Final Project

CS 598 Deep Learning for Healthcare

Spring 2023 – Team 4

Paper ID: 186

Jennifer Hajduk jhajduk2@illinois.edu

GitHub: https://github.com/jenniferamhajduk/ConvolutionMedicalNer

Presentation Video: https://youtu.be/_1mAICCceBM

Introduction

The paper discusses the use of Electronic Health Record (EHR) data in predicting clinical outcomes for patients in hospitals or Intensive Care Units (ICU) using deep learning algorithms. The main contribution of this paper is that the authors developed a Convolutional Neural Network model, combined with word vector data from clinical note data, to improve the predictions accuracy of mortality and length of stay predictions. The paper focuses on predicting two common clinical risk factors, mortality and length of stay, and the integration of structured data in EHR with medical entities to improve predictions. The article explores Named Entity Recognition (NER) as a means of extracting medical entities from clinical notes and compares different word embedding methods and techniques for embedding medical entities for use with the patient mortality and length of stay data in their proposed CNN model.

Scope of reproducibility

I plan to get the results data from the baselines used in the paper as a measuring stick for the proposed model and ablations for the proposed model. There have been changes made to the code to make some of the libraries work correctly. The main claims of the paper that I will be studying are a comparison of GRU versus LSTM in the baseline models and the proposed model created by the authors. I will be attempting to get the same results and compare results of the original with my ablation studies. My ablations were:

1. Remove each of the last two convolution layers in the proposed model and compare against the proposed CNN model.
2. Try sigmoid in the convolution layers rather than sigmoid in the proposed model to compare results against the proposed CNN model.
3. Change the filter size for the proposed model to compare results against the proposed CNN model.
4. Decrease the learning rate for the proposed model to compare against the proposed CNN model.
5. Choose LSTM rather than GRU for the proposed model to compare against the proposed CNN model.
6. Write a Focal Loss function to see if improvement was made in the proposed model metrics.

Methodology

I will reuse the code that the authors have given in the paper. This code, however, requires changes to work with modern versions of Keras, Python, and TensorFlow. Each of the notebooks required changes to have the code run successfully. There were also some variables that needed to be corrected in the code. Below are the resources that I used to run the code and descriptions of each of the models that were run.

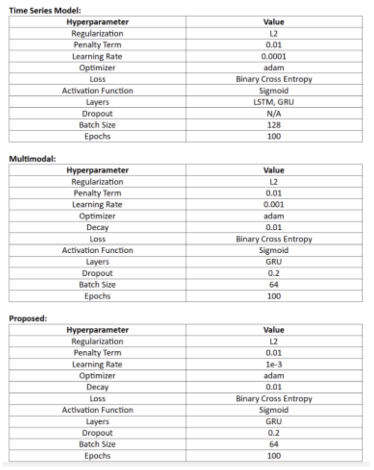
Model Descriptions

The baseline time series uses either LSTM or GRU layers, both of which are recurrent neural networks (RNNs). It takes in a sequence input of shape (24, 104), where 24 represents the number of time steps and 104 represents the number of features at each time step. After passing the sequence input through the LSTM or GRU layer, the output is fed into a dense layer with a sigmoid activation function, which produces a single output value between 0 and 1. The dense layer is regularized using L2 regularization with a regularization strength of 0.01. Finally, the model is compiled with binary cross entropy loss and the Adam optimizer, and accuracy is used as a metric to evaluate the model's performance.

The baseline multimodal mode defines an average NER (Named Entity Recognition) model. The function takes in three arguments: layer_name which specifies the type of RNN layer to use (either "GRU" or "LSTM"), number_of_unit which specifies the number of hidden units in the RNN layer and embedding_name which specifies the type of input embedding to use ("concat" or any other value). The sequence input is passed through the specified RNN layer (GRU or LSTM) and the output is concatenated with the additional input using the Concatenate layer. The concatenated output is then passed through a dense layer with 256 units and ReLU activation, followed by a dropout layer with a rate of 0.2. The output of this layer is fed into a final dense layer with a sigmoid activation function, producing a single output value between 0 and 1. The dense layer is regularized using L2 regularization with a regularization strength of 0.01, and the model is compiled using binary cross entropy loss and the Adam optimizer with a learning rate of 0.001 and a decay rate of 0.01.

The proposed mode defines a model for Named Entity Recognition (NER). The function takes in five arguments: layer_name which specifies the type of RNN layer to use (either "GRU" or "LSTM"), number_of_unit which specifies the number of hidden units in the RNN layer, embedding_name which specifies the type of input embedding to use ("concat" or any other value), ner_limit which specifies the maximum number of entities to consider, and num_filter which specifies the number of filters to use in the convolutional layers. The model takes in two inputs: a sequence input of shape (24,104) and an additional input of shape (ner_limit, input_dimension), where input_dimension is either 100 or 200 depending on the value of embedding_name. The additional input is fed into a series of convolutional layers with different filter sizes, followed by a global max pooling layer to obtain text embeddings. The sequence input is passed through the specified RNN layer (GRU or LSTM) and the output is concatenated with the text embeddings obtained from the convolutional layers. The concatenated output is then passed through a dense layer with 512 units and ReLU activation, followed by a dropout layer with a rate of 0.2. The output of this layer is fed into a final dense layer with a sigmoid activation function, producing a single output value between 0 and 1. The dense layer is regularized using L2 regularization with a regularization strength of 0.01, and the model is compiled using binary cross entropy loss and the Adam optimizer with a learning rate of 1e-3 and a decay rate of 0.01.

Model Architectures:


Time Series Model:

Hyperparameter	Value
Regularization	L2
Penalty Term	0.01
Learning Rate	0.0001
Optimizer	adam
Loss	Binary Cross Entropy
Activation Function	Sigmoid
Layers	LSTM, GRU
Dropout	N/A
Batch Size	128
Epochs	100

Multimodal:

Hyperparameter	Value
Regularization	L2
Penalty Term	0.01
Learning Rate	0.001
Optimizer	adam
Decay	0.01
Loss	Binary Cross Entropy
Activation Function	Sigmoid
Layers	GRU
Dropout	0.2
Batch Size	64
Epochs	100

Proposed:

Hyperparameter	Value
Regularization	L2
Penalty Term	0.01
Learning Rate	1e-3
Optimizer	adam
Decay	0.01
Loss	Binary Cross Entropy
Activation Function	Sigmoid
Layers	GRU
Dropout	0.2
Batch Size	64
Epochs	100

Data Descriptions

The data originates from the MIMIC-III dataset. Special training and credentials are needed to use this dataset. Making use of existing tooling, the authors use MIMIC-Extract, a novel pipeline for working with MIMIC-III data. The data consists of 61,532 ICU admissions from 46,520 patients in the ICU of the Beth Israel Deaconess Medical Center between 2001 and 2012. The final number of patients after processing is 34,472 patients with 104 clinically aggregated time-series variables. Only data concerning patients over age 15 and with at least 30 hours of admission data is used. Data from MIMIC-III includes.

Implementation

The libraries used for the Jupyter notebooks include: Keras, Sci-kit, Tensorflow, numpy, pandas, and gensim. A full list of libraries is available in my repository. Pre-processing includes using MIMIC-Extract to take input from MIMIC-III data from ADMINSSIONS.csv, ICUSTAYS.csv, and NOTEEVENTS.csv. Since the code was a couple of years old, there were updates that needed to be made to work with the current versions of libraries and Python 3.9. Here you can find my fork of the repository with working code.

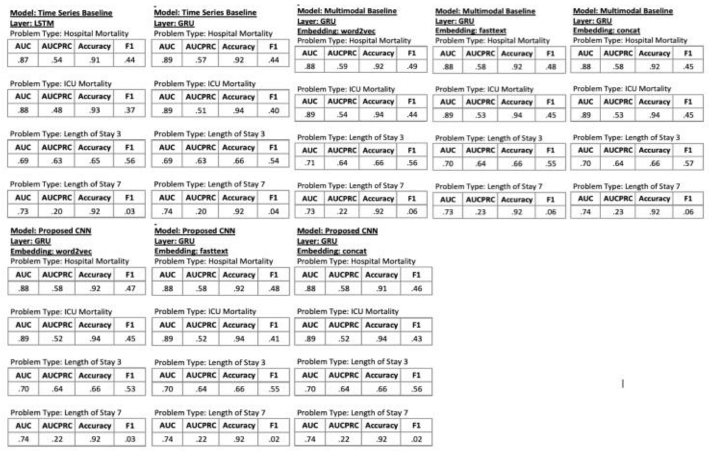
Computational Requirements

With all the data that is produced, it is important to have enough RAM to support the clinical data as well as the processing of it. I was able to use a gaming computer to complete the experiments. Here are the specs:

1. EVGA 3080 TI FTW 3 GPU
2. 24 core AMD Ryzen Threadripper 3960X (overclocked to 4.3 Ghz for each core)
3. 32GB RAM CORSAIR VENGENCE 3600 MHz DDR4
4. Paging file set to 256 GB

Both baseline models as well as the proposed model took between 6-8 hours to run with running the full set of data.

Results



Model: Time Series Baseline
Layer: LSTM

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.87	.54	.91	.44

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.88	.48	.93	.37

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.69	.63	.65	.56

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.73	.20	.92	.03

Model: Time Series Baseline
Layer: GRU

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.89	.57	.92	.44

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.51	.94	.40

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.69	.63	.66	.54

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.74	.20	.92	.04

Model: Multimodal Baseline
Layer: GRU
Embedding: word2vec

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.59	.92	.49

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.54	.94	.44

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.71	.64	.66	.56

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.73	.22	.92	.06

Model: Multimodal Baseline
Layer: GRU
Embedding: fasttext

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.58	.92	.48

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.53	.94	.45

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.70	.64	.66	.55

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.73	.23	.92	.06

Model: Multimodal Baseline
Layer: GRU
Embedding: concat

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.58	.92	.45

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.53	.94	.45

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.70	.64	.66	.57

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.74	.23	.92	.06

Model: Proposed CNN
Layer: GRU
Embedding: word2vec

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.58	.92	.47

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.52	.94	.45

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.70	.64	.66	.53

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.74	.22	.92	.03

Model: Proposed CNN
Layer: GRU
Embedding: fasttext

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.58	.92	.48

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.52	.94	.41

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.70	.64	.66	.55

Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.74	.22	.92	.02

Model: Proposed CNN
Layer: GRU
Embedding: concat

Problem Type: Hospital Mortality

AUC	AUCPRC	Accuracy	F1
.88	.58	.91	.46

Problem Type: ICU Mortality

AUC	AUCPRC	Accuracy	F1
.89	.52	.94	.43

Problem Type: Length of Stay 3

AUC	AUCPRC	Accuracy	F1
.70	.64	.66	.56

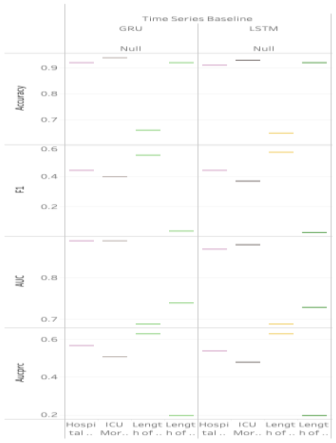
Problem Type: Length of Stay 7

AUC	AUCPRC	Accuracy	F1
.74	.22	.92	.02

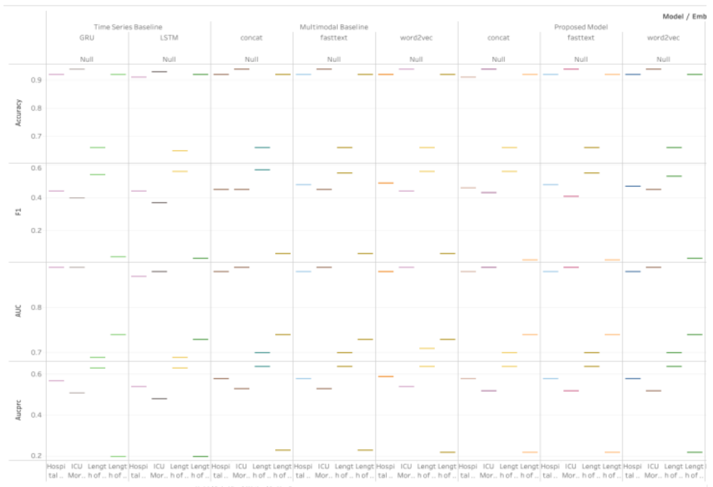
Here are the results for all models and ablation studies. Results are measured with AUC, AUCPRC, Accuracy, and F1 scores. Here is a tableau chart of the results for easier analysis.

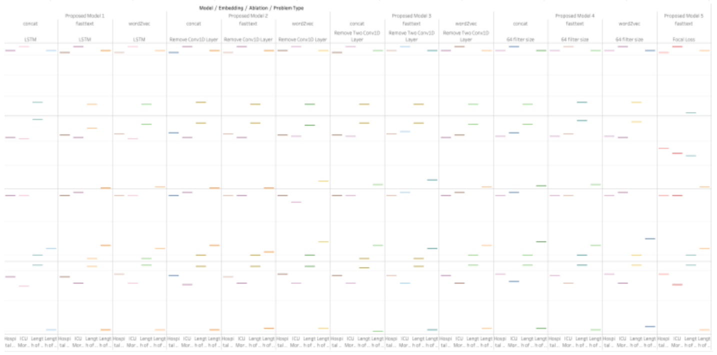
Examining the baseline and proposed model results, all models do a better job at hospital and ICU mortality problem types than length of stay. This could be due to the difficulty in capturing various features such as the severity of illness, comorbidities, and the effectiveness of the treatment provided for length of stay outcomes. The AUC and accuracy scores are similar for the time series with GRU RNN layer, the baseline multimodal models, and the proposed CNN model. The F1 scores are generally low across all models and problem types because, as the authors admit, the data is imbalanced. The choice of embedding (word2vec, fasttext, or concat) does not seem to have a significant impact on the performance of the models. The proposed CNN model, which uses a combination of convolutional and recurrent neural networks, shows promising results, but it does not outperform the baseline models (except for the baseline time series LSTM model).

A claim in the paper is that GRU performs better than LSTM. This is substantiated by some of the results from the time series baseline. This is a chart for the results among all four problem types for both GRU and LSTM in the baseline model.



The claim in the paper that the proposed model performs better than the baseline models not supported by the results. It appears that the baseline multimodal model performs similarly or better in certain problem types than the proposed model. The exception is the AUC for the length of stay 7 days problem type where the proposed model appears to do better than the baseline multimodal model. This could be due to several factors. There may not be enough data to showcase the differences in results between the two models or that the proposed model is not significantly different from the baseline multimodal model.





Discussion

Understanding the paper was fun and interesting. I found it very closely related to the material that we learned about in the course. Topics such as RNNs, CNNs, and embeddings proved to be helpful as these were the main points presented in the paper. Designing good ablation studies was new for me and took some time to implement. This is mainly due to the runtime of the models, which took ~8 hours per model to implement. It was difficult to refactor the code and understand new libraries or modules that could take the place of the outdated ones. It took a significant chunk of time to ensure that the new modules performed the same operations that the original ones did, considering I was new to designing neural networks. In the end, I am encouraged to try to implement different papers from the paper pool to get experience with creating and designing different AI systems. I was surprised that the focal loss function performed so much worse than the other models, but this goes to support the authors architecture of their proposed model. It is possible that BCE loss is the os optimized function for the model.

References

Authors: Batuhan Bardak and Mehmet Tan

Paper: Improving clinical outcome predictions using convolution over medical entities with multimodal learning 0933-3657/© 2021 Elsevier B.V. All rights reserved.

Authors Repository: https://github.com/tanlab/ConvolutionMedicalNer.git

MIMIC-Extract: https://github.com/MLforHealth/MIMIC_Extract

Pre-Trained word2vec and fastText embeddings: https://github.com/kexinhuang12345/clinicalBERT

Med7: https://github.com/kormilitzin/med7