Reproducibility Project Final Report for CS598 DL4H in Spring 2023

Yang Peng
yangp3@illinois.edu

Group ID: 168
Paper ID: 179
Paper link: https://aclanthology.org/S18-2014/
Presentation link: https://www.youtube.com/watch?v=QLxJPqqNfXM
Code link: https://github.com/yang03265/starsem2018-patient-representations

1 Introduction

The paper which I choose to reproduce for my final project this semester is Learning Patient Representations from Text (Dligach et al., 2018). It is written by two authors Dmitriy Dligach and Timothy Miller who are Natural Language Processing (NLP) researchers in the healthcare field. This paper uses electronic health record (EHR) that are publicly available online for healthcare researchers. The data that it uses include patients, billing codes, and clinical notes and the formatting consists of csv or xml formatting.

In this paper, the authors developed a neural network model for learning patient representations. They have proved that the learning patient representations can obtain spectacular results on a standard comorbidity detection task.

The authors developed the neural network model by extending the work from DeepPatient (Miotto et al., 2016). DeepPatient uses Unsupervised deep learning models to predict the future of patients through the use of EHR. Particularly, the DeepPatient features a three-layer stack of denoising autoencoders. This paper further extends the DeepPatient by constructing a neural network system for learning patient representations with only text variables.

2 Scope of reproducibility

This paper includes various data results that I can use for my reproduction studies. The first task is the the billing code classifier where the authors stated that they were able to achieve the macro F1 score on the billing code prediction with 0.447 on randomly initialized CUI embeddings. Then the authors also stated that they achieved macro F1 of 0.473 when using a set of pre-trained CUI embeddings. The takeaway of this is that the authors discovered the pre-trained CUI improved the performance of macro F1 score.

With the above result, the authors decided to a separate SVM classifier for each disease, and they presented the classifier performance for each SVM model. The performance is measured with the baseline SVM classifier trained by the Sparse approach using the standard bag-of-CUIs.

The authors are able to achieve an average F1 score of 0.675, which is equivalent to the best performance of i2b2 system being reported (Solt et al., 2009), despite that they have used a rule-based approach. The dense patient representations approach being performed by the authors however reported a average F1 score of 0.715, which is significantly better than the sprase approach by p=0.03.

In my reproductions studies, I aim to reproduce the authors results listed above, and exploring different ways of preparing the data, including exploration of parallelism in authors code due to my computer science background and industry working experiences.

3 Methodology

In this section, I will go through the authors paper and the models and data that is used by the authors, and the data is obtained. In addition, I will go through the implementation and computational requirements to run this reproduction.

3.1 Model description

The author proposes a neural network model. The input is a set of Unified Medical Language System (UMLS) Concept Unique Identifiers (CUIs) that are derived from the clinical notes of a patient and jointly predicts all billing codes associated with the patient. The CUIs are extracted from the clinical notes with mapping of clinically relevant text such as headaches, shortness of breath, appendectomy, and MRI to entries in the UMLS Metathesaurus.

3.2 Model Architecture

The input set is a set of CUIs which are mapped to 300- dimensional concept embeddings. Then those are averaged and passed onto a 1000-dimensional hidden layer, effectively creating a vectorial representation of a patient.

The model basically includes an embedding layer, averaging layer, hidden layer, and the output layer. The final network/output layer includes n sigmoid units that are used for joint billing code prediction. Then finally, it converts each sigmoid unit to binary (1/0) outcome.

For the multi-label billing code classifier, the authors trained to maximize the macro F1 score for billing code prediction using a learning validation set.

Finally, they created a 1000-dimensional vector representation for each patient in the i2b2 obesity challenge data with sparse (CUI-based) representation for each patient, as the input to the ICD code classifier.

3.3 Data descriptions

The first data source that are used in this paper includes the MIMIC-III Clinical Database (Johnson et, al. 2016). The MIMIC III is a large and free database which is composed of de-identified health-related data associated with over 40 thousand patients who have stayed in critical care units in the Beth Israel Deaconess Medical Center. The duration period is between 2001 and 2012.

The data is freely available through request on MIMIC website managed by MIT. To request access, user needs to obtain a PhysioNet credential which involves a completion of mandatory training course on human subject research and a reference check from your professor or supervisor. Then sign the DUA (Data Use Agreement) and the data can be accessed through various cloud providers such as AWS or GCP. Alternatively, user can also download the tables online which are in csv.gz formatting. Note that the although the approval time from PhysioNet stated that it can take up to 4 weeks, but in my experience it actually only took 2 business days.

The tables that this paper specifically used include the billing codes which reside in CPTEVENTS.csv, DIAGNOSES-ICD.csv, and PROCEDURES-ICD.csv. Then most importantly, the NOTEVENTS.csv which contains the clinical notes from each subjectid visitation details aggregated to a single column.

The second data source includes the use of another publicly available NLP dataset from the Informatics for Integrating Biology to the Bedside (i2b2) Obesity challenge (Uzuner, 2009). The dataset is de-identified and consists of discharge summaries from Partners Healthcare. The obesity information within the dataset have been marked as present, absent, questionable, or unmentioned in the dataset. To obtain the access of the data, a similar procedure compared to the above is to be followed. The approval for i2b2 dataset only took 1 day.

The last data source is the Unified Medical Language System (UMLS) from National Library of Medicine. The UMLS input sets concept unique identifiers (CUIs) are derives from the text notes of a patient and used jointly with all billing codes associated with the patient for learning dense patient vectors. The 2023AA UMLS Metathesaurus Precomputed Subsets will need to be downloaded for this reproduction project. Note that this file can be quite large and intensive computation will take place due to the massive size of CUIs.

3.4 Hyperparameters

In the paper, The authors trained the multi-label billing code classifier for 75 epochs with learning rate of 0.001 and batch size of 50. According to the author, these hyperparameters are obtained by tuning the model's macro F1 on the validation set. As soon as the best values have been determined, they combine the training and validation sets and retained the model.

As it is quite time consume to train the multi-label billing code classifier, I have opted to train with a much less epoch of 15 epochs and the results surprisingly are close to the 75 epochs. I have also tuned the learning rate to 0.0001 and the results were still not better. This shows that the author really did optimized the parameters.

According to the authors, there are two versions of the training model: (1) Randomly initialized CUI embeddings, (2) with word2vec (Mikolov et al., 2013a) pre-trained CUI embeddings. The word2vec pre-trained embeddings are obtained by extracting all CUIs from the clinical notes in MIMIC III NOTEVENTS.csv data using the CBOW method with windows size of 5 and embedding dimension of 300.

Finally, for the i2b2 obesity challenge data, the authors train the multi-class SVM classifiers for each disease and built sixteen SVM classifiers. By following the i2b2 obesity challenge, the models are then evaluated with macro precision, recall, and F1 scores (Uzuner, 2009).

3.5 Implementation

For the implementation, first, after obtaining the relevant tables as described in 3.2 from MIMIC III clinical database, I followed the authors by concatenate the available notes for each patient from the NOTEVENTS.csv table into a single document which is presented in ParseNOTEEVENTS.ipynb, then combine all ICD and CPT codes for a patient to form the targets for the prediction task. Then, I followed the authors way of processing the documents with cTAKES to extract UMLS CUIs. In order to make the data ready for cTAKES format, I then filter out all words that are less than 5 characters to reduce data noise. The code for this part is available at filter.py. The cTAKES are set to directly connected to the UMLS systems. However, I faced extreme slowness using cTAKES to extract UMLS CUIs as there are more than forty thousand patients clinical texts to be mapped. So I have decided to use a different method to accomplish the task of extracting UMLS CUIs. The method I use involves using QuickUMLS, "a unsupervised, approximate dictionary matching algorithm for medical concept extraction" (Soldaini, et al., 2016).

To use QuickUMLS, first I have to download and install the library through its github. Then I need to download MRCONSO.RRF and MRSTY.RRF from UMLS Metathesaurus Precomputed Subsets. In my implementation code, I use QuickUMLS for matching texts with UMLS CUIs. For faster performance, I utilized the concurrent ThreadPoolExecutor from Python so that I can process multiple files at a time. The details are presented in runcui.py in my code.

After the UMLS CUIs are extrated for each patient, I then modified the authors code for training billing code prediction model, then applying and evaluating the model on the i2b2 data. The authors do publish their code on github. The code includes two sections- Codes and Comorbidity. I actually tried to run their code first but identified lots of compatibility issues as it was written with Python2 and using older versions of packages such as keras, tensorflow, pandas, and sklearn. I modified their code extensively, and added my own code for the concatenation part to extract the UMLS CUIs. The code is available thorugh the Codes and Comorbidity section of my github.

3.6 Computational requirements

The authors does state that they have used Titan X GPU that are sponsored by NVIDIA for this research task, without going into details of CPU and Memory required. As I own two Apple M1 Laptop, I used both of my laptop for parsing data and training code prediction model.

Both of my M1 laptop comes with CPU of 8 Cores, and 16 GB of RAM. It also comes with 8 Core GPU. I was not able to run 75 epochs as the author has performed, but rather 10-15 epochs seems to be the maximum before running out of memory. The results however are closely similar and I have adjusted the learning rate to 1/10 of the authors and that seems to achieve better results. Although problem of overfitting could potentially be introduced.

The batch size of hyperparameters seems to impact the performance of the training. Average training time ranges from 3-5 minutes depending on the batch size and learning rate. So an Epoch of 75 can easily run a few hours with lots of memory. Specify a larger batch size for better performance in sacrificing a bit on the accuracy.

4 Results and Analysis

The authors first stated that for the F1 score on the billing code prediction, they were able to achieve with 0.447 on randomly initialized CUI embeddings. My F1 scores were also quite close achieving about 0.403 -0.435 ranges depending on the hypermeters being used.

I used the full dataset of 44,211 patients in MIMIC III dataset as according to the authors. To accomplish the task of matching texts to UMLS CUIs, I have to download a sample of NOTEEVENTS.csv file, write a python program to perform the task so that it outputs the filename as the subjectId and the contents as the text. Then, I put the text into QuickUMLS so that it will output the corresponding UMLS codes in xml formatting then do some data extraction to extract the codes so that now each patient will have a set of codes instead of texts.

Using the training model generated from billing code prediction, I ran the baseline SVM classifier trained by the Sparse approach using the standard bag-of-CUIs. The results I get are very close to what's listed from the authors. The results are the following:

- average p = 0.701
- average r = 0.612
- average f1 = 0.634

This is extremely close to p= 0.733, r= 0.650, f1= 0.675 as ran by the authors using the same Sparse approach.

Let's pick some of the diseases for comparison. The first disease Asthma has p 0.837, r 0.702, and f1 0.745, compared to 0.894, 0.736, 0.787 of the authors result respectively. The numbers are actually quite close of within 10 percent range. Then, picking Obesity which has p 0.727, r 0.72, and f1 0.722, compared to 0.825, 0.791, 0.798 of the authors result respectively. The f1 number this time also stays within 10 percent range. A sample of a full experiment result can be found in this spreadsheet.

Therefore, I conclude that the authors numbers are reproducible and that I am able to match the numbers very closely.

5 Discussion

From the sections above, we know that the numbers which the authors presented in the paper are reproducible through my experiment, matching within the 10 percent range. However, the author did not share multiple pre-training model files as they cannot be shared with the public due to the use of MIMIC III datasets. Therefore, I also do not know if my experiment is correct, and how the author trained the Word2Vec pre-training files.

In this section, I am going to discuss about the what was easy vs hard tasks that I encountered on running reproductions for this paper.

5.1 What was easy

Actually, one part that surprised me is that the i2b2 obesity challenge data is actually not large, so I was able to run the results pretty quick for the second part which I originally anticipated that it was going to be difficult.

Obtaining the datasets required for this reproduction project does require some time but the author has detailed out in the paper on what needs to be obtained. So I face no issues rather than completing the necessary requirements and waiting for a couple days to get every data I needed for.

Parsing data is also surprisingly easy as I was able to utilize many examples I found online for parsing csv files. All the MIMIC III dataset can be downloaded as CSV formatting and this has eased a lot of data parsing for clinical texts.

Furthermore, although the code which are provided by the authors on github cannot be ran due to old versions of library and python being used. However, it was good that I have a baseline for modification and this has saved me a significant amount of time on understand how the authors produced the numbers.

5.2 What was hard/ Recommendations for the authors

The most significant time spent is actually matching the clinical notes into the respective UMLS CUIs. On average, each minute my code could process about 10-20 records depending on the length of clinical notes. If using 15 records as an average per minute, to match the whole MIMIC III dataset would take approximately 2948 minutes, which would take approximately 2 full days. I ran the program on a background and it successfully parsed all the clinical notes and matching it into the respective UMLS CUIs. I also made sure to utilize all my CPU cores on this matching task by specifying the number of ThreadPools used in the code. I am not sure if there is a better way as if I use the cTAKES which is what the authors used it would take more than a week to process these, in addition to running code in Java and spending lots of time on installations. Furthermore, the cTAKES libaray has not being updated for more than 2 years which can be alarming for future reproduction and that I actually ran into errors running it in my macbook. Therefore, I explored other ways such as using QuickUMLS and as it is a python library maintained by GeorgeTown University I ran it without trouble. I believe this is a great feedback which I can share with the authors if necessary so that cTAKES should not be used for future clinical notes matching.

Furthermore, the authors did not state about the pre-training data models that they used. Like what data was used to train them and what was applied. This has led to difficulties to guess what the author has done in the pre-training, and not sure if I am overfitting the data somewhere.

The last challenge part is the resources it takes to consume during a training session. It takes a very long time for training the building code prediction that at times I see it consumes more than 64 GB of memory which freezes my laptop. I had to adjust the hyperparameters which leads to less accuracy as a result unfortunately.

6 Communication with original authors

The original author can be reached at their respective institution's website: Dmitriy Dligach from Loyola University Chicago and Timothy Miller from Harvard Medical School. I have not yet initiated a contact with any of them as I am still executing my plan.

References

Dmitriy Dligach and Timothy Miller. 2018. Learning Patient Representations from Text. In Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics, pages 119–123, New Orleans, Louisiana. Association for Computational Linguistics.

Miotto, R., Li, L., Kidd, B. et al. Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records. Sci Rep 6, 26094 (2016). https://doi.org/10.1038/srep26094

Yuqi Si, Jingcheng Du, Zhao Li, Xiaoqian Jiang, Timothy Miller, Fei Wang, W. Jim Zheng, Kirk Roberts, Deep representation learning of patient data from Electronic Health Records (EHR): A systematic review, Journal of Biomedical Informatics, Volume 115, 2021

Ozlem Uzuner. 2009. Recognizing obesity and co- ¨ morbidities in sparse data. Journal of the American Medical Informatics Association 16(4):561–570.

Illes Solt, Domonkos Tikk, Viktor G ´ al, and Zsolt T ´ Kardkovacs. 2009. Semantic classification of dis- ´ eases in discharge summaries using a context-aware rule-based classifier. Journal of the American Medical Informatics Association 16(4):580–584.

Johnson, A., Pollard, T., Mark, R. (2016). MIMIC-III Clinical Database (version 1.4). PhysioNet. https://doi.org/10.13026/C2XW26.

Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L. A., Mark, R. (2023). MIMIC-IV (version 2.2). PhysioNet. https://doi.org/10.13026/6mm1-ek67.

Luca Soldaini and Nazli Goharian. ”QuickUMLS: a fast, unsupervised approach for medical concept extraction.” MedIR Workshop, SIGIR 2016.