Prediction of Medical Codes from Clinical Text using Pre-trained Language Models

Yeow Long, Chua
University of Illinois at Urbana-Champaign
Https://youtu.be/c72axK-74NE
12th May 2021
ylchua2@illinois.edu

Abstract

Clinical notes are text documents created by clinicians to document/record each patient encounter. It contains medical codes which effectively describes the diagnosis and treatment. Manually annotating these medical codes can be very labor intensive and prone to human error and when medical codes are omitted, it becomes harder for others to understand the specific rationale behind certain specific diagnosis and treatment. As such, NLP (Natural Language Processing) techniques have been used to automate this labelling process. In this work, we introduce the use of pre-trained language representation models such as BERT (Bidirectional Encoder Representations from Transformers) which is simple but powerful and used to obtain state-of-the-art results for various NLP tasks without modifying the BERT architecture. Our method utilises the pre-trained BERT model and fine-tunes it with the MIMIC-III dataset to create a state-of-the-art model to predict medical codes from clinical text. The model is accurate and achieves a precision@8 of 0.42 and a Micro-F1 of 0.411 which performs better than the prior state of the art.

1. Introduction

Clinical notes are text documents created by clinicians during patient encounters and are typically recorded together with a set of metadata codes from the ICD (International Classification of Diseases), a standardised way of recording diagnosis and procedures performed during each patient encounter. These ICD codes have vast number of use cases, from prediction of patient state to billing (Avati et al., 2017, Choi et al., 2016; Ranganath et al., 2015; Denny et al., 2010). As manual coding is very labor intensive and prone to error, there were attempts to study automatic coding with limited success (de Lima et al., 1998). This task is very challenging mainly due to two reasons. First, there are over 15,000 codes in the ICD-9 taxonomy and over 140,000 codes in the newer ICD-10 taxonomies (World Health Organization, 2016). Second, clinical notes also contain alot of information and these includes irrelevant information, spelling errors and a large medical vocabulary. These reasons combined make the prediction of ICD codes from clinical notes a challenging tasks for computers and humans alike(Birman-Deych et al., 2005).

In this paper, we introduce the use of pre-trained language representation models such as BERT which have shown to be effective for improving many NLP tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). There are two steps in the BERT framework: pre-training and fine-tuning. First, during the pre-training process, the model is pre-trained on unlabelled data over various NLP tasks. Second, for fine-tuning, the BERT model is first initialised using the pre-trained parameters and all these parameters are fine-tuned using the preprocessed MIMIC dataset for the specific task of medical codes prediction. Our model design is motivated by the success of transfer learning and the use of pre-trained models to achieve superior performance in various NLP tasks.

We evaluate our approach on MIMIC-III dataset (Johnson et al., 2016), an open dataset of ICU medical records. We utilise the records which consists of clinical notes and codes which describes the patients diagnosis and procedures. Our approach outperforms previous results on medical code prediction using the MIMIC-III dataset.

As we consider the use of this work in a decision support environment, the system needs to be able to explain why it predicted each code and these considerations motivate the use of per-label attention mechanism (James Mullenbach, 2018) which provides justifications and texts for each prediction in the form of text snippets from the clinical notes.

2. Approach

We treat the ICD code prediction problem as a multi-label text classification problem (McCallum, 1999). Let L denote the set of ICD-9 codes; the labelling process for each instance i is to determine the $y_{i,l} \in \{0, 1\}$ for all $l \in L$. We will utilise a pre-trained BERT model and fine-tune the BERT model using the MIMIC-III dataset. As the maximum input length of BERT is only 512, we split the text into 5 different parts of equal length of 500 and for each of this part, have a separate BERT model trained on this specific portion of the text as illustrated in Fig. 1.

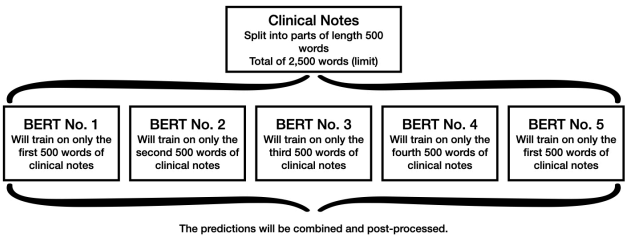

Fig. 1. Architecture of the entire model that will be used to make code predictions from clinical text

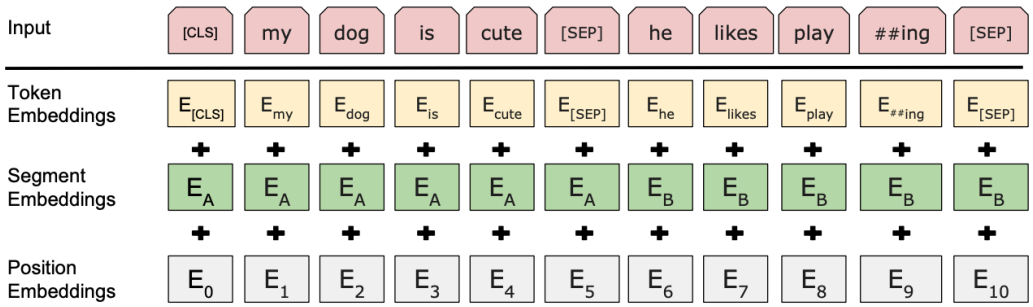

Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

2.1.Pre-trained BERT architecture

We introduce BERT in this section. There are two separate steps in the BERT framework: pre-training and fine-tuning. During the pre-training phase, the BERT model is first initialised with the pre-trained parameters and all of its parameters are fine-tuned using the MIMIC-III dataset. A distinctive feature that is unique to BERT is that there are only minor differences between the pre-trained and fine-tuned architecture.

The BERT model architecture is a multi-layer bidirectional Transformer encoder based on the implementation of (Vaswani et al., 2017) and since our implementation is similar to the original, we will omit an comprehensive description of the model architecture and refer readers to (Vaswani et al., 2017) and "The Annotated Transformer".

In this work, we denote the number of transformer blocks or layers as L and the hidden size of our classifier as H. We report results with L set to 30 and H set to 768.

To make BERT compatible, we'll need to tokenise the clinical notes that is fed into BERT. We'll need to ensure that the first token of every input sequence to our model is a special classification token ([CLS]), separate different sentences either with a special separation token ([SEP]) or construct a segment embedding to indicate which sentence it belongs to and the classification token ([CLS]) again to indicate the end of the input sequence as well as the start. For each input word, we construct its input representation by adding up the token, segment and position embeddings as illustrated in Fig. 2.

Fine-tuning is really easy since the attention mechanism in the BERT architecture allows it to model many downstream tasks by appropriately swapping out the inputs and outputs of the model. For our use-case, our inputs will be clinical notes and the outputs will be the medical codes. Compared to pre-training, fine-tuning takes up less time and all the results in this paper can be replicated within a few hours using the same initial pre-trained BERT model.

2.2.Fine-tuning BERT: Convolutional Attention Architecture

On top of the BERT model, we have pre-trained embeddings for each entry of the clinical notes which are horizontally concatenated into a matrix of the form $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N]$, where N is the length of the document. We make use of a convolutional filter to combine the word embeddings and at each step, we compute

$$H_n = g(W_c * x_{n:n+k-1} + b_c)$$

where * denotes the convolution operator, g denotes the element-wise non-linear transformation and b_c denotes the bias. We pad the input in order to achieve a resulting matrix of size d_c by N.

After convolution, we reduce this matrix vector by applying pooling across the length of the document and select the maximum or average value at each row (Kim, 2014). We assign multiple labels for each entry of the clinical notes as it is very possible that a single clinical note entry can lead to multiple code predictions. As a result, we use a per-label attention mechanism, selecting the k-grams that is most relevant for each predicted label.

For each label prediction l, we compute the product $H^T u_l$ and pass the result through a softmax operator obtaining the distribution over text location in the document

$$\alpha_l = SoftMax(H^T u_l),$$

The attention vector is then used to compute the vector representation for each label

$$v_l = \sum_{n=1}^{N} a_{l,n}, h_n$$

We use max-pooling to compute a single vector v for all labels to serve as a baseline.

$$v_j = \max_n h_{n,j}$$

Given the vector representation, we compute the probability for all labels using a linear layer and sigmoid transformation.

$$\hat{y}_l = \sigma(\beta_l^T v_l + b_l)$$

where beta contains the prediction weights and b is the bias.

For the training procedure, we minimise the binary cross-entropy loss,

$$L_{BCE}(X, y) = -\sum_{l=1}^{L} y_l log(\hat{y}_l) + (1 - y_l) log(1 - \hat{y}_l)$$

plus the L2 norm of the model weights (Kingma and Ba, 2015).

3. Experimental Evaluation

3.1. Text Pre-processing

In this work, we deal with a multi-label classification problem and before we build and train our model, we need to first pre-process the data. We make use of the MIMIC-III dataset and outlined the following pre-processing steps similar to (James Mullenbach, 2018).

We'll need to first extract the ICD-9 codes from the procedures and diagnoses files and format the medical codes accordingly before combining and filtering them to obtain the top 50 ICD-9 codes. Lastly, we combined the ICD-9 codes with the clinical notes on admission ID. The entire ETL process is as illustrated at Fig. 3.

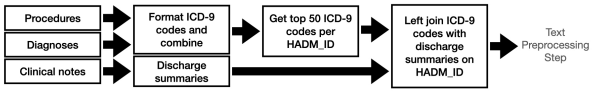

Fig.3 Extract-Transform-Load

After the ETL process, we proceed to the text pre-processing step where we first remove special characters and numeric-only words, remove the stop-words and tokenise the text and limit the tokens to a limit of 2,500. Finally, we train our word2vec embeddings which will create features from our preprocessed text. This will convert the words for every clinical note summaries into 2-dimensional array of vectors using word2vec (Tomas Mikolov, 2013). For every word we have a 1-dimensional representation of fixed depth with a specified vector size of 100. We will create a dictionary for each of the word in the clinical summaries. We zero-pad the vector so as to ensure that the clinical summaries are of a fixed length. The result is a matrix with each row the different clinical summaries embeddings of fixed depth 100. The entire text-preprocessing step to generate the word2vec embeddings is as illustrated at Fig. 4.

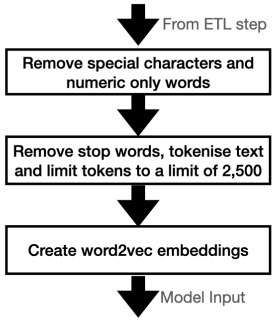

Fig.4 Text-processing

3.2. Evaluation of medical code prediction

We evaluate the performance of our model against several baselines. The dataset used to perform medical code prediction is the MIMIC-III dataset (Johnson, 2016).

MIMIC-III dataset is an open-access dataset of text and unstructured medical records from an undisclosed hospital intensive care unit (ICU). In this work, we focus on the discharge summaries which in a nutshell, contains information about a patient's period in the ICU. In MIMIC-III, for some of the admissions, there are addenda added to their discharge summary.

Each admission is manually tagged by human coders with ICD-9 codes describing both diagnoses and procedures during the patient's period in the ICU. There are 8,921 unique ICD-9 medical codes present in the dataset as well as 6,918 diagnosis codes and 2,003 procedures codes. Since some patients have multiple admissions and discharge summaries, we perform out train-test split by patient ID to ensure that no patient is in both the training and test dataset. In this work, we filter and extract only the top 50 most frequent codes resulting in 8.067 summaries for training, 1,574 for validation and testing.

We compare the performance of our BERT model against several baselines:

- 1-dimensional 1-layer CNN by (Kim, 2014)
- Bag-of-words logistic regression model CBOW-LR
- Bidirectional gated recurrent unit Bi-GRU
- Convolutional attention CAML by (James, 2018)

For the CNN, Bi-GRU and CAML models, the models are initialised using the pre-trained word2vec model and all these models are implemented using PyTorch.

3.3. Evaluation Metrics

In order to facilitate the comparison between different model, we will report various different metrics, with a focus on macro-averaged and micro-averaged F1 as well as area under the curve. Macro-averaged F1 are computed by averaging the F1-score computed per-label whereas micro-averaged F1 are computed by treating each text and code pair as a separate prediction. Other than F1-score, Micro-R scores will be reported as well.

$$Micro - R = \frac{\sum_{l=1}^{|L|} TP_l}{\sum_{l=1}^{|L|} TP_l + FN_l}$$

$$Macro - R = \frac{1}{|L|} \sum_{l=1}^{|L|} \frac{TP_l}{TP_l + FN_l}$$

where TP denotes the true positive and FN denotes false negatives. Precision will also be computed. We also report precision at n denoted as 'P@n' which is the combined precision of the n highest scored labels and this is motivated by the use case as a decision support application where a user is presented with a fixed number of predicted codes to review. We will present precision with values 5 and 8 to compare with prior work (Vani et al., 2017; Prakash et al., 2017). We will also compute precision@15 which roughly corresponds to the average number of medical codes for each of the MIMIC-III discharge summaries.

3.4. Results

Our main evaluation of our model involves predicting only the top 50 medical codes based on the discharge summaries of the MIMIC-III dataset. The BERT model achieves the best results on most of the metrics. We see that attention yields improvements on traditional or vanilla CNN models. The performance of the Bi-GRU is comparable to the CNN model whereas the logistic regression model perform worst than the other neural network architectures.

We calculate the precision@8 metrics which is a good measure to gauge a model's performance to return a small high-confidence predictions and this is closely tied to the model's ability to be used in a decision support environment. Our model achieves good precision and recall. It is also apparent how difficult it is to achieve a high Macro-F1 score due to the metric's emphasis on rare-label performance and our model has achieved a Macro-F1 of 0.285 and Micro-F1 of 0.411 which outperforms most of the other models. To put things into perspective, a hypothetical system that perform perfectly on 500 most frequent labels will achieve a Macro-F1 of 0.052 and a Micro-F1 of 0.842.

	AUC		F1		P@n	
Model	Macro	Micro	Macro	Micro	8	15
Logistic Regression	0.812	0.975	0.009	0.238	0.466	0.358
CNN	0.842	0.977	0.020	0.301	0.488	0.370
Bi-GRU	0.876	0.983	0.043	0.403	0.585	0.440
CAML	0.733	0.970	0.037	0.452*	0.592	0.455
BERT	0.801	0.846	0.285*	0.411	0.421	0.292

Table 1: Results on MIMIC-III, top 50 labels. Note that the results of CAML model is lower than James, 2018 as CAML was not hyper parameter tuned. Here, when denoted by (*) means that this is the best result and this result is significantly improved compared to the next best result.

4. Conclusion

In this work, we built a deep learning model and made use of pre-trained language model to perform ICD-9 code predictions from clinical notes. We evaluated the performance of the proposed BERT model against LR, GRU, CNN and CAML using only the top 50 medical codes to reduce the dimensionality and class imbalance issues of the MIMIC-III dataset and also to reduce training times.

Our contribution is that it is possible to utilise pre-trained language models without domain knowledge to perform medical codes prediction from clinical text. Definitely, this work would not be possible without the recent advances and improvements in transfer learning with pre-trained language models. Although we focus on a clinical setting, it is definitely possible to perform other multi-label document tagging such as ICD-10 code predictions. There are several directions for future work. It is possible to explore the different or hybrid combination of LR/CNN/BiGRU/CAML with BERT. The other aspect is ensembling of the different models to obtain a model that performs better than the prior state-of-the-art. From the linguistics side of things, it is definitely possible to cater for non-standard writing and other sources of vocabulary tokens. Ultimately, we want to be able to attempt to predict medical codes for future events from existing discharge summaries which is a much more difficult tasks compared to present (James 2018).

5. Challenges

This is definitely one of my first time embarking on a research project on my own and part of the reason why I decided to pursue this solo is so that I can learn as much as possible. Having a keen interest in natural language processing, I selected this project on NLP as I was already quite involved in the basics of NLP. It was the small things that I overlook which wasted a lot of my time and one of them was the conversion of the diagnoses and procedure codes into the full formats with dots. Luckily, I was able to find this mistake after reading (James 2018) paper multiple times as well as looking through his code repository on the paper. I was behind time especially during the week where we had to submit the project drafts. I was literally working til the last minute to try to see if I can get the baseline models working before the draft submission so that I can show some results. However, due to some mistakes in the text pre-processing stage as well as the ETL portion which I was able to correct them all later, I was delayed further. I probably should have tried to reduce the dimensionality of the problem/target labels to just the top 50 from the start. During the initial stages of model training, the model's performance on the rare-labels were pretty bad which made me think about reducing the dimensionality of the problem and reduce the number of labels/codes to just the top 50 and the idea came from James 2018 paper. I definitely should have read and understood the related papers instead of jumping straight to implementation and re-visiting reading the papers later. I would definitely do things differently the next time round and be careful and try to fully understand certain algorithms before implementing them. Being more careful and reading papers before implementing is probably the way to go.

The most challenging part about this work is to adapt the BERT model to perform medical codes from clinical text. The first issue is the inability of the BERT model to handle inputs of more than 512 in length. As the problem had an initial word limitation of 2,500 words or tokens, I decided to split the text equally into 5 different 500 words portions and have a BERT model specifically trained on this section of the clinical texts and having tried combining the results, I eventually resulted in manually combined the 5 different BERT models so as to filter cases where the texts does not involve all 5 of the models which means the text length is less than 2,000 (only involved the first 4 models), 1,500 (involve the first 3 models) 1,000 (involve the first 2 models), 500 (involve the first model only).

After the texts is preprocessed for BERT, there were many things I wanted to do and I realised I have limited time. First, fine-tune the entire BERT architecture using the MIMIC-III dataset and observe the results. Second, freeze the BERT architecture and add neural network layers and train the MIMIC-II dataset on these newly added layers and I realised that this second procedures is actually not that straightforward as anticipated. Third, given more time I would probably explore the possibility of adding LR/CNN/BiGRU/CAML layers on BERT. I find this idea pretty interesting and in this work, this was omitted due to a lack of time.

In order to establish a baseline, I rush to implement the different LR, CNN, GRU and also CAML models and definitely, these models can be further tuned and achieve higher scores but I decided to focus more on the BERT portion. The other aspect that I sort of underestimated is the time it takes to fine-tune the BERT model on the MIMIC-III dataset. On a typical iteration, I observe it takes around 2 to 3 hours to finish training and a lot of time is spent wasted on careless mistakes. I probably should have been more careful. Given more time, I would be able to further improve on this work. I should have been more organised and careful so that more can be work in this short 2 weeks of time allocated for this project. Probably also I should have started early. There were a lot of things I would have done differently.

6. References

Anand Avati, Kenneth Jung, Stephanie Harman, Lance Downing, Andrew Ng, and Nigam H. Shah. 2017. Improving palliative care with deep learning. *arXiv preprint arXiv:1711.06402*.

Edward Choi, Mohammad Taha Bahadori, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Doctor AI: Predicting clinical events via recurrent neural networks. In *Machine Learning for Healthcare Conference*. pages 301–318.

Rajesh Ranganath, Adler J. Perotte, Noémie Elhadad, and David M. Blei. 2015. The survival filter: Joint survival analysis with a latent time series. In *UAI*. pages 742–751.

Joshua C. Denny, Marylyn D. Ritchie, Melissa A. Bas- ford, Jill M. Pulley, Lisa Bastarache, Kristin Brown- Gentry, Deede Wang, Dan R. Masys, Dan M. Ro- den, and Dana C. Crawford. 2010. PheWAS: demon- strating the feasibility of a phenome-wide scan to discover gene–disease associations. *Bioinformatics* 26(9):1205–1210.

Luciano R.S. de Lima, Alberto H.F. Laender, and Berthier A. Ribeiro-Neto. 1998. A hierarchical ap- proach to the automatic categorization of medical documents. In *Proceedings of the seventh international conference on Information and knowledge management*. ACM, pages 132–139.

World Health Organization. 2016. International statistical classification of diseases and related health problems 10th revision. http://apps.who.int/classifications/icd10/browse/2016/en

Elena Birman-Deych, Amy D. Waterman, Yan Yan, David S. Nilasena, Martha J. Radford, and Brian F Gage. 2005. Accuracy of ICD-9-CM codes for identifying cardiovascular and stroke risk factors. *Medical care* 43(5):480–485.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *Advances in neural information processing systems*, pages 3079–3087.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical re- port, OpenAI

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*. Association for Computational Linguistics.

Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, Li- wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data* 3.

J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein. Explainable prediction of medical codes from clinical text. ACL Anthology, 2018.

Andrew McCallum. 1999. Multi-label text classification with a mixture model trained by EM. In *AAAI workshop on Text Learning*. pages 1–7.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pages 1746–1751.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Ankit Vani, Yacine Jernite, and David Sontag. 2017. Grounded recurrent neural networks. *arXiv preprint arXiv:1705.08557* .

Aaditya Prakash, Siyuan Zhao, Sadid A Hasan, Vivek V Datla, Kathy Lee, Ashequl Qadir, Joey Liu, and Oladimeji Farri. 2017. Condensed memory networks for clinical diagnostic inferencing. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. pages 3274–3280.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.