

Project Report Draft, CS598 DL4H in Spring 2023

Manu Vinod Shesha

manuv3@illinois.edu

Group ID: 96

Paper ID: 187

Presentation link: TBD

Code link: TBD

1 Introduction

This work reproduces the idea and results in the original paper: *Automated ICD-9 Coding via A Deep Learning Approach*, published by Min Li, Zhihui Fei, Min Zeng, Fang-Xiang Wu, Yaohang Li, Yi Pan, and Jianxin Wang (Li et al., 2019).

The paper aims to automate the extraction of ICD-9 (Ninth Revision of International Classification of Diseases) codes from patient discharge summary, through application of state-of-art (general) text processing technique called Document-to-Vector (D2V) in combination of more general Convolution Neural Network.

The extraction of ICD-9 codes from free-text Discharge summary is needed to do billing, as well as raising insurance claims with the provider. Today, this procedure to extract medical codes from patient discharge summary is largely a manual effort, undertaken by hospital's medical record department personnel. This has two problems:

- The process is very slow and inefficient, causing delay in the patient discharge process.
- The process requires specialized knowledge making it costly, and sometimes error prone.

The authors describe a novel approach to this problem through usage of DL techniques. They treat it as a task of multi-label classification (of ICD-9 codes). Specifically, the authors combine two approaches to produce vector embedding of documents, which are further used in multi-label classification:

- CNN (Convolution Neural Network) to discover, and extract **local features** in text. We can Intuitively, the local context of the text, like phrases describing a medical concept, should be important in deriving related ICD codes.

- D2V (Document to Vector) (Le and Mikolov, 2014) embedding technique to capture the **global features** of the document. D2V extends upon Word2Vec, and (unlike CNN) takes order of words into account. It should be noted that D2V unsupervised learning approach.

2 Scope of reproducibility

The paper proposes a novel DL (D2V + CNN) based approach to infer ICD-9 codes from unstructured patient discharge summary. It claims that proposed DL model combining Doc2Vec (D2V) and CNN, generates better embedding for documents, which perform better in multi-label classification task of identifying ICD-9 codes, when compared to traditional ML-based models.

More specifically it claims:

- Higher Micro-average F1-score in multi-label ICD-9 classification task for the proposed model when compared to baseline models: Flat-SVM and Hierarchical-SVM.

Model	Precision	Recall	F1-score
flat-SVM	0.635	0.158	0.253
hierarchical-SVM	0.415	0.280	0.335
D2V+CNN	0.486	0.351	.408

All values are Micro-averaged.

- Each of the components: CNN and D2V provide important contribution to the overall accuracy of the model (, with CNN being more important). The model accuracy degrades significantly when either of these components are removed.

Model	Precision	Recall	F1-score
Only CNN	0.440	0.366	0.399
Only D2V	0.375	0.261	0.308
D2V+CNN	0.486	0.351	.408

All values are Micro-averaged.

2.1 Addressed claims from the original paper

- Higher Micro-average F1-score in multi-label ICD-9 classification task for the proposed model when compared to baseline models: Flat-SVM and Hierarchical-SVM.
- The model accuracy degrades significantly when either of the components: D2V or CNN are removed.

3 Methodology

3.1 Model descriptions

The proposed model is a DL network with two "logical" components:

- Encoder to generate document embeddings: The function of this component is to generate effective fixed-length embedding for a given discharge summary document. This component consists of two "logical" sub-components:
 - D2V: This sub-component first trains (as pre-processing step) Doc2Vec model to learn input document vectors of length **128**, in an unsupervised way. It then fine tunes this vector, using a fully connected layer of **64** neurons, followed by a non-linear activation like sigmoid. This fine-tune layer is trained in supervised way.
 - CNN: This sub-component trains a Word2Vec model as pre-processing step to build word vectors for the whole vocabulary of the collective corpus of documents. For each document, all the vectors corresponding to the contained words, are concatenated, to represent the given document. These document vectors are used as input to the CNN sub-component. This sub-component actually comprises of 3 single-layer multi-channel CNN models. Three CNN models correspond to 3 kernel sizes (of 3, 4, and 5 words)) with 64 output channels each. For CNN layer in each model is followed by a MaxPool layer to perform temporal pooling. The outputs of each of these CNN models are concatenated to generate the output vector per document of size **192 (3 models * 64 channels each)**.

The output vectors from the two sub-components (D2V and CNN) are concatenated to produce the final vector for each document in the batch. This final vector size is **256 (64 from DNN + 192 from CNN)**.

- Classifier to perform multi-label classification of ICD-9 codes. This component consists of:
 - Dropout layer: The document vector generated by encoder component is regularized by stochastically dropping different dimensions.
 - Fully connected layer with sigmoid activation: This layer generates the final output of size **6984 (total number of ICD-9 codes)**. Each dimension (representing an ICD-9 code) is assigned a probability by sigmoid activation.

Architecture is summarized in following diagram:

The objective is to do end-to-end training of this model. Doc2Vec and Word2Vec models are trained unsupervised, in the pre-processing step. During the supervised learning phase, the above described components are trained for multi-label classification task using back-propagation. The goal is to achieve similar or close enough performance in terms of micro-averaged F1-score, as achieved in original paper.

The model is medium complexity model in terms of the number of parameters to train.

CNN layer (300, 400, 500 sized 1D-kernels with 64 channels each)	76992
D2V Fine-tune layer (128 input dimension, 64 neurons)	8256
Fully-connected classification layer (256 input dimension, 6984 neurons)	1794888

3.2 Data descriptions

The model is trained and tested on MIMIC-III **MIMIC-III** (Medical Information Mart for Intensive Care) database. As per Physionet website (which maintains MIMIC database): *MIMIC-III is a large, freely-available database comprising de-identified health-related data associated with over forty thousand patients who stayed in critical care units of the Beth Israel Deaconess Medical Center between 2001 and 2012.* The data is publically available, upon credentialing user identity on Physionet, and completing the mandatory training: Data or Specimens Only Research.



- w2v-d2v.ipynb: This notebook generates and saves following:

- Gensim Word2Vec model based vectors for all the tokens in the vocabulary of our corpus.
- Gensim Doc2Vec model based vectors for all the documents in our corpus.

- dl-model.ipynb: This notebook trains and validates the DL model (as described in the model section).

Link to source code: [link](#)

3.5 Computational requirements

The computational requirements for different steps are described below:

Specifically, following datasets in MIMIC-III were used:

- DIAGNOSES_ICD.csv: Each row in this file maps HADM_ID (Hospitalization ID) of a patient with a unique ICD9_CODE. We transform ICD9_CODE to one-hot encoding and group them per per HADM_ID to generate multi-hot encoding.

Total # codes	6984
Avg. # codes per patient	11
Max # of codes per patient	39
Min # of codes per patient	1

- NOTEEVENTS.csv: Each row maps HADM_ID (Hospitalization ID) with a TEXT (free text Discharge summary) field.

Total # discharge summary	52691
Avg. # words per discharge summary	1524
Max # of words per discharge summary	7980
Min # of words per discharge summary	9

3.3 Hyperparameters

TODO

3.4 Implementation

The implementation has been entirely done by the team. To the best of knowledge of the team, no implementation code by authors of original paper, or others is available publicly.

The implementation consists of three Jupyter notebooks:

- data-preprocessing.ipynb: This notebook imports the datasets (DIAGNOSES_ICD.csv and NOTEEVENTS.csv), applies necessary pre-processing and saves the final datasets.

- MIMIC-III datasets import and preprocessing: All the steps were performed on personal machine with 16 GB of total RAM and Intel® Core™ i7-10750H × 12 processor. Only one operation was a bit compute intensive: a "join" operation, during creation of multi-hot ICD9 embeddings for each hospitalization. This operation took around 14 seconds on the CPU of personal machine.

- Word2Vec and Doc2Vec model training and vectors generation: All the steps were performed on personal machine. Gensim implementation of Word2Vec and Doc2Vec support streamed source. So, there is no need to load the whole corpus in memory. However, this is no GPU support in Gensim yet. Since there is no parameter tuning in scope, and this is one time operation, it was performed on personal machine. The time taken to generate these encodings are as following:

Word2Vec embeddings	37 minutes
Doc2Vec embeddings	51 minutes

- Training the full Deep Learning model: The computational requirements were high due to following factors:

- Each input data to DL model is 70000 dimensional float vector (700 words max with 100 dimensional W2V encoding each). With more than 52600 documents, this requires both large amount of GPU

memory (14 GB) to store, as well as parallel computing on GPU. Such computing is not possible on available personal computing environment (correctly predicted in the proposal).

- High dimension of problem. There are 6984 ICD9 codes found in the dataset, forming as many dimensions in the multi-label classification problem. As a result the final fully connected layer has over 1.79 million parameters.

Here, we improve the training time per epoch at the cost of consuming more GPU memory by loading the whole float tensor (52691 x 70000) of size 14 GB in GPU memory. Such a strategy requires premium GPU which is available through Google Collab Pro. On Colab Pro, we could train and validate the model per run, with following resource utilization:

System RAM used	8GB
GPU RAM used	23GB (Includes data and model)
Computation units burn rate	13.08 per hour
Time taken to train per epoch	3.5-4 minutes
Total time taken to run 80 epochs	5 hours

Multiple such runs (4 to 5) were executed for hyperparameter tuning.

- Training partial model with only D2V (for ablation): During the training of this model we just need to tune the parameters of 2 fully connected layers, which is not very computationally intensive. On Google Collab Free environment, we could achieve this easily:

System RAM used	4.6GB
GPU RAM used	1.2GB
Total time taken to run 80 epochs	24 minutes

As, estimated during proposal, most of the steps in the whole reproduction process could be performed in personal machine with 16 GB RAM and Intel I7 12-Core CPU. The task of training the main DL model is compute intensive, and required premium GPU support on Google Collab environment.

4 Results

The results achieved so far validate the main claim made by the paper, which is that the combined D2V + CNN model performs the multi-label ICD9

classification task with better micro-averaged f1-score, in comparison to traditional ML models: flat-SVM and hierarchical-SVM.

Start with a high-level overview of your results. Does your work support the claims you listed in section 2.1? Keep this section as factual and precise as possible, reserve your judgement and discussion points for the next “Discussion” section.

Go into each individual result you have, say how it relates to one of the claims, and explain what your result is. Logically group related results into sections. Clearly state if you have gone beyond the original paper to run additional experiments and how they relate to the original claims.

Tips 1: Be specific and use precise language, e.g. “we reproduced the accuracy to within 1% of reported value, that upholds the paper’s conclusion that it performs much better than baselines.” Getting exactly the same number is in most cases infeasible, so you’ll need to use your judgement call to decide if your results support the original claim of the paper.

Tips 2: You may want to use tables and figures to demonstrate your results.

4.1 Result 1

We reproduced the DL model, based on D2V and CNN components, to generate document vectors which capture local and global characteristics of the text, and perform multi-label classification task of identifying the associated ICD9 codes. The result so far are summarized in table below:

Precision	Recall	F1-score
0.467	0.344	0.396

All values are Micro-averaged.

The result is based on following hyperparameters:

Dropout-rate	Probability-Threshold	Epochs
0.20	0.20	80

Probability-Threshold is the threshold for binary classification.

- The F1-score of 0.396 is better than flat-SVM (0.253) by 36% and hierarchical-SVM (0.335) by 15
- The F1-score of the model trained so far is 2.9% worse than the one obtained by authors, in their work.

4.2 Result 2

We trained a partial DL model (without the CNN component), as part of ablation, to validate the importance of CNN component. The results are provided below:

Precision	Recall	F1-score
0.291	0.324	0.307

All values are Micro-averaged.

The f1-score of 0.307 of this partial model is very close to that obtained by original authors 0.308. This result clearly indicates the severe degradation from the performance of combined model, emphasizing the importance of CNN model.

4.3 Additional results not present in the original paper

TODO

5 Discussion

The original paper is reproducible to a large extent. Based on the results achieved so far, performance claim on the main model has been reached within 3% of original values. Moreover, it has been conclusively proved that the proposed model improves on the performance of baseline models by 15-36

The results also validates the overall approach taken:

- Maintain the quality and quantity of data, to ensure that some important features are not missed due to any data loss.
- Do not alter the main parameters of the model (as proposed by authors) like word/document embedding lengths, kernel sizes, count of activation units, number of layers, etc.
- Vary the hyperparameters like dropout rate, classification threshold etc. to achieve the desired performance metrics.

One weakness of the approach was to not plan for faster and cheaper way to perform hyperparameter training. So long training duration (of about 5 hrs for 80 epochs) was costly, as well as limited the ability of experiment with hyperparameters.

As next steps:

5.1 What was easy

- It is relatively easy to obtain the raw MIMIC-III data, and be able to extract almost same amount of samples as used by original authors, with basic pre-processing.
- It was easy to train Gensim Word2Vec and Doc2vec models. Gensim library is pretty stable, and intuitive to use, with decent documentation.

5.2 What was difficult

- The authors left some gaps in the architecture of the model. For example, the architecture of D2V fine-tuning layer was not clarified. The overall architecture diagram provided by authors is abstract and leaves out finer details like activation functions in DL layers. So, it was an effort to bridge some missing pieces.
- The size of the model is relatively big, and lots of time was spent during training cycles. Also, cloud computing costs added up, due to intensive use of GPU.

5.3 Recommendations for reproducibility

- The authors should provide more detailed architecture of the DL model. For example, where exactly are the dropout layers located in the model, what are the actual activation functions used, etc.
- The authors should give more detailed description of the implementation. For example, what was their training environment, how many epochs did they run, etc.

6 Communication with original authors

No communication happened with original authors, so far.

References

- Quoc V. Le and Tomas Mikolov. 2014. [Distributed representations of sentences and documents](#).
- Min Li, Zhihui Fei, Min Zeng, Fang-Xiang Wu, Yao-hang Li, Yi Pan, and Jianxin Wang. 2019. [Automated icd-9 coding via a deep learning approach](#). *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(4):1193–1202.