Predicting Patient Outcomes from Clinical Notes Using Deep Learning
CS598 DL4H Final Project Report

Edward Zhao
etz3@illinois.edu

Paper ID: 162
Code link: https://github.com/zhao852/clinical-notes-prediction-dl
Notebook link: https://colab.research.google.com/drive/1-tzdeEnnWX48yJ3F26uN37IqUEdAj20q#scrollTo=BR5cSn6L-_Ys

1 Introduction

As the modern era of technology has progressed, so has the use of Electronic Health Records (EHRs) to store people's medical records. EHRs contain information such as a patient's diagnosis, symptoms, performed procedures, and text notes. This data describes a patient's clinical trajectory, which is their sequence of clinical events through time. This is because every time a patient sees a doctor, they have a new visit record with a new timestamp.

Within electronic health records are the free-text clinical notes, written by the healthcare professionals. This kind of free-text data is unstructured and has many ambiguities, redundancies, and non-obvious semantics, making it more challenging to process and compute. For example, there are often acronyms used by healthcare professionals. These clinical notes can be very brief or very long, in a formal structure, or in brief bullet points. Usually, they have many grammatical errors and typos. All these work against any computational approach.

The problem the paper (Zaghir et al., 2021) aims to solve is to improve personalized healthcare by applying deep learning and natural language processing on free-text clinical notes to predict the most likely medical problems to follow a patient's last admission. The three areas of prediction explored in the paper are mortality prediction, diagnosis codes prediction, and readmission prediction. The overall goal of the paper is to use the clinical notes free text in a patient's EHR to predict their clinical trajectory.

2 Scope of reproducibility

This research aims to reproduce the experiments conducted in the original paper that proposed a deep learning approach to predict readmission based on free-text clinical notes using the MIMIC-III dataset. Specifically, I focus on verifying the hypothesis that feed-forward networks perform better than recurrent neural networks, given the weak temporal dependencies in the MIMIC-III dataset. I will also explore the impact of different optimization algorithms and adding a bidirectional GRU on admission prediction performance.

To achieve my goals:

1. I will first replicate the experiments conducted in the original paper using a sample dataset and evaluation metrics.

2. I will then implement the ablation studies that explore the impact of different optimization algorithms and adding a bidirectional GRU on performance.

3. Then I will evaluate the performance of the reproduced models using standard performance metrics and compare the results with those reported in the original paper.

4. I will also validate the assumption about the weak temporal dependencies in the MIMIC-III dataset by comparing the performance of different network architectures and the results with those reported in other studies.

The results of this study will provide insights into the effectiveness of deep learning models in predicting admission based on free-text clinical notes and the impact of different network architectures on performance. It will also help to validate the reproducibility of the original paper's findings and provide a basis for further research in this area.

2.1 Addressed claims from the original paper

The claim I are testing is the readmissions prediction part of the paper. I are testing the probability of a patient's readmission following their most recent visit based on their healthcare professional's free text clinical notes. I plan to verify the following hypotheses: Feed-Forward Networks (FFNs) perform better than Recurrent Neural Networks (RNNs) in predicting patient trajectories, given the weak temporal dependencies in the MIMIC-III dataset.

3 Methodology

The first step is to preprocess a large dataset of real-world clinical notes using natural language processing techniques. I will randomly select a 45% sample of the MIMIC-III dataset with the same admission distribution for my study. After preprocessing, I will train different deep-learning models to predict the patient's readmission and evaluate the results.

3.1 Model descriptions

The first model architecture used in the paper that I will test with is the feed-forward architecture, where information flows in a forward direction from input to output without any feedback loop. The input data is fed into the network, processed through a series of layers of neurons, and finally produces the output. Since each layer of neurons in the network is connected to the next layer in a unidirectional manner, the output from one layer is only passed on to the next layer and not back to the previous one. For the readmission prediction, only the output layer is altered which demands retraining.

The second model architecture I will test with the Gated Recurrent Unit (GRU). The GRU uses a gating mechanism to control the flow of information through the network which allows it to selectively update and forget information from the previous hidden state to flow into the current hidden state. This makes it effective for tasks that involve long-term dependencies such as language modeling.

In addition to the two model architectures mentioned above that were used in the paper, I will explore the impact of using different optimization algorithms, such as Adagrad or RMSprop, on the performance of the models. Different optimization algorithms have different convergence properties and may be more suitable for the dataset.

By comparing the performance of these different model architectures and optimization algorithms, I aim to evaluate the effectiveness of deep learning models in predicting admission based on free-text clinical notes and understand the impact of these variations on performance. This will help to validate the reproducibility of the original paper's findings and provide a basis for further research in this area.

3.2 Data descriptions

After preprocessing the free text clinical notes, the concept extraction takes place and maps the text to Clinical Classification Software codes. The notesevents.csv file contains the clinical notes. The admissions.csv file contains data on each visit by a patient and includes their admit and discharge times, admission type (emergency, elective, etc.), and their diagnosis. The diagnoses_icd.csv file contains the patient's corresponding ICD9 codes. These files are joined by the SUBJECT_ID, each patient's unique id.

I are also taking a 45% sample of the data due to computational restrictions. To ensure the replication of data distribution, I sampled the data by considering the percentage of patients with fewer than two visits and those with more than two visits. The project used the MIMIC-III (Medical Information Mart for Intensive Care) dataset, a large, openly available dataset of de-identified health-related data associated with over 2M clinical notes. After sampling, I reduce the number of notes to 800K with 15594 patients with less than two admissions and 3014 with at least two admissions (Figure 1).

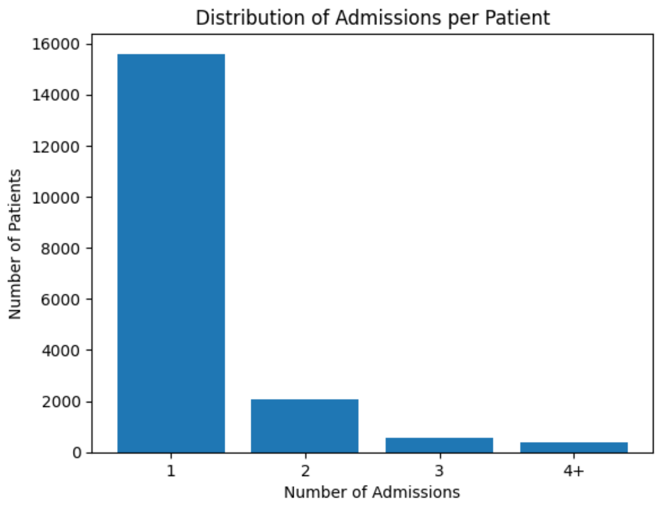


Figure 1: Admission Distribution by Patient

3.3 Hyperparameters

I evaluated the models using the default paper's code hyperparameters, then introduced new optimizers for the GRU network.

Feed forward network (FFN) parameters:

- `hiddenDimSize`: hidden layer size, default value is `50`.
- `batchSize`: the batch size, default value is `100`.
- `nEpochs`: training iterations, default value is `100`.
- `lr`: the learning rate, default value is `0.001`.
- `dropOut`:dropout, default value is `0.5`.

Gated recurrent units (GRU) parameters:

- `hiddenDimSize`: GRU hidden layer size, default value is `200`.
- `numLayers`: GRU layers, default value is `1`.
- `batchSize`: the batch size, default value is `10`.
- `nEpochs`: training iterations, default value is `100`.
- `lr`: the learning rate, default value is `0.01`.
- `dropOut`:dropout, default value is `0.5`.

3.4 Implementation

I started the implementation by forking the original code from the GitHub repository, and I recognized improvements were necessary to enhance the model's performance and adapt it to my requirements.

First, I addressed bugs in the code to ensure smooth execution and prevent issues during the preprocessing and training phases. Next, I introduced training data sampling to the codebase, taking into consideration the patient admission distribution to have a valid test for the admission prediction. The sampling allowed for faster processing and model training while maintaining the original data distribution.

Also, I updated the code to store and read data from external files rather than working directly under the Git repository to facilitate data management and keep the repository clean and organized. Also, I had to generate a version from the model training files to support CPUs while waiting for Amazon AWS to approve my GPU access. This enabled us to start running the models until I could access GPU instances.

I also improved the GRU neural network training for admission prediction by updating the code to support a variety of optimizers for performance validation, including Adagrad, RMSprop, and SGD, in addition to Adam in the origin code, providing more flexibility in the model training process.

```
if ARGS.optimizer == "adam":
    optimizer = optim.Adam(..)
elif ARGS.optimizer == "adagrad":
    optimizer = Adagrad(..)
elif ARGS.optimizer == "rmsprop":
    optimizer = RMSprop(..)
elif ARGS.optimizer == "sgd":
    optimizer = SGD(..)
```

3.5 Computational requirements

During the computational tasks of this project, I utilized multiple instances based on the specific needs of each task. To download the necessary code and data and pre-process the CSV file for "NOTEEVENTS.csv," a t2.xlarge instance was used with four vCPUs and 16GB of memory.

For the CPU-intensive task of extracting the QuickUMLS Unique Identifier (TUI) from the clinical notes and referencing large UML files by QuickUMLS, a c6i.8xlarge instance was used. This instance has 32 vCPUs and 64GB of memory. Due to the lengthy duration of this task, which can take several hours to several days, the swap memory was increased from the default 4GB to 10GB to ensure optimal performance.

The server resources were fully occupied during this process which impacted the performance due to the filled memory (Figure 2).

For the training part, I used AWS g5.xlarge which comes with 1 GPU and 4 CPUs. This allowed us to utilize GPU acceleration to improve training time.

4 Results

I trained several neural network models and optimization algorithms on MIMIC-III dataset clinical notes. I evaluated the models using ROC and AUC scores.

4.1 Results analyses

My findings (Table 1) are consistent with the original paper's results. The Feed-Forward Network (FFN) performed better than GRU, and bidirectional GRU, which suggests that the MIMIC-III dataset has weak temporal dependencies between admissions. However the proposed bidirectional GRU improvement results were close to the FFN.

I proposed the optimization algorithm options as an improvement to the original paper that helped

Model	ROC Score	Mean AUC
FFN SGD	0.6861	0.6682
GRU Adam	0.6418	0.6205
GRU Adagrad	0.6534	0.6492
GRU SGD	0.6405	0.6546
GRU RMSprop	0.6297	0.6429
Bi-GRU Adam	0.6739	0.650

Table 1: ROC and AUC re-produced scores from the sample data.

to improve GRU performance. Adagrad optimizer performed the best, followed by the SGD optimizer. However, none of the optimization algorithms significantly improved the GRU performance compared to the FFN model. The results show the importance of the dataset's characteristics, model choice, and optimization algorithms in achieving optimal performance.

4.2 Plans

I have completed all the paper experiments that I intend to do. My results are described above are consistent with the paper's results. I can introduce new techniques to fine-tune the medical code extraction from the doctor's notes, like applying large language models for future improvement. However, this comes with a cost due to the notes file size.

4.3 Additional results not present in the original paper

I conducted additional experiments to explore the impact of bidirectional GRU and different optimization algorithms on the performance of models trained on the MIMIC-III dataset. The results (Table 1) revealed the bidirectional GRU (Bi-GRU) improved the ROC score by 5% compared to the unidirectional GRU used in the original paper. Furthermore, I tested different optimization and found that Adagrad optimizer performed the best, followed by the SGD optimizer.

5 Discussion

Improving the prediction from clinical notes is of great interest for us to choose (Zaghir et al., 2021), as it can improve the efficiency of medical systems. One of the strengths of this paper is that the authors published their code with clear instructions.

I successfully reproduced the results, although I faced challenges with the massive data size and some code bugs. I added a code to sample the data to address this issue while respecting the original data distribution, and I fixed some code issues to run on the server. My findings matched the author's hypothesis that simple feed-forward network networks had better results than recurrent neural networks, given the weak temporal dependencies in the MIMIC-III dataset, which suggests that data type is essential to influence the choice of neural networks. For the ablation study, I had two ideas: one to improve the network itself and the other to improve the extraction of medical codes. Due to the enormous data size, focusing on improving the network was more practical. My experiments showed that introducing new optimizers and using bi-directional GRU layers improved the model's performance.

Overall, this paper provides valuable insights into predicting patient outcomes from clinical notes; the optimizers and bi-directional GRU layers helped improve the results.

I have also shared the link to the google colab notebook at the top of the report on the first page.

5.1 What was easy

One of the strengths of this research was the Github project instructions clarity and the clean code documentation, which made it smooth for us to reproduce the results. Another aspect was the precise organization of the paper itself. The authors provided a detailed description of the dataset, the used methods, and the evaluation metrics.

5.2 What was difficult

Dealing with the data size was the most challenging aspect, specifically the medical notes, which consumed all the server resources for pre-processing and took a long time to complete. To overcome this limitation, I sampled the data while preserving the same visits per patient distribution as the original data.

Additionally, I encountered difficulties with the third-party processing tool used by the authors, QuickUMLS, which required access to the Metathesaurus Unified Medical Language System (UMLS) license from the National Library of Medicine. While I had planned for access to the MIMIC-III data from the beginning, the need for this UMLS access needed to be clarified, and I had to spend additional time and effort obtaining it.

5.3 Recommendations for reproducibility

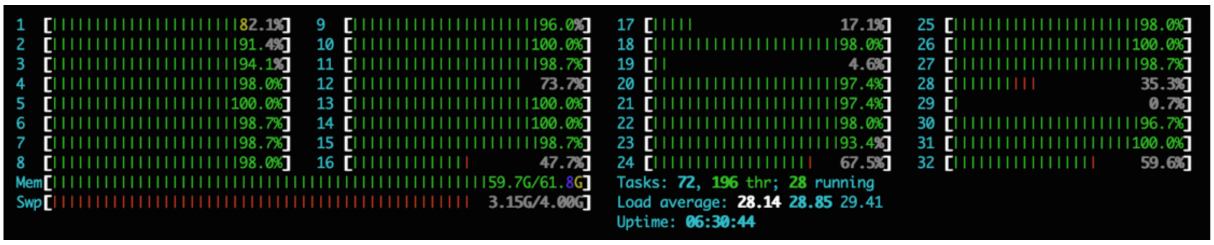
Creating a docker container with the necessary dependencies and libraries can ensure the code runs consistently across different environments, improving reproducibility.

6 Communication with original authors

I contacted one of the co-authors of the reviewed paper ”Jamil Zaghir” for feedback, and the responses quoted here: ”I appreciate your efforts to reproduce the experiments in the paper, improve the code, and get results (that align with the original paper). Your new contribution to BiGRU demonstrates the importance of bidirectionality over GRU, although it still falls short of the performance of the FFN approach. Nevertheless, it is an interesting discovery that adds to the body of knowledge shared in the article. I’m happy to answer questions if you have any. Keep up the good work!”

References

Jamil Zaghir, Jose F Rodrigues-Jr, Lorraine Goeuriot, and Sihem Amer-Yahia. 2021. Real-world patient trajectory prediction from clinical notes using artificial neural networks and umls-based extraction of concepts. *Journal of Healthcare Informatics Research*, 5(4):474–496.



```
1  [||||||||||||||||||||||82.1%]   9  [||||||||||||||||||||||96.0%]   17 [|||||                17.1%]   25 [||||||||||||||||||||||98.0%]
2  [||||||||||||||||||||||91.4%]   10 [|||||||||||||||||||||100.0%]   18 [||||||||||||||||||||||98.0%]   26 [|||||||||||||||||||||100.0%]
3  [||||||||||||||||||||||94.1%]   11 [||||||||||||||||||||||98.7%]   19 [||                    4.6%]   27 [||||||||||||||||||||||98.7%]
4  [||||||||||||||||||||||98.0%]   12 [||||||||||||||||||    73.7%]   20 [||||||||||||||||||||||97.4%]   28 [|||||||||||          35.3%]
5  [|||||||||||||||||||||100.0%]   13 [|||||||||||||||||||||100.0%]   21 [||||||||||||||||||||||97.4%]   29 [|                     0.7%]
6  [||||||||||||||||||||||98.7%]   14 [|||||||||||||||||||||100.0%]   22 [||||||||||||||||||||||98.0%]   30 [||||||||||||||||||||||96.7%]
7  [||||||||||||||||||||||98.7%]   15 [||||||||||||||||||||||98.7%]   23 [||||||||||||||||||||||93.4%]   31 [|||||||||||||||||||||100.0%]
8  [||||||||||||||||||||||98.0%]   16 [||||||||||||||       47.7%]   24 [|||||||||||||||||||   67.5%]   32 [||||||||||||||||      59.6%]
Mem[||||||||||||||||||||||||||||||||||||||||||||||||||59.7G/61.8G]   Tasks: 72, 196 thr; 28 running
Swp[||||||||||||||||||||||||||||||||||||||||||||||    3.15G/4.00G]   Load average: 28.14 28.85 29.41
                                                                     Uptime: 06:30:44
```

Figure 2: The reserved EC2 resources for the data preprocessing show the vast amount of notes and the extensive process of extracting the medical codes.