Length of Stay Prediction Using Diagnosis Notes in MIMIC-III

Isha Kanani
The University of Texas at Austin
ishakanani@utexas.edu

Ting Pan
The University of Texas at Austin
ting.pan@utexas.edu

Yuntao Lu
The University of Texas at Austin
yuntaolu@utexas.edu

Abstract

MIMIC-III (Medical Information Mart for Intensive Care III) is a large, freely-available database comprising deidentified health-related data [6] including thousands of patient information in and out of hospitals.

This project aims to predicting the Length of Stay of patients according to their notes from caregivers in MIMIC III. We added text notes data to Harutyunyan et al.'s [4] benchmark dataset to change the Length of Stay prediction using Standard Long Short Term Memory (LSTM) networks. In order to add the text data as a parameter, Latent Dirichlet Allocation is used to allocate a topic number to the corresponding text data along with the confidence score. Otherwise, to reduce the time consuming and work difficulty, we processed data and built a model based on BERT (Bidirectional Encoder Representations from Transformers) as well.

1. Introduction

MIMIC-III (Medical Information Mart for Intensive Care III) is a large, freely-available database comprising deidentified health-related data [6] including thousands of patient information in and out of hospitals. MIMIC-III dataset has provided a lot of clinical details for machine learning projects. Since 2016, many researchers, such as [7], [8], and [1] have used the dataset to build different algorithms that are helpful to use Artificial Intelligence in the healthcare industry. The MIMIC III dataset also holds the notes taken for each patient. The notes are recorded in English text, which can be used to get more insights or make better predictions. Length of stay prediction is an important attribute in the healthcare research given its benefits. If a proper Length of Stay prediction is provided, the hospital can predict what kind of equipment and in which amount will it be needing in the next few days.

Motivation Harutyunyan et. al. [4] use different regression and LSTM models [5] to predict the length of stay. Their work builds a benchmark dataset by using tables "CHARTEVENTS.csv" and "LABEVENTS.csv" from the MIMIC III dataset. Following their work using standard LSTM, Harutyunyan et al. were able to achieve the Cohen kappa score of 0.46. The Cohen kappa score here is shown to evaluate the model's performance.

The MIMIC III dataset has a table "NOTEEVENTS.csv", which stores text inputs for each patient. The text field provides a text data, which can be added to the dataset developed by Harutyunyan et al. Figure 1 introduces the research objective, which is to add the notes data to Harutyunyan's dataset and compare the algorithm performance.

In order to add the text data to the benchmark dataset, the text data is classified into 30 different topics using the Latent Dirichlet Allocation method [2].

In addition, Bidirectional Encoder Representations from Transformers (BERT) [3] provides pretrained Natural Language Processing (NLP) models developed by GOOGLE in 2018. As the time expense in conducting a model, we used BERT to transform a pretrained model on diagnosis notes data to reduce the barrier and time on processing text features.

2. Experimental Design

In order to reach the project objective, the process is divided into four steps. The figure 2 demonstrates the experimental design. The first step is to imitate Harutyunyan et. al.'s work and get the benchmark dataset. The second step is to classify text from noteevents.csv into 30 different categories. The third step is to combine data from step 1 and 2, and train a standard LSTM and compare the Cohen kappa score to the author's work. The last step attempts to build a model efficiently and easily using BERT pretrained model.

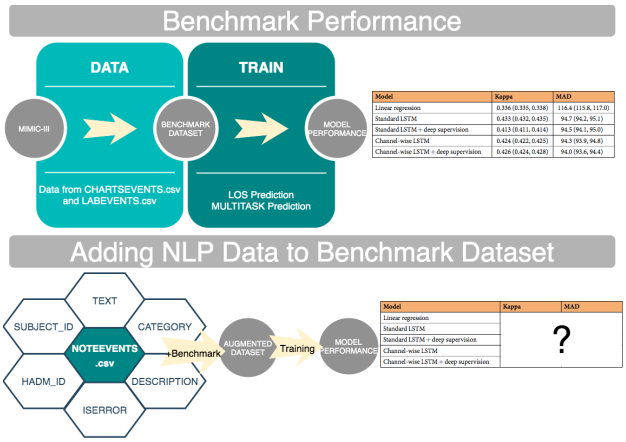

Figure 1. Research Objective

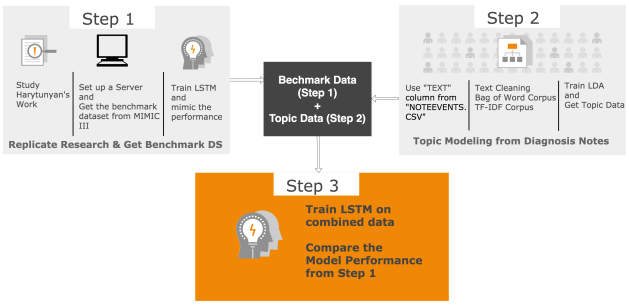

Figure 2. Experimental Design Diagram.

2.1. Step 1: Replicate Research and Get Benchmark Dataset

The first step is to replicate Harutyunyan's work. The author's work builds a benchmark dataset by using the MIMIC III dataset. The author uses different python scripts to process the original data and store it int the newer version which can be fed to the machine learning model. Since the MIMIC III dataset is not available to anyone without certification, the author has not provided the dataset.

In order to get the benchmark dataset and get the data processing done, Amazon Lightsail server is used in this research to get more processing power and more memory. A server instance with 8 virtual CPUs and 32 GB memory was used throughout the project to run python scripts. Once the server instance was launched, the original MIMIC III dataset was moved to the server, and then the instance environment was updated to meet the research requirements. Later, the steps provided on the benchmark work's GitHub page. Once the benchmark dataset is achieved, the next step is to run the standard LSTM on the dataset and get the model performance. The goal is to achieve the model performance similar to the author's work so that it can be compared later in Step 3.

2.2. Step 2: Topic Modeling on Diagnosis Notes

The second step is to get the text data in a format that can be added to the benchmark dataset. The text data is provided in the "NOTEEVENTS.csv" table in the MIMIC III dataset. The text data from this table is processed into a pandas data frame and then preprocessed using nltk and gensim packages. The preprocessed text inputs are then converted into a bag of words corpus and TF-IDF corpus which can be used as a training input for the LDA model. The LDA model provides 30 different topics by reading the data and later predicts all the text inputs into one of the 30 topics, along with providing a confidence score for the same.

By the end of this step, we have classified all the text into 30 topics. The topic number will be later used as a parameter to be added to the benchmark dataset. This step will also provide a trained LDA model which can be later loaded to predict topic related to new text data if needed.

2.3. Step 3: LSTM Performance on Combined Data

This step involves combining the data from the previous two steps. The first step provides the benchmark dataset used by Harutyunyan et. al., seven different python scripts are run to achieve that dataset. Similar scripts will be needed to run on the data from step 2 in order to add it to the benchmark dataset.

Once the new dataset is achieved, standard LSTM is trained with the dataset, and the cohen kappa score will be compared to the cohen kappa score from step 1. The comparison will show is adding the text data improved the algorithm performance or not.

2.4. Step 4: Preprocess and Build a model with BERT

The step integrates training features and predicted label together from MIMIC-III database as well as converts data types which meet the requirements of BERT model. In this step, we hope to conduct a model quickly and easily.The classifier predicts Length of Stay of each patients into 10 categories ranging from 0 to 10 which represents less than 1 day and 10 days stayed in the ICU of hospitals with diagnosis notes from MIMIC-III NOTEEVENTS and ICUSTAYS tables.

In order to feed text features into the pretrained BERT model, we convert the text feature of diagnosis notes to word tokens and encode tokens by mapping each token with a index of a word vocabulary. All procedures of processing input text features are combined in a function of BERT Tokenizer based on a BERT provided word vocabulary. Afterwards, we construct a BERT pretrained model for the se-

```
Mean absolute deviation (MAD) = 126.94910682287501
Mean squared error (MSE) = 46392.70128879119
Mean absolute percentage error (MAPE) = 278.86277021948933
Cohen kappa score = 0.4561062567818349

==>predicting on validation


Custom bins confusion matrix:
[[1797  175    5    0    0    0    0    0   87   68]
 [ 882  235    2    0    0    0    0    0  142  110]
 [ 441  167    4    0    0    0    0    0  190   83]
 [ 252   95    1    0    0    0    0    0  176   85]
 [ 111   74    4    0    0    0    0    0  134   70]
 [  82   47    0    0    0    0    0    0  127   86]
 [  74   51    0    0    0    0    0    0  106   60]
 [  43   26    1    0    0    0    0    0   94   56]
 [  96  123    1    0    0    0    0    0  325  314]
 [  82   80    5    0    0    0    0    0  302  429]]
Mean absolute deviation (MAD) = 128.525328753875
Mean squared error (MSE) = 43730.182944112974
Mean absolute percentage error (MAPE) = 296.9832684579447
Cohen kappa score = 0.44639547676521674
```

Figure 3. Step 1 results

quences classification task and training the model on notes data.

3. Experimental Results

Step 1 Results: Amazon's lightsail instance was set up with all the requirements, and the benchmark dataset was achieved from the MIMIC III dataset. Using this dataset, standard LSTM was implemented and the model performance can be seen in figure 3. The achieved Cohan kappa score is 0.45, which is very close to Harutyunyan's model performance. Since the model and data used is the same, it can be said that research replication was successful.

Step 2 Results: Implementing topic modeling had major complications dealing with the server's memory size. Since the server took a long time compiling the entire code, the code was broken into two chunks. The first chunk of the code included data preprocessing, generating a dictionary, a bag of word corpus, and a TFIDF corpus. This chunk is compiled on the local computer since it gave the results quicker than the server for this step. The dictionary and corpus are saved into gensim dictionary and pickle formats respectively and then transferred onto the server. The second chunk of the code includes compiling the model training task onto the server. In this step, the model is fed the pre-stored dictionary and corpus and the trained model is saved at the end of the code for predictions later.

Loading the saved TFIDF model

```
saved_model_tfidf = gensim.models.LdaMulticore.load("500k/tfidf_lda_500k.model")
```

```
for idx, topic in saved_model_tfidf.print_topics(-1):
    print('TFIDF Topic: {} Word: {}'.format(idx, topic))
```

```
TFIDF Topic: 0 Word: 0.006*"assess" + 0.005*"pain" + 0.005*"line" + 0.005*"tube" + 0.005*"plan" + 0.005*"medic" + 0.005*"total" + 0.005*"order" + 0.004*"hour" + 0.004*"fluid"
TFIDF Topic: 1 Word: 0.008*"wave" + 0.007*"interpret" + 0.007*"trace" + 0.006*"tablet" + 0.005*"namei" + 0.005*"previous" + 0.005*"order" + 0.005*"offic" + 0.005*"physician" + 0.005*"diagnost"
TFIDF Topic: 2 Word: 0.011*"trace" + 0.010*"wave" + 0.009*"comparison" + 0.008*"avail" + 0.008*"normal" + 0.008*"previous" + 0.007*"valv" + 0.006*"lead" + 0.006*"rhythm" + 0.006*"abnorm"
TFIDF Topic: 3 Word: 0.009*"valv" + 0.006*"normal" + 0.006*"mild" + 0.006*"aortic" + 0.006*"ventricular" + 0.005*"leav" + 0.005*"mitral" + 0.005*"assess" + 0.005*"leaflet" + 0.005*"pain"
TFIDF Topic: 4 Word: 0.014*"trace" + 0.012*"atrial" + 0.010*"nonspecif" + 0.009*"previous" + 0.009*"fibril" + 0.009*"wave" + 0.008*"ectopi" + 0.008*"rhythm" + 0.006*"compar" + 0.006*"respons"
TFIDF Topic: 5 Word: 0.009*"assess" + 0.009*"sound" + 0.009*"lung" + 0.008*"ventil" + 0.007*"cuff" + 0.007*"continu" + 0.006*"action" + 0.006*"airway" + 0.006*"plan" + 0.006*"respons"
TFIDF Topic: 6 Word: 0.007*"assess" + 0.006*"medic" + 0.005*"total" + 0.005*"puls" + 0.005*"balanc" + 0.005*"line" + 0.005*"hour" + 0.005*"code" + 0.005*"respiratori" + 0.005*"lab"
TFIDF Topic: 7 Word: 0.008*"tablet" + 0.005*"patient" + 0.005*"hospit" + 0.005*"discharg" + 0.005*"continu" + 0.005*"assess" + 0.005*"pain" + 0.004*"daili" + 0.004*"medic" + 0.004*"blood"
TFIDF Topic: 8 Word: 0.016*"action" + 0.013*"respons" + 0.012*"assess" + 0.011*"plan" + 0.008*"failur" + 0.007*"acut" + 0.006*"pain" + 0.006*"continu" + 0.006*"renal" + 0.005*"cont"
TFIDF Topic: 9 Word: 0.007*"assess" + 0.006*"continu" + 0.005*"trace" + 0.005*"hour" + 0.005*"renal" + 0.005*"medic" + 0.005*"failur" + 0.005*"tachycardia" + 0.005*"line" + 0.004*"plan"
```

Figure 4. Step 2 results

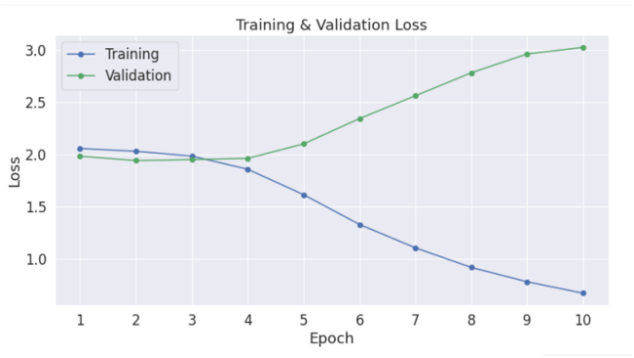

Figure 5. Step 4 Loss of training and validation datasets

After trying different servers, the challenges limited the amount of data to be used for training purposes. Instead of the original 2.3M text inputs, the model here is trained on 500K inputs due to memory limitations. The saved model is used to predict the topic for each of the text rows, using 30 different topics. The first 10 topics' words can be seen in figure 4. At the end of this step, topic modeling is achieved for 500K rows instead of the original size, and the topic data is ready for the next step.

Step 3 Results: Due to the time and processing limitations, a lot of effort was spent on completing step 2. The results for the final step are not available since the task could not be performed.

Step 4 Results: After we conduct 10 training epochs on diagnosis notes data, the loss results on training and validation sets (2000 samples of entire notes data) demonstrate in Figure 5. From the Figure 5 illustrates after the fourth epoch the model is overfitting. For test set, the prediction accuracy is only 0.3 which implies the performance of this pretrained classifier doesn't work well and need to modify the word vocabulary and pretrained model for longer text sequences.

4. Challenges

Large data processing: The first challenge we have is how to deal with a large dataset. MIMIC III Clinical Dataset[6] contains 26 tables and its total size is larger than 46GB. It will require lots of storage space and computing capacity for our experiment which exceeds the performance of our local machine. To cope with this situation, we learn how to use AWS and train our model using a remote server[9].

Server Limitations: The server instance used in this project had a memory capacity of 32 GB and processing capacity of 8 virtual CPUs, however developing the benchmark dataset in step 1 consumed a lot of time. While the model performance in step 1 was replicated from the original work, the model crashed several times while going through training iterations. For step 2, the server instance could not process the entire code together. The training data was attempted to be fed in chunks as well, however the memory allocation could not be understood well and the training data had to be minimized.

Time Limitations: The project timeline could not be followed well, and brainstorming sessions could not be arranged to take further decisions about how to approach the project. The time allocation for step 2 was underestimated and resulted in an incomplete final step.

5. Future Work

Currently, we have implemented methods from Harutyunyan et. al. [4] and have been able to replicate the benchmark performance using the benchmark dataset. Topic modeling was successfully implemented to convert the text input into a topic input as well. Due to time restrictions, the final step of adding the topic data to the benchmark data and comparing the new model performance could not be completed. For future work, the topic data can be added and standard LSTM can be trained to see whether the model performance improves or not.

MIMIC III Clinical Dataset[6] has lots of tables and they are interconnected. From our implementation here, we implemented all steps following specific instructions, and our study was limited to adding only one new table "NOTEEVENTS.csv". Data from other tables can also be processed and added to Harutyunyan's work to improve the length of stay prediction.

6. Conclusion

The current project provides a method for using the notes dataset by implementing topic modeling with latent dirichlet allocation. An accurate Length of stay prediction can be beneficial to researhers, doctors, and patients. Using the MIMIC III dataset to explore the model performance with different parameters can bring new insights to different model evaluations.

Code Availability: The GitHub repository can be found at: https://github.com/IshaKanani/LengthOfStay

References

[1] T. Baumel, J. Nassour-Kassis, R. Cohen, M. Elhadad, and N. Elhadad. Multi-label classification of patient notes: case study on icd code assignment. In *Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[4] H. Harutyunyan, H. Khachatrian, D. C. Kale, G. V. Steeg, and A. Galstyan. Multitask learning and benchmarking with clinical time series data. *arXiv preprint arXiv:1703.07771*, 2017.

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[6] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[7] S. Purushotham, C. Meng, Z. Che, and Y. Liu. Benchmark of deep learning models on large healthcare mimic datasets. *arXiv preprint arXiv:1710.08531*, 2017.

[8] H. Song, D. Rajan, J. J. Thiagarajan, and A. Spanias. Attend and diagnose: Clinical time series analysis using attention models. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[9] J. Varia. Migrating your existing applications to the aws cloud. *A Phase-driven Approach to Cloud Migration*, 2010.