International Journal of Business Quantitative Economics and Applied Management Research
Volume-6, Issue-06, 2020 ISSN No: 2349-5677

EARLY DETECTION OF SEPSIS USING CLINICAL DATA

Chananagari Prabhakar Rohit Reddy,
Computer Science and Engineering,
CMR Technical Campus
cprohit1998@gmail.com

Abstract

The timeliness of detection of a sepsis incidence in progress is a crucial factor in the outcome for the patient. Machine learning (ML) models built from data in electronic health records (EHR) can be used as an effective tool for improving this timeliness, but so far the potential for clinical implementations has been largely limited to studies in intensive care units (ICUs). This study will employ a richer data set that will expand the applicability of these models beyond ICUs. Furthermore, we will circumvent several important limitations that have been found in the literature: (1) Models are evaluated shortly before sepsis onset without considering interventions already initiated. (2) ML models are built on a restricted set of clinical parameters, which are not necessarily measured in all departments. (3) Model performance is limited by current knowledge of sepsis, as feature interactions and time dependencies are hard coded into the model. Methods: In this study, we present a model to overcome these shortcomings using a deep learning approach on a diverse multicenter data set. We used retrospective data from multiple Danish hospitals over a seven-year period. Our sepsis detection system is constructed as a combination of a convolutional neural network (CNN) and a long short-term memory (LSTM) network. We suggest a retrospective assessment of interventions by looking at intravenous antibiotics and blood cultures preceding the prediction time. Results: Results show performance ranging from AUROC 0.856 (3 hours before sepsis onset) to AUROC 0.756 (24 hours before sepsis onset). Evaluating the clinical utility of the model, we find that a large proportion of septic patients did not receive antibiotic treatment or blood culture at the time of the sepsis prediction, and the model could therefore facilitate such interventions at an earlier point in time. Conclusion: We present a deep learning system for early detection of sepsis that can learn characteristics of the key factors and interactions from the raw event sequence data itself, without relying on a labor-intensive feature extraction work. Our system outperforms baseline models, such as gradient boosting, which rely on specific data elements and therefore suffer from many missing values in our dataset.

Keywords—Sepsis; Machine Learning; Deep Learning; Early Detection; Clinical Data; CNN; LSTM

I. INTRODUCTION

Sepsis is a life-threatening organ dysfunction caused by a dysregulated response to infection, and early detection is crucial for improving patient outcomes. Traditional clinical scoring

systems, such as the National Early Warning Score (NEWS), may not adequately identify at-risk patients with sufficient lead time. While machine learning approaches have shown promise in identifying sepsis risk and improving early detection, much of the work has been focused on ICUs where data availability is extensive [1], [2]. Studies have shown that ML methods, including gradient boosting and deep learning, can outperform conventional clinical tools in predicting sepsis onset [3], [20]. However, several challenges remain. First, many current approaches rely on a limited set of clinical parameters typically available in ICUs, limiting their application to other hospital wards [4], [5], [20]. Second, performance metrics such as the area under the ROC curve (AUROC) often do not reflect the actual clinical utility or the imbalance in data [6], [7]. Third, few studies have considered the timeline of interventions already being initiated (e.g., antibiotics, blood cultures) prior to sepsis onset [8], [9]. Recent research suggests that rich data and deep learning can address these shortcomings by learning complex temporal and feature interactions directly from raw EHR data [20], [21], [22]. For instance, recurrent neural networks and CNNs have been applied to heterogeneous clinical data to capture subtle patterns in patient trajectories [20] and multitask Gaussian Process RNN classifiers have improved accuracy while reducing the need for manual feature engineering [22]. In the field of sepsis detection, these methods have achieved promising results by leveraging not only vital signs and labs but also medication administrations and treatment histories. This paper builds upon these advancements by employing a CNN and LSTM-based architecture on a multi-center dataset that spans beyond the ICU and incorporating a retrospective assessment of interventions.

To diagnose sepsis, doctors often order several tests to pinpoint the underlying infection. Blood tests check for evidence of infection, clotting problems, abnormal liver or kidney function, and electrolyte imbalances. Depending on symptoms, tests may also be performed on urine, wound secretions, and respiratory secretions. Imaging tests such as X-ray, CT, ultrasound, and MRI may be used to locate the infection site. Treatment typically involves early, aggressive measures including antibiotics, IV fluids, and vasopressors, as well as supportive care like oxygen and dialysis. Sometimes surgery is needed to remove infected tissue. While these tests and treatments form the current clinical practice, integrating them into data-driven sepsis prediction models can optimize the timing of interventions. Recent deep learning models have shown that complex temporal patterns of varying measurements can be learned without hard-coded rules [20], [21]. However, challenges remain in adapting these models to non-ICU settings, handling missing data, and evaluating them in a way that ensures real-world clinical utility.

II. EXISTING SYSTEM

Machine learning models trained from individual patient EHRs may be used for the early detection of sepsis. ML models for sepsis detection far exceed the predictive ability of existing clinical early warning systems such as NEWS. However, there are disadvantages. (1) Most studies build their ML models on a limited set of clinical parameters and assume regular measurement intervals, making them less applicable to general hospital wards. (2) Current evaluation practices often rely solely on AUROC, which may not reflect true clinical utility or the impact of data imbalance. (3) The clinical utility of the model is typically not investigated in relation to potential interventions, neglecting the fact that some patients may have already begun receiving treatment by the time the model issues an alert.

III. PROPOSED SYSTEM

In the proposed model we consider all elements in the entire event sequence E for a patient. Unlike previous approaches that rely on a limited number of features, we incorporate events from multiple sources and use a CNN-LSTM architecture to learn complex, hierarchical representations. We consider events present in at least 100 sequences in the training data and group events into five-minute non-overlapping blocks. This temporal aggregation, gap-filling, and context concatenation step preserves the temporal order and ensures robustness to missing data. The advantage of this approach is that it learns from heterogeneous data, requires no labor-intensive feature engineering, and has the potential for scaling to various hospital units. It offers a sequence evaluation approach that provides realistic estimations of model performance and evaluates clinical utility by considering potential interventions like blood cultures and antibiotics at earlier time points.

IV. LIMITATIONS/CHALLENGES

While our proposed system shows promise, there remain limitations and challenges. First, the availability and consistency of EHR data outside ICUs can vary widely, leading to data sparsity and missing values. Second, the model's interpretability remains a challenge, as deep learning approaches often operate as "black boxes" and may be less transparent to clinicians. Third, differences in hospital protocols, patient populations, and local antimicrobial resistance patterns can affect model generalizability and require careful calibration and validation. Fourth, while the model improves early detection, it does not inherently provide clinical decision support in choosing specific interventions. Finally, the retrospective nature of our study means that true prospective validation and real-time integration into clinical workflow remain areas for future work.

V. SYSTEM REQUIREMENTS

Software requirements include Windows OS, Python 3.x and above, Jupyter Notebook, and Anaconda 3.5. Hardware requirements include at least 4GB RAM, Intel i3 processor or above, and a minimum of 500GB hard disk space. These requirements ensure efficient data processing, model training, and result generation.

VI. IMPLEMENTATION

Implementation involves converting the system design into an operational one. Three main types of implementations are considered: (1) Implementing a computer system to replace a manual system, focusing on converting files, training users, and verifying data integrity. (2) Implementing a new computer system to replace an existing one, requiring careful planning to avoid operational disruptions. (3) Implementing a modified application to replace an existing one on the same computer, which is usually simpler. In our context, implementation includes user-level identification, table creation with specified conditions, updating modules for

International Journal of Business Quantitative Economics and Applied Management Research
Volume-6, Issue-06, 2020 ISSN No: 2349-5677

insert/delete actions, and generating reports in 2D or 3D views. These steps provide a comprehensive framework for managing the EHR data and the model's predictions.

VII. OUTPUT SCREENSHOTS

```
import os
from multiprocessing import Pool, cpu_count
import pandas as pd
import numpy as np
import sys
from sys import platform
from IPython.display import display, HTML
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_rows', 500)
import time
from sklearn.preprocessing import normalize
from tqdm import tqdm
from sys import platform
```

```
file =  os.path.join(train_A_path, os.listdir(train_A_path)[0])
subject = pd.read_csv(file, sep = "|")
print(' shape of subject is {}'.format(subject.shape))
subject.head(20)
```

```
shape of subject is (36, 41)
```

	HR	O2Sat	Temp	SBP	MAP	DBP	Resp	EtCO2	BaseExcess	HCO3	FiO2	pH	PaCO2	SaO2	AST	BUN	Alkalin
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	120.5	91.0	NaN	NaN	80.0	NaN	29.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	116.0	95.0	NaN	NaN	81.0	NaN	28.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	114.0	95.5	NaN	NaN	54.0	NaN	23.0	NaN	NaN	NaN	0.7	NaN	NaN	NaN	NaN	NaN	
4	114.0	92.0	NaN	NaN	84.0	NaN	28.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
5	118.0	92.0	35.56	NaN	77.0	NaN	24.0	NaN	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	
6	112.0	94.0	NaN	NaN	NaN	NaN	29.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
7	119.0	92.0	NaN	97.5	79.0	62.0	25.0	NaN	0.0	NaN	NaN	7.49	28.0	NaN	NaN	NaN	
8	120.0	91.0	NaN	92.0	71.0	59.0	32.0	NaN	0.0	NaN	NaN	7.49	NaN	NaN	NaN	NaN	
9	112.0	93.5	37.17	91.0	70.0	59.0	30.5	NaN	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	
10	114.0	90.0	NaN	92.0	72.0	59.0	34.0	NaN	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	
11	110.0	91.0	NaN	94.0	70.0	58.0	29.0	NaN	0.0	20.0	NaN	7.47	32.0	NaN	NaN	49.0	
12	95.0	92.0	NaN	99.0	73.0	60.0	25.0	NaN	0.0	20.0	NaN	7.47	NaN	NaN	NaN	49.0	
13	105.0	92.0	36.22	102.0	77.0	63.0	34.0	NaN	NaN	NaN	1.0	NaN	NaN	NaN	NaN	NaN	
14	101.0	91.0	NaN	113.0	81.0	64.0	26.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
15	103.0	91.0	NaN	106.0	79.0	65.0	30.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
16	100.0	93.0	NaN	130.0	89.0	69.0	23.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
17	106.0	90.0	NaN	105.0	78.0	62.0	24.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
18	96.0	93.0	NaN	116.0	86.0	71.0	35.0	NaN	0.0	NaN	1.0	7.50	26.0	NaN	NaN	NaN	
19	105.0	93.0	NaN	111.0	85.0	71.0	30.0	NaN	0.0	NaN	NaN	7.50	NaN	NaN	NaN	NaN	

International Journal of Business Quantitative Economics and Applied Management Research
Volume-6, Issue-06, 2020 ISSN No: 2349-5677

```
MAIN_DIR = '/content'
TRAIN_DIR= MAIN_DIR +'/training_setA/training/'
TEST_DIR = MAIN_DIR +'/training_setB/training_setB/'
train_files = os.listdir(TRAIN_DIR)
test_files = os.listdir(TEST_DIR)
generate_info()
```

```
n value is ...10168
setA
working in this diretory ..... /content/training_setA/training/
generated the following files.....
['.config', 'training_setA', 'training_setB', 'physionet-challenge-2019early-detection-of-sepsis.zi
p', 'yes_Sepsis_subject_id_setA.csv', '.ipynb_checkpoints', 'no_Sepsis_subject_id_setA.csv', 'info_tr
aining_setA.csv', 'kaggle.json', 'sample_data']
n value is ...10000
setB
working in this diretory ..... /content/training_setB/training_setB/
generated the following files.....
['.config', 'training_setA', 'training_setB', 'no_Sepsis_subject_id_setB.csv', 'physionet-challenge-2
019early-detection-of-sepsis.zip', 'yes_Sepsis_subject_id_setA.csv', '.ipynb_checkpoints', 'no_Sepsis
_subject_id_setA.csv', 'yes_Sepsis_subject_id_setB.csv', 'info_training_setA.csv', 'info_training_set
B.csv', 'kaggle.json', 'sample_data']
```

```
output_path= '/content'
info_setA= pd.read_csv(os.path.join(output_path,'info_training_setA.csv'))
info_setA.head()
```

	subject_id	Age	Gender	Unit1	Unit2	HospAdmTime	nb_samples	SepsisLabel
0	p014784	67.79	0.0	NaN	NaN	-0.02	20.0	0.0
1	p007740	75.04	0.0	0.0	1.0	-21.01	37.0	0.0
2	p013276	54.16	1.0	NaN	NaN	-1.20	35.0	0.0
3	p015491	52.84	1.0	0.0	1.0	-0.04	51.0	0.0
4	p014145	60.77	1.0	0.0	1.0	-0.02	36.0	0.0

```
output_path= '/content'
info_setB= pd.read_csv(os.path.join(output_path,'info_training_setB.csv'))
info_setB.head()
```

	subject_id	Age	Gender	Unit1	Unit2	HospAdmTime	nb_samples	SepsisLabel
0	p113249	48.0	1.0	0.0	1.0	-170.66	51.0	0.0
1	p108740	60.0	1.0	1.0	0.0	-2.15	22.0	0.0
2	p103306	62.0	0.0	1.0	0.0	-105.15	17.0	0.0
3	p119021	25.0	1.0	0.0	1.0	-14.99	38.0	0.0
4	p105995	43.0	0.0	1.0	0.0	-71.10	45.0	0.0

```
df= pd.concat([info_setA,info_setB], axis=0)
df.shape , info_setA.shape, info_setB.shape
```

```
((40336, 8), (20336, 8), (20000, 8))
```

```
info_setB.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 8 columns):
subject_id      20000 non-null object
Age             20000 non-null float64
Gender          20000 non-null float64
Unit1           13905 non-null float64
Unit2           13905 non-null float64
HospAdmTime     20000 non-null float64
nb_samples      20000 non-null float64
SepsisLabel     20000 non-null float64
dtypes: float64(7), object(1)
memory usage: 1.2+ MB
```

```
info_setB.isnull().sum()
```

```
subject_id      0
Age             0
Gender          0
Unit1           0
Unit2           0
HospAdmTime     0
nb_samples      0
SepsisLabel     0
dtype: int64
```

International Journal of Business Quantitative Economics and Applied Management Research
Volume-6, Issue-06, 2020 ISSN No: 2349-5677

```
info_setB['SepsisLabel'].value_counts()
```

```
0.0    18858
1.0     1142
Name: SepsisLabel, dtype: int64
```

```
info_setB['SepsisLabel'].value_counts()
```

```
0.0    18858
1.0     1142
Name: SepsisLabel, dtype: int64
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
```

```
X_train,X_test,y_train,y_test =  train_test_split(X,y,test_size=0.2)
```

```
lr = LogisticRegression()
lr.fit(X_train,y_train)
pred = lr.predict(X_test)
print(accuracy_score(pred,y_test))
```

```
0.943
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/linear_model/logistic.py:432: FutureWarning: Default s
olver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
print(confusion_matrix(pred,y_test))
```

```
[[3757  221]
 [   7   15]]
```

The implementation phase is complemented by output screenshots illustrating user authentication, data retrieval, model training progress, and final predictive analytics results. Such outputs demonstrate the system's functionality, user interface, and the progression from raw input data to clinically interpretable predictions.

VIII. CONCLUSION

1. The proposed deep learning model can detect sepsis at an early stage by learning directly from raw heterogeneous event sequence data.
2. The model achieves strong AUROC performance, outperforming baseline gradient boosting approaches, especially in settings with sparse and missing data.
3. Retrospective assessment shows that earlier detection with the model could facilitate timely interventions such as intravenous antibiotics and blood cultures.
4. The approach can be scaled beyond ICUs to multiple hospital departments, thereby increasing applicability and impact.
5. The proposed deep learning model outperforms baseline models like gradient boosting (GB-Vital) that depend on specific data elements and thus face challenges with missing values.

6. Future directions include integrating interpretability methods, validating in prospective trials, and extending the model to offer more targeted clinical decision support.

REFERENCES

1. G. Canbek and Ù. Sa÷Õro÷lu, "A Review on Information, Information Security and Security Processes," Politek. Derg., vol. 9, no. 3, pp. 165–174, 2006.
2. L. McCluskey, F. Thabtah, and R. M. Mohammad, "Intelligent rule-based phishing websites classification," IET Inf. Secur., vol. 8, no. 3, pp. 153–160, 2014.
3. R. M. Mohammad, F. Thabtah, and L. McCluskey, "Predicting phishing websites based on self-structuring neural network," Neural Comput. Appl., vol. 25, no. 2, pp. 443–458, 2014.
4. R. M. Mohammad, F. Thabtah, and L. McCluskey, "An assessment of features related to phishing websites using an automated technique," Internet Technol. …, pp. 492–497, 2012.
5. W. Hadi, F. Aburub, and S. Alhawari, "A new fast associative classification algorithm for detecting phishing websites," Appl. Soft Comput. J., vol. 48, pp. 729–734, 2016.
6. N. Abdelhamid, "Multi-label rules for phishing classification," Appl. Comput. Informatics, vol. 11, no. 1, pp. 29–46, 2015.
7. N. Sanglerdsinlapachai and A. Rungsawang, "Using domain top-page similarity feature in machine learning-based web phishing detection," in 3rd International Conference on Knowledge Discovery and Data Mining, WKDD 2010, 2010, pp. 187–190.
8. W. D. Yu, S. Nargundkar, and N. Tiruthani, "A phishing vulnerability analysis of web based systems," IEEE Symp. Comput. Commun. (ISCC 2008), pp. 326–331, 2008.
9. P. Ying and D. Xuhua, "Anomaly based web phishing page detection," in Proceedings - Annual Computer Security Applications Conference, ACSAC, 2006, pp. 381–390.
10. M. Moghimi and A. Y. Varjani, "New rule-based phishing detection method," Expert Syst. Appl., vol. 53, pp. 231–242, 2016.
11. DATASET: Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
12. G.-B. Huang et al., "Extreme learning machine: Theory and applications," Neurocomputing, vol. 70, no. 1–3, pp. 489–501, 2006.
13. C. S. Guang-bin Huang, Qin-yu Zhu, "Extreme learning machine: A new learning scheme of feedforward neural networks," Neurocomputing, vol. 70, pp. 489–501, 2006.
14. T. S. Guzella and W. M. Caminhas, "A review of machine learning approaches to Spam filtering," Expert Systems with Applications, vol. 36, no. 7, pp. 10206–10222, 2009.
15. Ö. F. Ertuğrul, AşırıÖğrenmeMakineleriilebiyolojiksinyalleringizlikaynaklarınaayrıştırılması. D.Ü. MühendislikDergisi Cilt:7, 1,3-9-2016.
16. M. E. Tagluk, M. S. Mamiş, M. Arkan, and Ö. F. Ertugrul, "AşiriÖğrenmeMakineleriileEnerjiİletimHatlari Ariza Tipi veYerininTespiti," in 2015 23rd Signal Processing and Communications Applications Conference, SIU 2015 - Proceedings, 2015, pp. 1090–1093.

17. Ö. Faruk Ertuğrul and Y. Kaya, "A detailed analysis on extreme learning machine and novel approaches based on ELM," Am. J. Comput. Sci. Eng., vol. 1, no. 5, pp. 43–50, 2014.
18. Ö. F. Ertugrul, "Forecasting electricity load by a novel recurrent extreme learning machines approach," Int. J. Electr. Power Energy Syst., vol. 78, pp. 429–435, 2016.
19. G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," Neurocomputing, vol. 70, no. 1, pp. 489–501, 2006.
20. Desautels T, Calvert J, Hoffman J, et al. "Prediction of Sepsis in the Intensive Care Unit with Minimal Electronic Health Record Data: A Machine Learning Approach," BMC Medical Informatics and Decision Making, vol. 16, no. 55, 2016.
21. Kamaleswaran R, Akbilgic O. "Predicting sepsis onset using multi-dimensional vital signs time-series embeddings." IEEE Int. Conf. Big Data, 2018, pp. 1202–1212.
22. Futoma J, Hariharan S, Heller K, et al. "Learning to Detect Sepsis with a Multitask Gaussian Process RNN Classifier." In Proc. ICML, 2017.