Early Sepsis Detection Through Time-Aware Embedding and Temporal Short-term Memory-based LSTM

Kanan Mahammadli
Mathematics Department
Middle East Technical University
Ankara, Turkey
kanan.mahammadli@metu.edu.tr

Erdem Akagündüz
Graduate School of Informatics
Middle East Technical University
Ankara, Turkey
akaerdem@metu.edu.tr

Abstract— Early sepsis detection is crucial as it has a 6 million per year mortality rate, and it is one of the costliest medical conditions. Yet, it is still challenging to forecast sepsis onset due to the lack of a general methodology for identifying dynamic patterns ofthe syndrome. Deep learning approaches, especially recurrent neural networks-based architectures, have been explored for automatic sepsis detection, considering fixed 4, 6, and 8 hours before sepsis onset. As the last 6 hours before sepsis onset can have life- threatening ramifications, it is more important to forecast sepsis till the 6th hour in a multistep-ahead manner. This study aims to test the hypothesis that sepsis onset can be determined before crucial hoursby focusing on data enrichment and custom modeling methodology for sepsis-specific data patterns. This paper suggests: i) data cleaning pipeline and feature engineering through masking, lagging, rolling window, and delta features for data enrichment, ii) time-aware embeddings and temporal feature-based short-term memory to capture irregularities in data, and iii) customized loss function to handle imbalanced sequential learning. The system achieves an f1- score of 0.097 on validation data and 0.116 on test data, close to current short-term sepsis detection scores in the literature, around 0.1.

Keywords—sepsis, time-aware embedding, temporal short-term memory, feature engineering

I. Introduction

Sepsis is a life-threatening condition triggered by the body's improper response to the infection. It can lead to tissue damage, organ failure, and, eventually, death [1]. Failing to detect this syndrome on time increases the chance of fatality, and makes treatment harder. The first few hours of sepsis are considered vital, as patients can be given antibiotics to avoid damage to internal organs fighting the infection. These criticalhours start from 6 hours (6h) before the actual sepsis damages the patient's body. Sepsis identification systems by doctors try to consider 6h, 12h, 24h, and 48h before detecting sepsis onset, with 6h being the last critical point for effective treatment. [2]. Besides human loss, sepsis is also one of the leading costly conditions in the world, adding economic damage of 17B USD annually to the healthcare system in the US alone [2]. Therefore, automated early sepsis forecasting can help save lives and reduce medical costs.

II. Literature Review

Various tree-based machine learning algorithms [2], [3] and sequential deep learning architectures, such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) [1], [4], [5], have been explored to detect sepsis onset 6 hours before the actual identification with the consideration of some data preprocessing and feature engineering methods. The authors have used data from *the PhysioNet Computing in Cardiology Challenge 2019* [6], which contains vital signs, lab results, demographics, and sepsis target values for patientsfrom two hospitals. Comprehensive details of the dataset will be explored in the dataset subsection of the methodology section. John et al. [2] have used the XGBoost algorithm with a weighted loss function and utilized 8 vital sign features, their variances, and log-transformation and achieved f1-scores of 0.152 and 0.144 on validation and test sets, respectively, to detect sepsis 6h before onset. Another tree-based model has been used by Zabihi et al. [3], with a combination of under-sampling and lagging features of 13h shifting to result in an f1-score of 0.058 and 0.044 on validation and test sets, respectively. Lee et al. [1] have also used 8 vital signs with addition of the duration of missing hours and masked labresults features and LSTM model with dropout to get 0.129 and 0.123 validation and test f1-scores. Roussel et al. [4] consider the LSTM model with skip connections in between to detect sepsis with vital signs and demographic features, resulting in an f1-score of 0.1 on the test set.

III. Problem Definition

Current state-of-the-art (SOTA) tree models and deep learning methods focus on detecting sepsis either 4h or 6h before onset, at its earliest detection, 8h beforehand. As for this detection, timestamps are either within critical hours or very close. There is a lack of multistep-ahead forecasting well before the critical point in the literature, and this study aims to fill that gap by providing automated sepsis detection within thefirst and second critical points - between the 12th hour and 6th hour before sepsis onset, making 7 steps ahead forecasting with close evaluation metric values to current SOTA results. To download the used for this research and reproduce the results, visit this paper's official GitHub Repository at github.com/KananMahammadli/SepsisForecasting

IV. Methodology

A. Dataset

As used in the discussed papers, this study also uses the PhysioNet dataset [6]. Data have been collected from hospitals, with 20336 patients in set A and 20000 in set B, where A and B are hospital identifiers. The dataset has 3 different feature types and target column: 8 Vital Signs – heart rate, pulse oximetry, temperature, systolic BP, mean arterial pressure, diastolic BP, respiration rate, and end-tidal carbon dioxide; 26 lab results, and 6 demographics: age, gender, time between hospital admission to Intensive Care Unit (ICU) admission, length of stay, admission identifiers unit1 and unit2; and sepsis target column representing time 6h before actual sepsis scenario. As the original dataset has sepsis indication before 6 hours of sepsis onset, target values are shifted 6 hours backward to have sepsis indication starting 12h before onset, and data points after the 6th hour have been cut, so that the model does not see patterns after passing critical point, as goalis to forecast before that point. Furthermore, challenges with the given dataset can be categorized as i) missing values – except glucose level, all other lab results are missing over 90%,and vital signs are missing around 10%; ii) irregular length-of-stay – patients' Electronic Health Records (EHR) data length differ too much, by having 9675 patients with less than 24 samples, and only 8575 patients with more than 24 samples;
iii) data is severely imbalanced with only around 2% of the target being positive samples

B. Preprocessing

Considering the challenges with the PhysioNet dataset, which is normal for EHR data, proper data analysis and cleaning are required to improve the data quality. First, patients who had already sepsis at the beginning of the record are removed as their data does not have a sepsis development sequence. Then, to handle irregular length-of-stays of patients, a minimum length-of-stay threshold has been used to filter out underrepresented patients from training. This threshold is a hyperparameter that has been tuned to 60. Lastly, the forward-fill method has been used to fill in missing values for vital signs. If values are missing at later steps of the measurements, then we can carry the previous ones forward. This process is followed by a backward fill so that if a vital sign starts with a missing value, the first measurement will be the starting point. If there are still missing vital signs, then some patients have missing vital signs, and these patients are removed from the training.

C. Feature Engineering

Various feature engineering methods have been applied based on the feature type. For lab results, the purpose is to represent a few measurements and a high level of missingness over 90%. To make the model learn from lab results as well, binary masking has been applied to these 26 features; if the value is missing, it has been filled with an arbitraryplaceholder, in this study, with zero, and if lab results exist, then 1. As an extra feature, only filling missing values with zeroes and keeping original values at non-missing points have been applied, too, resulting in doubled lab features. The modeling subsection will discuss later steps of featurerepresentation for irregular missingness. For vital signs, all 8 features have been utilized, and additionally, lagging features, rolling window-based min, max, mean values, and delta features have been added to provide a more historical representation of data for the model. As the target is 7 hours ahead of forecasts, to avoid leakage, lagging features start fromperiod 7 till the previous 13th period (the last period is 12); forexample, the lagging feature of period 8 for temperature is the temperature feature shifted 8 steps forward, and the first 8 values who has no history are filled with zeros. For rolling features, vital signs are first shifted by a period of 7 to skip target values, then min, max, and mean of vital features over the rolling window period are added as new features, where periods are 7, 12, and 24. Delta features are similar to lagging features in terms of period range – from period 7 to 13, and foreach timestep T, the delta feature of period N is calculated as:

$$(\text{vital\_sign}_{T-N} – \text{vital\_sign}_{T-2N})$$

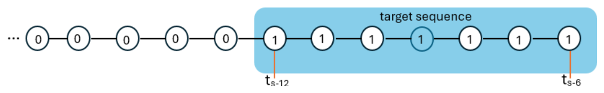

Fig. 1 Target Sequences

In other words, it is a difference between the last 2 periods' vital signs. Considering lab results and vital signs as dynamic features that can change over time, 210 dynamic features are engineered. From static features - demographics, only age, gender, and hospital admission time are considered; unit1, unit2, and length of stay features didn't yield any improvements.

D. *Modelling*

After data enrichment with cleaning, preprocessing, and feature engineering pipeline, patients are split into two groups: patients who develop sepsis at the end of the patient's records and patients who never develop sepsis. From each group, training, validation, and test sets are taken randomly at a 70:15:15 ratio, respectively, and combined to form final sets, such that each set will keep a ratio positive and negative class imbalance and there will not be a set of only non-sepsis patients. This ensures fair evaluation and helps measure the developed model's performance on unseen data containing both classes. Then, each patient's data in each set is converted into sequences of length 12, and this window length is tuned hyperparameter. If we consider a patient who develops sepsis at the end, the last target sequence will have the number of 1s in the sequence equal to the output length, which is 7, and the previous 10 timestamp features will be considered. This is equivalent to forecasting sepsis from timestamp t_{s-12} to t_{s-6}, considering the patient develops sepsis at timestamp s, using features from timestamp t_{s-24} to t_{s-13} (Figure 1).

Each sequence of window length 12 in the data contains both the last 12 timestamp vital signs and lab results and memory of previous vital sign dependencies thanks to extensive feature engineering. However, lab results are irregularly missing inthe dataset. Besides, for patients who develop sepsis, the number of samples for sepsis varies in length. Feature engineering alone cannot capture these time-dependent irregularities and variations, and passing raw features directly into LSTM does not lead to reliable results. Therefore, two assumptions are made in this study to handle this problem. The first assumption is that adding a time-aware embedding layer makes capturing patient-level patterns possible and transforms raw space into dense embedding space where noise will be preliminarily removed, and useful dependencies will be compressed.

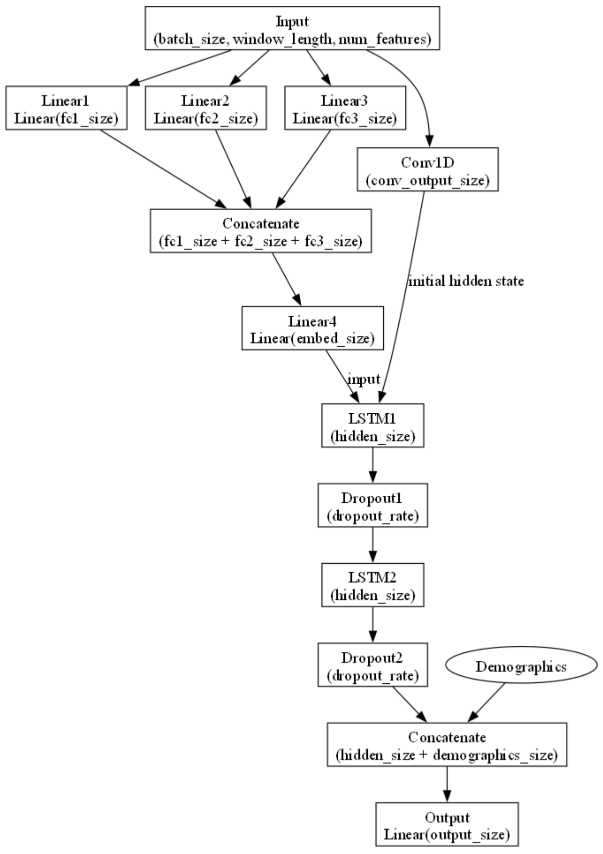

Fig. 2 Model Architecture

$$L_1(y,\hat{y},w) = \begin{cases} -y * log(\hat{y}) - (1-y) * log(1-\hat{y}) \; if \; \forall y_i = 0 \\ -w * y * log(\hat{y}) - (1-y) * log(1-\hat{y}) \; if \; \exists y_i = 1 \end{cases}$$

$$L = \sum_{i=1}^{N} w_i * L_1(y_i, \hat{y}_i, w)$$

Fig. 3 Loss function

To achieve this goal, 3 linear layers of output sizes 32, 64, and 100 are applied to input data in parallel, followed by ReLU non-linearity, so that each layer learns different timely varying irregularities, then outputs of these layers are concatenated and linearly transformed into embedding space of size 128, which reduces input size from 210 total dynamic features (Figure 2). The second assumption is that using the convolution layer will extract the temporal neighborhood dependencies and give convolution output as initial short-termmemory, or in other words, as an initial hidden state, which will provide LSTM with starting context information rather than starting from scratch. The convolution layer has tuned kernel size and output size of 3 and 32, respectively (Figure 2). After data preparation, the embedding output is given to the LSTM layer as an input and convolution output as an initial hidden state. As a single LSTM layer was underfitting, 2 LSTM layers were used, each has a hidden size aligned by window length and convolution output size, which are 12 and 32, respectively, making a hidden size of 384, and to avoid overfitting, dropout of rate 0.1 is added after each LSTM. Finally, the last hidden state of the last LSTM is concatenated with static demographic features, making the feature output size 387. These features are mapped to an output size of 7 by the final linear layer, and each of these 7 outputs is passed into sigmoid to detect if there is sepsis at that step or not. Due to severe imbalance in the data and positive classes being minority, a customized Binary Cross Entropy loss function is designed (Figure 3). If the sequence contains all no sepsis targets, then regular BCE loss is used; however, if sepsisexists at any step of output, then positive class errors arepenalized by extra weight, and this weight is tuned to 55. The reason for extra weight is that positive classes are minority and less frequent; also, having false negatives is more costly than having false positives. Besides, the output is 7 stepsahead of forecasting, and each following hour is closer to a critical point. Therefore, the sequence of weights is also applied so that errors in later timestamps of the sequence will be penalized more. There is no extra sequence weight for the first timestamp forecasting, so the weight is 1 here, and the following timestamp weight is 20. The rest have weights incremented by 10, making the last point have an extra weight of 120 (Figure 3). The Adam optimizer with a starting learning rate of 0.001 and an exponential decay-based learningrate scheduler of gamma value 0.1 is adapted to train the model. PyTorch library is used for model architecture buildingand dataset preparation, and PyTorch-Lightning is used for data processing and training, together with Neptune AI for experiment tracking.

V. Results and Discussion

The LSTM model has been trained in 4 different modes, with the latest mode being the main architecture utilized in this study and the first three modes being intermediate results through research. The first LSTM mode istraining based only on vital signs and demographic data, without feature engineering, embedding, or convolution operation. The second mode is engineered features added to the previous structure. The third mode considers engineered features with embeddings for LSTM inputs. Finally, mode 4 adds convolution-based short-term memory extraction,replacing the initial hidden state of LSTM with newly created memory, and keeps all previous improvements the same. The number of LSTM layers, hidden size, dropout rate, and batch sizes have been tuned for all modes. Besides, LSTM has been tested on bidirectional mode but has shown no improvement; therefore, it is not included in the evaluation. The main evaluation metric for this study is considered as f1-score (Figure 4) as data is highly imbalanced. However, AUROC score and Accuracy are also calculated to compare results with current literature methods over various evaluation metrics. From both Validation (Table 1) and Test Results (Table 2), it is clear that both the f1-score and AUROC score improve with each mode advances, with mode 4 having f1-scores of 0.097 and 0.116 and AUROC scores of 0.680 and 0.636 for validation and test sets, respectively. The most basic mode 1

	F1-score	AUROC	Accuracy
Lee et al.	0.129	0.793	0.825
John et al.	**0.152**	**0.880**	**0.901**
Zabihi et al.	0.128	0.814	-
Roussel et al.	0.123	0.798	0.854
LSTM Mode1	0.059	0.499	**0.988**
LSTM Mode 2	0.074	0.515	0.694
LSTM Mode 3	0.102	0.559	0.641
LSTM Mode 4	**0.097**	**0.680**	0.497

Table. 1 Val Results

	F1-score	AUROC	Accuracy
Lee et al.	0.123	0.812	**0.889**
John et al.	**0.144**	**0.846**	0.822
Zabihi et al.	0.044	0.793	-
Roussel et al.	0.117	0.793	0.836
LSTM Mode1	0.028	0.523	**0.981**
LSTM Mode 2	0.084	0.598	0.699
LSTM Mode 3	0.101	0.425	0.662
LSTM Mode 4	**0.116**	**0.636**	0.457

Table. 2 Test Results

$$\text{F1-score} = \frac{TP}{TP + \frac{1}{2}*(FP + FN)}$$

TP – True Positive

FP – False Positive

FN – False Negative

Fig. 4 F1-score formula

has the highest accuracy of 0.988 on validation and 0.981 on the test set with very low f1-scores, which indicates that the model cannot learn positive labels and is highly biased towards negative classes. As positive labels are less than 2 % in the data, and the same ratio is preserved in validation and test results, this explains high accuracy. Therefore, accuracy is not a reliable metric for this task. Compared to other methods in the literature, LSTM trained in mode 4 outperforms Zabihi et al.'s [3] test f1-score and is very close to Roussel et al.'s [4] LSTM model with only 0.001 lower. However, the LSTM mode 4 model could not surpass Lee et al.'s [1] transformer- based approach and John et al.'s [2] Tree-based solutions.Finally, the LSTM model in mode 4 is evaluated in 3-fold cross-validation, which resulted in an f1-score of 0.083 +/- 0.012, AUROC score of 0.405 +/- 0.053 and Accuracy of 0.191 +/- 0.059.

VI. Conclusions

In this study, various data cleaning and feature engineering methods have been suggested together with time-aware embeddings and convolution-based temporal short-term memory with LSTM to forecast sepsis onset from 12h before for 7 steps ahead till 6h before actual sepsis detection. It has been shown that, with proper data enrichment and alignmentof input data into sequence models, we can forecast sepsis onset well before the critical point by performing close SOTA methods in the literature that forecast sepsis onset for fixed 6h before. This paper's main improvements can be summarized as a result of the well-representation of features with high missing rates and irregular time variations. Therefore, future work can use customized RNN models to handle missing values internally with time-aware embedding inputs.

References

[1] B. Lee, K. Cho, O. Kwon, and Y. Lee, "Improving the Performance of a Neural Network for Early Prediction of Sepsis," 2019 Computing in Cardiology (CinC), Singapore, doi: 10.22489/CinC.2019.162.

[2] J. A. Du, N. Sadr and P. d. Chazal, "Automated Prediction of Sepsis Onset Using Gradient Boosted Decision Trees," 2019 Computing in Cardiology (CinC), Singapore, 2019, doi: 10.22489/CinC.2019.423.

[3] Zabihi, Morteza & Kiranyaz, Serkan & Gabbouj, Moncef. (2019). Sepsis Prediction in Intensive Care Unit Using Ensemble of XGboost Models. 10.22489/CinC.2019.238.

[4] B. Roussel, J. Behar and J. Oster, "A Recurrent Neural Network for the Prediction of Vital Sign Evolution and Sepsis in ICU," 2019 Computing in Cardiology (CinC), Singapore, doi: 10.22489/CinC.2019.082.

[5] Rafiei, A., Rezaee, A., Hajati, F., Gheisari, S., & Golzan, M. (2020). SSP: Early prediction of sepsis using fully connected LSTM-CNN model. *Computers in Biology and Medicine*, *128*, 104110.https://doi.org/10.1016/j.compbiomed.2020.104110.

[6] Reyna, M. A., Josef, C. S., Jeter, R., Shashikumar, S. P., Westover, M. B., Nemati, S., Clifford, G. D., & Sharma, A. (2020). Early Prediction of Sepsis From Clinical Data: The PhysioNet/Computing in Cardiology Challenge 2019. *Critical care medicine*, *48*(2), 210–217. https://doi.org/10.1097/CCM.0000000000004145