Using a LSTM-RNN Based Deep Learning Framework for ICU Mortality Prediction

Hanzhong Zheng[1] and Dejia Shi[2(✉)]

[1] Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15213, USA
haz78@pitt.edu

[2] Key Laboratory of Hunan Province for New Retail Virtual Reality Technology, Hunan University of Commerce, Changsha 410205, China
shidejia@126.com

Abstract. In Intensive Care Units (ICU), the machine learning technique has been widely used in ICU patient data. A mortality risky model can provide assessment on patients' current and when the disease may worsen. The prediction of mortality outcomes even intervenes doctor's decision making on patient's treatment. Based on the patient's condition, a timely intervention treatment is adopted to prevent the patient's condition gets worse. However, the common major challenges in ICU patient data are irregular data sampling and missing variables values. In this paper, we used a statistical approach to preprocess the data. We introduced a data imputation method based on Gaussian process and proposed a deep learning technology using LSTM-RUN that emphasizes on long time dependency relation inside the patient data records to predict the probability of patient's mortality in ICU. The experiment results show that LSTM improved the mortality prediction accuracy than base RNN using the new statistical imputation method for handling missing data problem.

Keywords: Deep learning · Recurrent neural network
Mortality prediction · Gaussian process

1 Introduction

Intensive Care Unit, also referred as ICU, is the important unit in modern hospital for saving patients with serious diseases. In the past several decades, the number of ICUs has dramatically increased by 50%. As populations in many countries age, doctors who can work in emergency and ICU could become increasingly pressed for time. For example, by the end of 2015, the number of people in China who is above 60 years old is approximately 222 million, which is 16.1% of the total population. Among them, 143.86 million people aged 65 or above, accounting for 10.5% of the total population. Now, the number of people who

X. Meng et al. (Eds.): WISA 2018, LNCS 11242, pp. 60–67, 2018.
https://doi.org/10.1007/978-3-030-02934-0_6

is above 85 years old in U.S. is 3 million. This number is estimated to be 9 million in 2030, which will bring great pressure to ICU. Automation may be an important solution to this problem. Under the background of rapid development of machine learning, many researchers try to use data mining and deep learning approach to study the mortality prediction problem for ICU patients.

Nowadays, machine learning techniques have been widely used in medical fields, such as the diagnosis procedure [3], gentic information extraction [6], etc. Continuous monitoring patients in ICUs can easily generate sufficient amount of medical records, which provide large enough amount of medical data to build a risk assessment model for ICU patients. This model can be used to evaluate the current patient's condition and predict the mortality probabilities at each timestamp to prevent the circumstance of patient worsen. The prediction of ICU outcomes is essential to underpin critical care quality improvement programs.

Deep learning neural network has also applied in the area of medical research: classifying the bio-medical text, disease symptoms identification and visualization, bio-medical images analysis, etc. However, Electronic Health Records (EMR) is another source of information that can be used to provide the assistance on disease diagnosis or evaluation on caring procedure for patients. However, EMR is very different comparing with other medical data resources. EMR has the time dependency inside the data. Deep learning neural network is a forward-feeding neural network that is not suitable for modeling the time dependent data. In this paper, we used a Recurrent Neural Network (RNN) to model the time-series data. The Recurrence in the RNN allows it to remember the information from previous calculation and the previous information will influence the calculation on current input. In addition to base RNN, we also experimented LSTM-RNN, which is a variation of RNN. Comparing with base RNN, the LSTM-RNN has the long term memory that can memorize the information from the calculations in the much further time stamps. For the data that crosses over a long time interval, LSTM-RNN is more suitable than base RNN.

In this paper, we studied the problem inside the data set: irregular sampling and missing values and built two deep learning neural network models using base-RNN and LSTM-RNN. We used the supervised learning method to train and test our models. Then, we compared the test results of RNN, LSTM-RNN and other machine learning algorithms to evaluate their performance on real hospital data.

2 Related Work

The irregular data sampling in medical records is a very common problem. Many researches have done using the LSTM to solve the time irregularities. Inci Baytas et al. [1] proposed a novel LSTM framework called Time-Aware LSTM, also referred as T-LSTM. In their approach, they modified the sigmoid layer of the LSTM cell, which enables time decay to adjust the memory content in the cell. Their experiment results indicate the T-LSTM architecture is able to cluster the patients into clinical subtypes. Che et al. [2] studied the task of pattern

recognition and feature extraction in clinical time series data. They used a different variation of recurrent neural network so called GRU-RNN that can also uses the prior knowledge. Che evaluated their model on two real-word hospital data sets and showed their neural nets can learn interpretable and clinical relevant features form the data set. Harutyunyan [4] also used deep learning framework to make predictions on clinical times series data. In their work, they studied multiple tasks involving modeling risk of mortality, forecasting length of stay in ICU, detecting physiological decline, and phenotype classification. They built a RNN model to explore the correlations between those multiple tasks. However, they only explored the traditional data imputation method that fills the missing data using the summary statistics. The tradition data imputation method ignores the correlation between variables. For example, the variable Temperature and Heart Rate may be highly correlated. High Temperature value also could also indicate a high Heart Rate value. However, if we impute the low mean Heart Rate value under a high Temperature condition, it could influence the prediction accuracy.

In this paper, we focused on developing a new data imputation approach using Gaussian process and propose a deep learning framework to predict the probability of mortality in ICU on real hospital patient data. For this prediction task, we built and compared the performance of base-RNN and LSTM-RNN model, especially on false positive errors made by these two models.

3 Data Imputation and Multivariate Data Modelling

In this paper, we used ICU data set: *The PhysioNet*, it includes over 4000 patient records. Each record maintains the 36 variables measurement at least once during the first 48 h after admission to the ICU. Each patient has a result variable: *In-hospital death* is a binary value (0: survivor, or 1: died in-hospital). However, there are three major problems existing in this data set: (1) missing value problem: not all variables are available in all cases. For example, at time stamp t_i, there could be only 7 values out of 36 variables. (2) Irregular sampling: The each record was measured at irregular time stamp. Patient's measurements were taking at different time stamp. The interval between two measurements are not the same. These two problems require the data pre-processing before using the data to train our model. (3) "Imbalanced" data sets: the number of dead patients only contains a very small proportion of the data set.

We use the time window and statistical summary imputation method to manually fill the missing values. To be more specific, we divide each patient's record into equal length window and the length of the window can be 2 h, 5 h, 10 h, etc. For each of the variable, we use the 5 summary statistics min, max, mean, median, standard deviation. Using the summary statistic of a time interval can solve the problem of missing data at a specific time stamp. However, this dataset has the problem of serious missing data. Many time interval still does not have the values.

At each time stamp t_i, we use a tensor that contains the value of each variable for each data entry. We proposed to use statistical model to study overall data

form for each tensor. For each missing interval value, we introduced a Gaussian process that estimates the mean and variance from the recorded measurements. Let $\chi = \{x_1, x_2, x_3, ..., x_j\}$ be the collection of tensors from the patient record with j number of known measurements, in particularly, we can denoted it as $\{f(x_i) : x_i \in \chi\}$, where which is drawn from a Gaussian process with a mean function $m(\cdot)$ and kernel function $k(\cdot, \cdot)$. Then, the distribution of the set χ is denoted as,

$$\begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_j) \end{bmatrix} \sim N \left(\begin{bmatrix} m(x_1) \\ m(x_2) \\ \vdots \\ m(x_j) \end{bmatrix} \begin{bmatrix} k(x_1, x_2) \ldots k(x_i, x_j) \\ k(x_2, x_1) \ldots k(x_2, x_j) \\ \vdots \quad \ddots \quad \vdots \\ k(x_j, x_1) \ldots k(x_j, x_j) \end{bmatrix} \right)$$

or $f(\cdot) \sim gp(m(\cdot), k(\cdot, \cdot))$. The purpose of kernel function is to transform to a valid covariance matrix corresponding to some multivariate Gaussian distribution. For a kernel transformation, the kernel function must satisfy the Mercer's condition (illustrated in Definition 1). In Mercer's condition, the function needs to be square-integrable (illustrate in Definition 2) Therefore, we choose the squared exponential kernel, defined in Eq. (1), where parameter τ determines the smoothness of the Gaussian process prior with $k_{SE}(\cdot, \cdot)$.

$$k_{SE}(x_i, x_j) = exp(-\frac{1}{2\tau^2} \left\| x_i - x_j \right\|^2) \tag{1}$$

Definition 1. *A real-valued kernel function $K(x, y)$ satisfies Mercer's condition if $\int \int K(x, y)g(x)g(y)dxdy \geq 0$ for all square-integrable functions $g(\cdot)$.*

Definition 2. *A function $g(x)$ is square-integrable if $\int_{-\infty}^{+\infty} |g(x)|^2 \, dx < \infty$*

Then each patient record can be modelled through multivariate Gaussian distribution, illustrated in Eq. (2), where $\mu = m(\cdot)$ and $\mathbf{\Sigma} = k(\cdot, \cdot)$.

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} \left| \boldsymbol{\Sigma} \right|^{1/2}} exp \left| -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right| \tag{2}$$

4 RNN and LSTM-RNN

Comparing with feedforward network, the recurrent neural network takes the current input and it also takes the what they previously perceived. The information from previous inputs can be kept into the hidden layers in the RNN, which will influence the calculation of the current input. The main difference between the recurrent network and feedforward is the feedback look connected to their past decisions.

$$h_t = \Phi(W * x_t + U * h_{t-1}) \tag{3}$$

Equation (3) shows the mathematical expression of updating the hidden layers in RNN. It takes the previous hidden layer h_{t-1} and current input x_t to

calculate the current hidden layer output. However, the main disadvantage of the RNN is the "long term memorization". To be more explicit, if there are two data inputs d_i and d_j across a long time interval, the RNN cannot remember the information from the input d_i when it does the calculation on current input d_j. From the PhysioNet, each patient record contains the information more than 48 h in ICU. Using the RNN may not be able to "remember" the patients' information many hours ago. The loss of information in the neural network is referred as the "vanishing gradient" problem.

A variation of recurrent neural network, so called Long Short-Term Memory Unit (LSTM), was proposed by the German researchers Sepp Hochreiter and Juergen Schmidhuber as a solution to the vanishing gradient problem [5]. The architecture of the LSTM can be viewed as a gated cell. The cell decides which information will be remembered, or forgot through gate opening and closing. By maintaining this gate switch, it allows LSTM to continue to learn over a long time interval.

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{4}$$

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{5}$$

$$\tilde{C}_t = tanh(W_c * [h_{t-1}, x_t] + b_c) \tag{6}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{7}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{8}$$

$$h_t = o_t * tanh(C_t) \tag{9}$$

The above 6 equations illustrate the update procedure for a layer of memory cell between time stamp $t-1$ and t. The sigmoid layer, also called "forget gate layer" decides what information will be dropped out from the cell, illustrated in Eq. (4). Equations (5) and (6) refer to the "input gate layer", which contains two parts: one sigmoid layer and one tanh layers. This sigmoid layer decides what information the cell will update and the tanh layer controls the new information will be stored into the cell. The Eq. (5) illustrates the process of forgetting information and updating information. Eventually, the LSTM cell will generate outputs using Eqs. (7), (8), and (9), where h_t is the output of the hidden layer and C_t is the output of the cell, which represented as a tensor with 2 dimensions. Since the LSTM decides to drop up some information at each time stamp, it is able to store the information from longer time stamp, when comparing with base-RNN. Then, we defined the softmax layer that maps the outputs generated by the LSTM cell into the probability representation using Eq. (10), where $f(C_{t_i})$ denotes as the probability of class i.

$$f(C_{t_i}) = \frac{exp^{C_{t_i}}}{\sum_j^{|C_t|} exp^{C_{t_j}}} \tag{10}$$

5 Results and Discussion

We built two neural networks RNN and LSTM-RNN with the same structure: 36 feature inputs, 1296 hidden units with 2 layers. We split the 4000 data samples into the training group and testing group. In order to resolve the "imbalanced" number of dead patients records and survival patients. We randomly selected 400 survival patients and 400 dead patients as the training set and 200 survival patients and 200 dead patients as the testing set.

The output of the model is two probabilities: $[Prob(survival), Prob(dead)]$, denoted as $(v1), p(v2)]$ If $p(v1) > p(v2)$, then we classify the patient as dead (0), otherwise, we classify the patient as survival (1). We used the mean squared error as the loss measurement of the model. The mean squared error is measured by the sum of the variance of the model and the squared bias of the model. If the patient outcome is survival (0), then the target variable is $[1, 0]$, where can be interpreted as $[p(v1) = 1, p(v2) = 0]$. If the patient outcome is dead (1), then the target variable is [0,1].

$$MSE = \frac{1}{N} * \sum^{N} [(p(v_1) - \widehat{p(v_1)})^2 + (p(v_2) - \widehat{p(v_2)})^2] \tag{11}$$

Equation (11) is the mathematical expression of the Mean Squared Error measurement of our model, where N is the batch size. The model uses the loss during the learning phase to gradually adjust the model until there is no improvement or very small improvement.

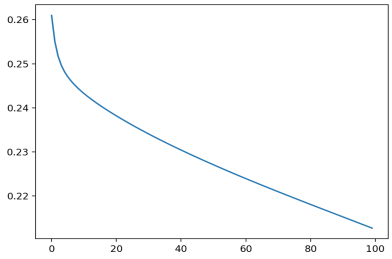

Fig. 1. The loss of the RNN: epoch = 100, learning rate = 0.01, batch size = 800, window size = 10 h.

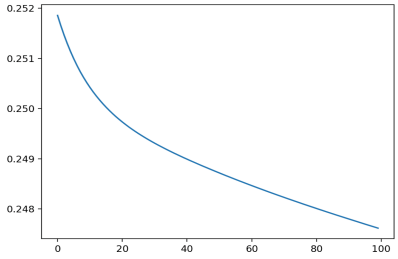

Fig. 2. The loss of the LSTM: epoch = 100, learning rate = 0.01, batch size = 800, window size = 10 h

The epoch actually represents the learning phase of RNN and LSTM. Figures 1 and 2 show the loss of RNN and LSTM at each epoch. The loss is calculated by MSE mathematical function and it indicates the model's learning outcomes. Both RNN and LSTM can reduce their loss through each epoch. At the initial, the loss of LSTM is lower than RNN; then during the beginning, both of RNN and LSTM can rapidly reduce their loss. However, at the end, RNN received lower loss than LSTM. The window size could be the reason because the model

Table 1. Confusion Matrix of RNN and LSTM on testing data: batch size = 400 (200 survival and 200 dead)

	RNN: survival	RNN: dead	LSTM: survival	LSTM: dead
Target: survival	137	63	170	30
Target: dead	90	110	55	145

Table 2. Evaluation statistics of RNN and LSTM

	RNN	LSTM
Specificity	60.35%	82.86%
Sensitivity	63.58%	75.56%

does not need to remember too many previous information when we have a large window size.

From the Tables 1 and 2 of RNN and LSTM, even though the RNN can achieve a lower loss than LSTM, the testing result shows that LSTM did a better job than RNN for time-series data that has a long term dependency. For ICU mortality prediction, the most important error is the false positive. LSTM had lower errors than RNN. For specificity and sensitivity, the LSTM also achieves higher values than RNN.

Figure 3 also shows that the ROC curve of LSTM is always higher than the RNN. In Table 3, we compared several different machine learning algorithms. Th

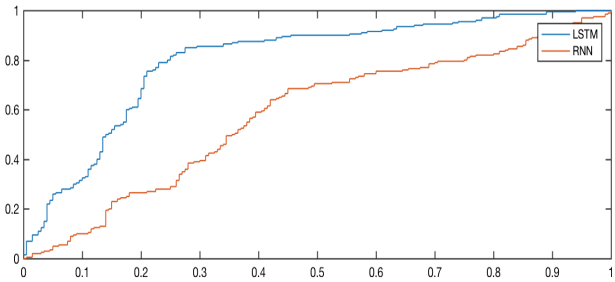

Fig. 3. The ROC curve of RNN and LSTM

Table 3. The comparison of the AUC score of different machine learning algorithms

Algorithm	AUC score	Algorithm	AUC score
SVM	0.563	LDA	0.608
QDA	0.673	LSTM	0.8025
RNN	0.581	RF	0.642
LR	0.602		

AUC value of LSTM is the highest among the results obtained from different algorithms. Therefore, for the task of modeling time-series data, especially for the long term data, LSTM can produce an improved prediction results using Gaussian data imputation method among different algorithms.

6 Conclusion and Future Work

The major problems of Electronic Health Record (EMR) are irregular data sampling and missing values. In the paper, we imputed the missing variable values by 5 summary statistics. The mean and stand deviation values were modeled by multivariate Gaussian distribution through kernalization of Gaussian process, which ensures the correlation between variables is considered into the imputation process. The recurrent neural network emphasizes on the time dependency relationship in the data. The experimental results indicate that LSTM produces higher accuracy than RNN on modeling time series data that has the long term dependency.

For the future work, we plan to use a Convolutional Neural Network based LSTM (CNN-LSTM). We can consider the each patient record as an image and use the CNN to automatically extract useful features. In addition, we also need to consider that whether the missing values are also informative. Seriously ill patients normally has less missing variable values than less ill patients. Therefore, we can use indicator variables for each value. For example, if the variable's value is missing, the indicator sets to 0, otherwise, the indicator sets to 1. Then, the indicator variables would also be the input of the LSTM.

References

1. Baytas, I.M., Xiao, C., Zhang, X., Wang, F., Jain, A.K., Zhou, J.: Patient subtyping via time-aware LSTM networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2017, pp. 65–74. ACM, New York (2017). https://doi.org/10.1145/3097983.3097997
2. Che, Z., Kale, D., Li, W., Bahadori, M.T., Liu, Y.: Deep computational phenotyping. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2015, pp. 507–516. ACM, New York (2015). https://doi.org/10.1145/2783258.2783365
3. Deligiannis, P., Loidl, H.W., Kouidi, E.: Improving the diagnosis of mild hypertrophic cardiomyopathy with mapreduce. In: Proceedings of Third International Workshop on MapReduce and Its Applications Date, pp. 41–48, MapReduce 2012. ACM, New York, NY, USA (2012). https://doi.org/10.1145/2287016.2287025
4. Harutyunyan, H., Khachatrian, H., Kale, D., Galstyan, A.: Multitask learning and benchmarking with clinical time series data, March 2017
5. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
6. Hussain, A.: Machine learning approaches for extracting genetic medical data information. In: Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing, ICC 2017, p. 1. ACM, New York (2017). https://doi.org/10.1145/3018896.3066906