights in the previous layer connected to the \( k \)th neuron and the bias of the current neuron i.e. \( k \)th.

\[
\frac{\partial E}{\partial w^l_{ik}} = \Delta^l_k y^{l+1}_k \quad \text{and} \quad \frac{\partial E}{\partial b^l_k} = \Delta^l_k
\]  

The BP executed from the first Multilayer Perceptron (MLP) to the last CNN layer is,

\[
\frac{\partial E}{\partial o^l_k} = \Delta^l_k = \frac{\sum_{i=1}^{N_l} \frac{\partial E}{\partial o^l_k}}{\sum_{i=1}^{N_l} \frac{\partial o^l_k}{\partial o^l_k}} \frac{\partial o^l_k}{\partial w^l_{k}}
\]

We can further back-propagate the error to the input \( \Delta^l_k \), once the BP has been performed from the next layer, \( l+1 \) to the layer, \( l \). Let zero-order up-sampled map be: \( w^l_k = up(o^l_k) \), then we can say:

\[
\Delta^l_k = \frac{\partial E}{\partial y^l_k} \frac{\partial y^l_k}{\partial x^l_k} = \frac{\partial E}{\partial o^l_k} \frac{\partial o^l_k}{\partial y^l_k} = \frac{\partial E}{\partial y^l_k} \frac{\partial E}{\partial y^l_k} = up(\Delta^l_k)\beta f(x^l_k)
\]  

where \( \beta = (ss)^{-1} \) since averaging \( ss \) number of elements of the intermediate output, \( y^l_k \) yielded each element of \( \Delta^l_k \). The inter BP of the delta error \((\Delta^l_k)^{-1} \) can be defined as:

\[
\Delta^l_k = \sum_{i=1}^{N_l} \text{Conv1D}(\Delta^l_k, \text{rev}(w^l_k))
\]

where Conv1D( ) does full convolution in 1D with zero padding of \( K-1 \) and rev( ) reverses the array. Following are the expressions for bias and weight sensitivities:

\[
\frac{\partial E}{\partial b^l_k} = \frac{\sum_{i=1}^{N_l} \Delta^l_k}{n} \quad \frac{\partial E}{\partial w^l_{ki}} = \text{Conv1D}(\Delta^l_k, \Delta^l_k^{-1})
\]

### Architecture of 1D CNN

The architecture shown in Table 4 is the best tuned output received after numerous runs with different sets of hyperparameters. A brief comparison among few of the various sets of hyperparameters tested along with their respective AUC is shown in Table 3. An input layer of input shape as \((40, 1)\) is used first. Next, one convolution layer with 32 filters and each of kernel size 3 is added followed by an activation function of ReLU (Rectified Linear Unit). Again, a convolution layer similar to as prescribed above is concatenated to the above system followed by another ReLU unit. Then a batch normalization layer and a max-pooling layer of pool size 2 followed by a dropout of 0.6 is attached (shown in Figure 7). This described block is repeated for one more time with a kernel size of convolution layer as 64 before it is flattened. Then further we add a dense layer or a fully connected layer of 100 neurons, then an activation function of ReLU, then a batch normalization layer, then again, a dropout of 0.4 and then finally the last dense layer of size 2 for the two classes for classification and then a softmax activation function is used. Note that we used a dropout of 0.6 two times in starting and the last one was of 0.4. The architecture of 1D CNN is described in Table 4 where none means any batch size used.

### Optimization

Next, the model is compiled with loss as binary cross-entropy using an optimizer as Adam. The metric used was the area under the ROC curve. We created a custom
callback class in python to evaluate the train and test ROC after each epoch. The loss function is defined as:

\[
\text{Loss} = - \sum_{i=1}^{n} (y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i))
\]  

(16)

where \(n\) is the number of patients admitted in ICU, \(y_i\) is the binary indicator (0 or 1) of the mortality of the \(i\)th patient. Where 1 means in-hospital mortality and 0 means the survival, and \(\hat{y}_i\) represents the estimated value for \(i\)th patient by the model. Then we considered \(k\)-fold cross-validation for evaluating our model performance. Here, we considered \(k\) as 5, therefore 5-fold cross-validation. So, in each iteration, we have 4 folds for training making roughly a total of 3,200 rows and one fold with 800 rows for validation. The final batch size we used was 32, and the number of epochs was 150. We targeted lower learning rates with larger epochs to train our model.

**Model evaluation**

Here, we will describe how we went on to tune the hyperparameters of the statistical models other than 1D-CNN. First, we defined a parameter space for each of the parameter in the model. The parameter space is the domain from where a parameter value will be selected each time and that too randomly using a randomized search. For this, we defined a parameter grid, which corresponds to a
different combination of parameter spaces of each of the parameter. Second, as some models may have many parameters to tune, we did not use all parameters of a particular model in the parameter grid to deal with the time constraints. Therefore, we used a few dominant parameters (parameters having a large impact on models’ performance) in each model. For instance, we used C, gamma and kernel as our parameters to be searched for in the parameter grid of SVM. We checked for about 65–80% of the combinations in the parameter grid depending upon the model’s training time. We too changed the parameter spaces for each parameter a couple of times, thus giving birth to new combinations each time and hence new parameter grids. Figure 8 shows the training of the model.

Once we achieved the outcome of the 5-fold cross-validation for each combination selected by randomized search, we took the best out of them and noted the best corresponding parameters as well. Then we went for the grid search or exhaustive search with 5-fold cross-validation for testing all the possible combinations in the vicinity of the best parameter values achieved as an output from a randomized search. Thus, to compile the above thought we used multiple randomized searches for each model, declared the best search out of them, took the corresponding best parameter values, created new parameter space in the vicinity of received best parameter values and grid searched them. We even experimented with different random states.

## Results

All models, RF, XGBoost, LGBM, DT, KNN, SVM, LSTM, LSTM-AT and 1-D CNN were constructed to predict the vital status of the patient as an outcome. We place high attention on AUC for evaluating the performance of the models. The best model that we come up with was 1D-CNN with an AUC of 0.8480, as shown in Figure 9.

We experimented with multiple architectures and each with multiple hyperparameter combinations considering the training resources that we had. The layers were added, deleted and jumbled up multiple times to test the outcomes of the model. Numerous combinations of batch size, optimizers and different constants in learning rate decay function were tested before arriving at a decision. Since the CNNs are very sensitive to batch size and learning rate, we took special care of them too. Apart from the custom callback for evaluating and saving the model weights with the highest ROC achieved while training, we used two other callbacks. First, model checkpoint to save the best model with least validation loss and the other was the learning rate scheduler function which returned the best model with least validation loss and the other was the learning rate scheduler function which returned exponential decaying learning rate (Eq. (17)) after each epoch and took current epoch number as an input. Table 5 establishes the effectiveness of feature engineering by stating the best ROC with constructed features under consideration.

\[
a\prime = a\cdot e^{-k\cdot\text{epoch number}}
\]  

(17)

where \(a\) is a new learning rate, \(a\) is initial learning rate with initial value as 0.1 and \(k\) is constant and is assigned a value of 0.01. Table 6 compares the existing AUC and accuracy for the clinical event of mortality prediction.

## Conclusion

The objective of the research was to utilize the relations among variables and thus constructing new variables from

---

### Table 5: Accuracy and AUC of various algorithms on train and test set.

| Models       | Train  | Test   |
|--------------|--------|--------|
|              | Accuracy | AUC | Accuracy | AUC |
| XGBoost      | 0.9474 | 0.9783 | 0.8525 | 0.8294 |
|             | PRE    | PRE+FE | PRE+FE | PRE+FE |
| LBGM         | 0.9421 | 0.9801 | 0.8612 | 0.8398 |
|             | PRE    | PRE+FE | PRE+FE | PRE+FE |
| 1D-CNN       | 0.8765 | 0.9193 | 0.8638 | 0.815  |
|             | PRE    | PRE+FE | PRE+FE | PRE+FE |
| SVM          | 0.8715 | 0.888 | 0.8587 | 0.7914 |
|             | PRE    | PRE+FE | PRE+FE | PRE+FE |
| RANDOM FOREST| 0.8743 | 0.913  | 0.8575 | 0.8261 |
| KNN          | 0.8635 | 0.8133 | 0.8525 | 0.7866 |
|             | PRE    | PRE+FE | PRE+FE |
| DECISION TREE| 0.8705 | 0.823  | 0.8312 | 0.7358 |
|             | PRE    | PRE+FE | PRE+FE |
| LSTM         | PRE    | –      | 0.8525 | 0.7593 |
| LSTM-AT      | PRE    | –      | 0.8013 | 0.8357 |

### Table 6: Comparison with the existing literature.

| Authors                 | Method            | AUC  |
|-------------------------|-------------------|------|
| Ghose et al., 2015 [29] | Random forest     | 0.79 |
| Awad et al., 2019 [22]  | Random forest and Bayesian network | 0.83 |
| Che et al., 2015 [30]   | Deep computational phenotyping | 0.82 |
| Bhattacharya et al., 2017 [31] | CHISQ | 0.837 |
| Proposed                | 1-D CNN           | 0.848 |
the pool of variables available. These new variables proved to be important for predicting mortality among patients. Compared with the traditional machine learning models, 1-D CNN model has produced promising results with AUC of 0.8480. For testing the accuracy, first 48 h of recorded data (lab recordings, ICU bedside recordings) was utilized for accomplishing the objective. Total nine models were developed including six baseline models, two recurrent models and one 1-D CNN model. Patients admission information, health condition, etc. served as the predictor variable, and discharge status of ICU served as the target variable.

This work acknowledges the impractical application of traditional baseline models (RF, XGBoost, LGBM, DT, KNN, SVM) for predicting ICU mortality. This is because of the complexity associated with the recorded data (time series patients characteristics, etc.). Also, these methods use statistical techniques like mean, median and mode to deal with the time series nature of the dataset. However, there are chances of losing information in these statistics. Consequently, they are now inadequate to produce good results. Time series models like LSTM and LSTM-attention are also used which are capable of exploiting the time series information in the dataset. We also establish that the result of the LSTM and LSTM-attention was remarkable on this dataset.

The major contributions of this paper were first the construction of new features or variables with the use of feature engineering. The four constructed features establish their effectiveness for the prediction of mortality. Second, we had built time series models as well as traditional models for predicting the clinical event. 1-D Convolutional Neural Network model has also been developed which has surpassed the baseline machine learning models. Inline of research for mortality prediction for ICU patients, our 1-D CNN model have presented promising results.

In future work, we can include expert knowledge into our framework. Additionally, we can include the staff notes about patients and ECG waveform signals, available in the related dataset to refine the results.

Research funding: None declared.

Author contributions: All the authors have accepted responsibility for the entire content of this submitted manuscript and approved submission.

Competing interests: The authors declare no conflict of interest.

Ethical approval: The conducted research is not related to either human or animal use.

References

1. Maheshwari S, Verma R, Shukla A, Tiwari R. Feature engineering combined with 1D convolutional neural network for improved mortality prediction; 2019. arXiv preprint arXiv: 1912.03789.
2. Ding Y, Wang Y, Zhou D. Mortality prediction for ICU patients combining just-in-time learning and extreme learning machine. Neurocomputing 2018;281:12–9.
3. Berger JT, Holubkov R, Reeder R, Wessel DL, Meert K, Berg RA, et al. Eunice Kennedy Shriver National Institute of Child health and human development Collaborative pediatric critical care research network. Morbidity and mortality prediction in pediatric heart surgery: physiological profiles and surgical complexity. J Thorac Cardiovasc Surg 2017;154:620–8.
4. Le Gall JR, Loirat P, Alperovitch A, Glaser P, Granthil C, Mathieu D, et al. A simplified acute physiology score for ICU patients. Crit Care Med 1984;12:975–7.
5. Knaus WA, Zimmerman JE, Wagner DP, Draper EA, Lawrence DE. APACHE-acute physiology and chronic health evaluation: a physiologically based classification system. Crit Care Med 1981; 9:591–7.
6. Le Gall JR, Lemeshow S, Saulnier F. A new simplified acute physiology score (SAPSI) based on a European/North American multicenter study. Jama 1993;270:2957–63.
7. Knaus WA, Draper EA, Wagner DP, Zimmerman JE. APACHE II: a severity of disease classification system. Crit Care Med 1985;13: 818–29.
8. Knaus WA, Wagner DP, Draper EA, Zimmerman JE, Bergner M, Bastos PG, et al. The APACHE III prognostic system: risk prediction of hospital mortality for critically ill hospitalized adults. Chest 1991;100:1619–36.
9. Poole D, Rossi C, Anghileri A, Giardino M, Latronico N, Radrizzani D, et al. External validation of the Simplified Acute Physiology Score (SAPS) 3 in a cohort of 28,357 patients from 147 Italian intensive care units. Intensive Care Med 2009;35:1916.
10. Katsaragakis S, Papadimitropoulos K, Antonakis P, Stergiopoulos S, Konstadoulakis MM, Androulakis G. Comparison of acute physiology and chronic health evaluation II (APACHE II) and simplified acute physiology score II (SAPS II) scoring systems in a single Greek intensive care unit. Crit Care Med 2000;28:426–32.
11. Beck DH, Smith GB, Pappachan JV, Millar B. External validation of the SAPS II, APACHE II and APACHE III prognostic models in South England: a multicentre study. Intensive Care Med 2003;29: 249–56.
12. Nassar AP Jr., Mocelin AO, Nunes AL, Giannini FP, Brauer L, Andrade FM, et al. Caution when using prognostic models: a prospective comparison of 3 recent prognostic models. J Crit Care 2012;27:423–e1.
13. De Lange DW, Brinkman S, Flaatten H, Boumendil A, Morandi A, Andersen FH, et al. Cumulative prognostic score predicting mortality in patients older than 80 years admitted to the ICU. J Am Geriatr Soc 2019;67:1263–7.
14. Ngulie-Makao M, Zahar JR, François A, Tabah A, Garrouste-Orgeas M, Allaouchiche B, et al. Attributable mortality of ventilator-associated pneumonia: respective impact of main characteristics at ICU admission and VAP onset using conditional logistic
regression and multi-state models. Intensive Care Med 2010;36:781–9.
15. Rosenberg AL. Recent innovations in intensive care unit risk-prediction models. Curr Opin Crit Care 2002;8:321–30.
16. Xu J, Zhang Y, Zhang P, Mahmood A, Li Y, Khatoon S. Data mining on ICU mortality prediction using early temporal data: a survey. Int J Inf Technol Deci Mak 2017;16:117–59.
17. Delen D, Walker G, Kadam A. Predicting breast cancer survivability: a comparison of three data mining methods. Artif Intell Med 2005;34:113–27.
18. Sierra B, Serrano N, Larrañaga P, Plasencia EJ, Inza I, JiménezJJ, et al. Using Bayesian networks in the construction of a bi-level multi-classifier. A case study using intensive care unit patients data. Artif Intell Med 2001;22:233–48.
19. Vieira SM, Mendonça LF, Farinha GJ, Sousa JM. Modified binary PSO for feature selection using SVM applied to mortality prediction of septic patients. Appl Soft Comput 2013;13:3494–504.
20. Liu J, Chen XX, Fang L, Li JX, Yang T, Zhan Q, et al. Mortality prediction based on imbalanced high-dimensional ICU big data. Comput Ind 2018;98:218–25.
21. Greff K, Srivastava RK, Koutnik J, Steunebrink BR, Schmidhuber J. LSTM: a search space odyssey. IEEE Trans Neur Netw Learn Syst 2016;28:2222–32.
22. Awad A, Bader-El-Den M, McNicholas J, Briggs J, El-Sonbaty Y. Predicting hospital mortality for intensive care unit patients: time-series analysis. Health Inf J 2020;26:1043–59.
23. Goldberger AL, Amaral LA, Glass L, Hausdorff JM, Ivanov PC, Mark RG, et al. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. Circulation 2000;101:e215–20.
24. BUN to Creatinine ratio. Wikipedia. Available from: https://en.wikipedia.org/wiki/BUN-to-creatinine_ratio [Accessed: 24 Oct 2018].
25. Jha BK, Sharma MR. Correlation between serum albumin and initial GCS in patient with head injury in a tertiary hospital. J Soc Surg Nepal 2015;18:63.
26. Bernard F, Al-Tamimi YZ, Chatfield D, Lynch AG, Matta BF, Menon DK. Serum albumin level as a predictor of outcome in traumatic brain injury: potential for treatment. J Trauma Acute Care Surg 2008;64:872–5.
27. Pandey MK, Baranwal SK, Panwar DS, Saha SK, Roy K, Ghosh S, et al. Serial estimation of serum albumin and its role in traumatic brain injury patients. Asian J Med Sci 2016;7:31–8.
28. Ince T, Kiranyaz S, Eren L, Askar M, Gabbouj M. Real-time motor fault detection by 1-D convolutional neural networks. IEEE Trans Ind Electron 2016;63:7067–75.
29. Ghose S, Mitra J, Khanna S, Dowling J. An improved patient-specific mortality risk prediction in ICU in a random forest classification framework. Stud Health Technol Inform 2015;214:56–61.
30. Che Z, Kale D, Li W, Bahadori MT, Liu Y. Deep computational phenotyping. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Sydney: Association for Computing Machinery; 2015:507–16 pp.
31. Bhattacharya S, Rajan V, Shrivastava H. Icu mortality prediction: a classification algorithm for imbalanced datasets In: Thirty-First AAAI Conference on Artificial Intelligence. San Francisco: AAAI Press; 2017.