Deep Learning Models for Identification of Key Medical Classifier Terms from EHR related to Cardiology

March, April 2021

OVERVIEW

As an internship activity, the team of Noel Alben, Niraj Anil, Sai Deepika, and J Sumanth were in charge of training and testing various deep learning models for the identification and extraction of key medical terms from electronic health records (EHR's) related to Cardiology to be later used as a feature in the mobile application.

GOALS

1. Find and implement methods to extract medical terms and classifiers from SNOMED CT for purposes of identifying Key Medical Classifier terms from free text.
2. Search for additional databases for terms related to cardiology, specifically discharge summaries and Electronic Health Records.
3. Find, Test, and Train various Deep learning Models for key Medical term identification and extraction from free text Electronic Health Records [Named Entity Recognition].
4. Create a plan for integration and implementation of this problem statement.

SPECIFICATIONS

The models are designed and altered using python script and most of the training of the models took place on local systems which brought about testing efficiency between 90-95%. We discovered multiple models each with its purpose and different parameters and in this document, we will go over each of them in detail along with their merits and demerits.

Keywords: Natural Language Processing, SNOMED CT, Named Entity Recognition, Clinical Term Identification, BERT

Table Of Contents

MILESTONES

1. Initial Literature Survey for SNOMED CT Integration

1.1. SNOMED-CT:

SNOMED CT is the most comprehensive, multilingual clinical healthcare terminology in the world [1]. It provides a standardized way to represent clinical phrases captured by the clinician. Our preliminary goal was to use SNOMED-CT as a database and dictionary to identify, extract and add descriptions for the key medical terms from the free text Electronic Health Records [2].

Now, we will explain the various aspects of Snomed-CT and how it is a valuable asset for our problem statement,

Snomed-CT browser can be accessed with this link.

The SNOMED CT logical model defines how each type of SNOMED CT component and derivative is related and represented [3]. The core component types in SNOMED CT are ***Concepts***, ***Descriptions,*** and ***Relationships***.

Concepts: Every concept represents a unique clinical meaning, which is referenced using a unique, numeric, and machine-readable SNOMED CT identifier. The identifier provides an unambiguous unique reference to each concept and does not have any ascribed human interpretable meaning [3].

Descriptions: Every **concept** is represented with two types of descriptors:

- *Fully Specified Name [FSN]:* The FSN represents a unique, unambiguous description of a concept's meaning. This is particularly useful when different concepts are referred to by the same commonly used word or phrase. Each concept can have only one FSN in each language or dialect.
- *Synonym:* A synonym represents a term that can be used to display or select a concept. A concept may have several synonyms. This allows users of SNOMED CT to use the terms they prefer to refer to a specific clinical meaning.

Relationships: A relationship represents an association between two concepts. Relationships are used to logically define the meaning of a concept in a way that can be processed by a computer. There are different types of relationships available within SNOMED CT.

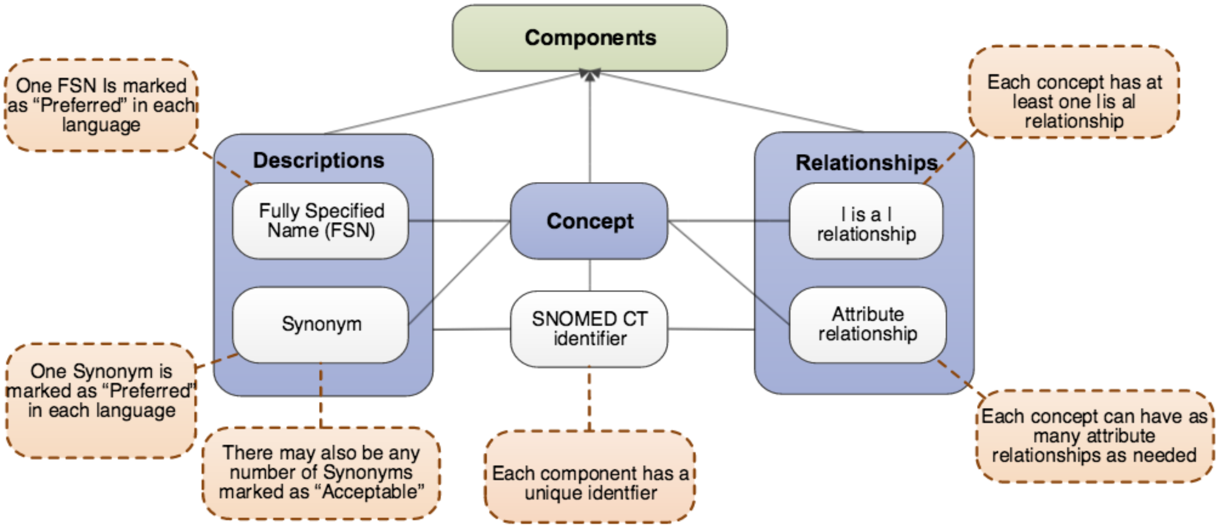

Components
One FSN Is marked as "Preferred" in each language
Descriptions
Fully Specified Name (FSN)
Synonym
Concept
SNOMED CT identifier
Relationships
| is a | relationship
Attribute relationship
Each concept has at least one |is a| relationship
One Synonym is marked as "Preferred" in each language
There may also be any number of Synonyms marked as "Acceptable"
Each component has a unique identfier
Each concept can have as many attribute relationships as needed

Figure 1. Snomed CT concept, description, and relationship.

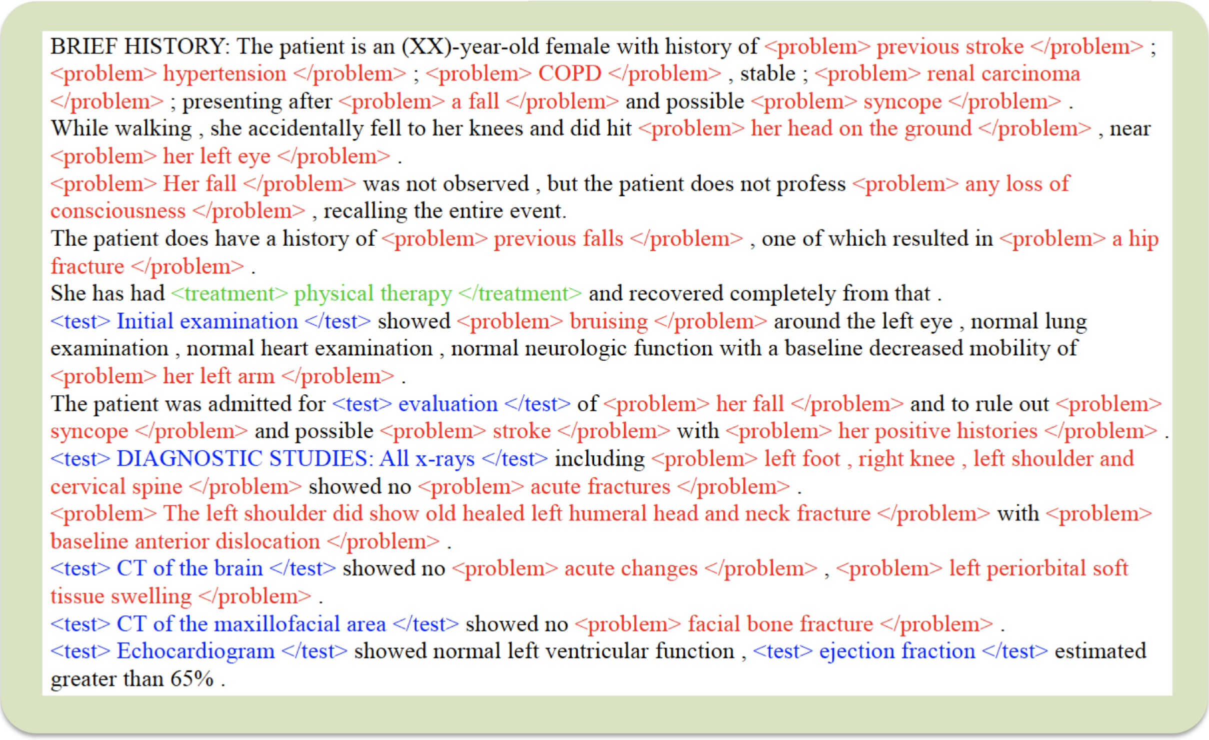
1.2. Natural Language Processing:

The first step taken by us was to understand the multiple facets of Natural Language Processing for free text and extract what is necessary and essential for our problem statement. **NAMED ENTITY RECOGNITION (NER)** is the algorithm that will be required to provide a solution to the identification of key clinical entities from free text Electronic Health Records [4].

Within **NAMED ENTITY RECOGNITION (NER)** an important step is the **Word2Vec**. **Word2vec** is a semantic learning framework that uses a shallow neural network to learn the representations of words in a text [5]. The main objective of it is a technique that can be used to learn high-quality word vectors from huge data sets with billions of words, and with millions of words in the vocabulary.

Once the relevant terms are extracted the next task required is to classify the extracted terms into predefined classifiers, the most common classifiers for Natural Language Processing related to Parts Of Speech classifiers and in the medical domain we can classify them as Disease, treatment, problem, and tests (refer fig 2.)

In the medical domain, **NER** systems [6] are called Medical Entity Recognition (MER). These systems try to detect and delimit Medical entities in texts and classify them into a given category [7]. Most of the MER systems utilize machine learning solutions that use large amounts of annotated datasets as input to the system. Clinical language is at the heart of our problem statement and along with that arrive an array of problems that we as a team have had to navigate through in the process of our internship.



BRIEF HISTORY: The patient is an (XX)-year-old female with history of <problem> previous stroke </problem> ; <problem> hypertension </problem> ; <problem> COPD </problem> , stable ; <problem> renal carcinoma </problem> ; presenting after <problem> a fall </problem> and possible <problem> syncope </problem> .
While walking , she accidentally fell to her knees and did hit <problem> her head on the ground </problem> , near <problem> her left eye </problem> .
<problem> Her fall </problem> was not observed , but the patient does not profess <problem> any loss of consciousness </problem> , recalling the entire event.
The patient does have a history of <problem> previous falls </problem> , one of which resulted in <problem> a hip fracture </problem> .
She has had <treatment> physical therapy </treatment> and recovered completely from that .
<test> Initial examination </test> showed <problem> bruising </problem> around the left eye , normal lung examination , normal heart examination , normal neurologic function with a baseline decreased mobility of <problem> her left arm </problem> .
The patient was admitted for <test> evaluation </test> of <problem> her fall </problem> and to rule out <problem> syncope </problem> and possible <problem> stroke </problem> with <problem> her positive histories </problem> .
<test> DIAGNOSTIC STUDIES: All x-rays </test> including <problem> left foot , right knee , left shoulder and cervical spine </problem> showed no <problem> acute fractures </problem> .
<problem> The left shoulder did show old healed left humeral head and neck fracture </problem> with <problem> baseline anterior dislocation </problem> .
<test> CT of the brain </test> showed no <problem> acute changes </problem> , <problem> left periorbital soft tissue swelling </problem> .
<test> CT of the maxillofacial area </test> showed no <problem> facial bone fracture </problem> .
<test> Echocardiogram </test> showed normal left ventricular function , <test> ejection fraction </test> estimated greater than 65% .

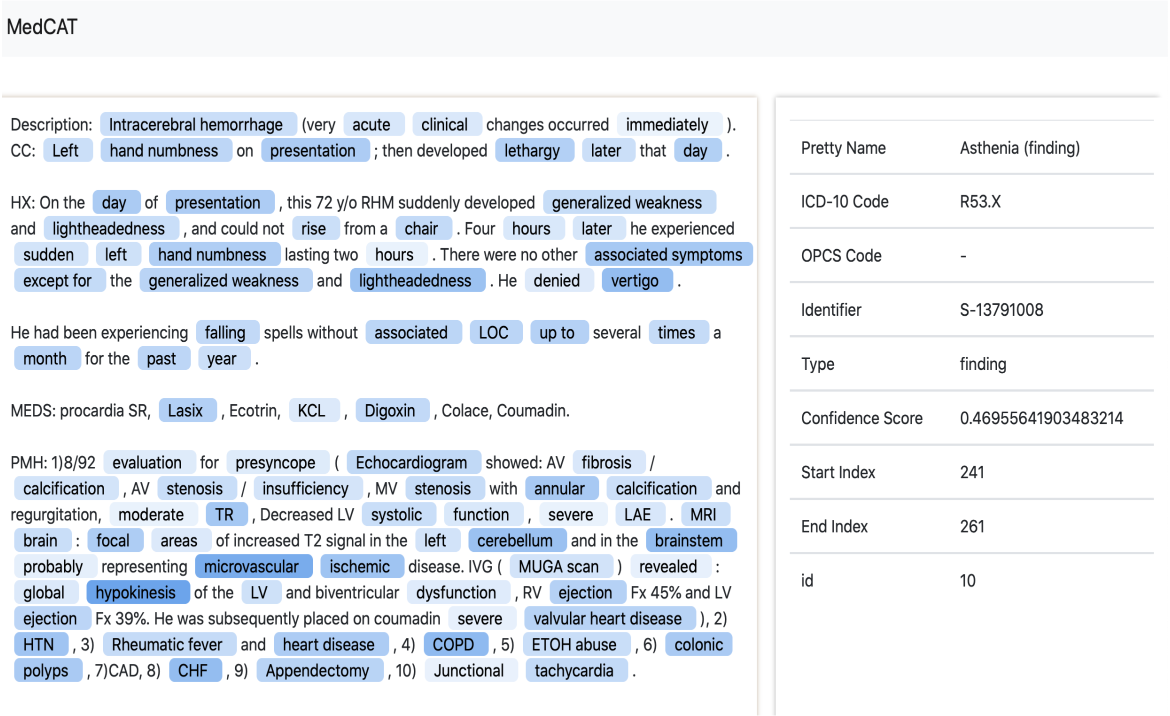
Figure 2. Clinical Named Entity Recognition and Classification usingCliNER

2. MEDCAT

The MEDCAT model is the first model we came across: https://github.com/CogStack/MedCAT.

We used the demo version of the model for initial testing: https://medcat.rosalind.kcl.ac.uk

MedCAT can be used to extract information from Electronic Health Records (EHRs) and link it to biomedical ontologies like SNOMED-CT and UMLS [8].



MedCAT

Description: Intracerebral hemorrhage (very acute clinical changes occurred immediately).
CC: Left hand numbness on presentation ; then developed lethargy later that day .

HX: On the day of presentation , this 72 y/o RHM suddenly developed generalized weakness and lightheadedness , and could not rise from a chair . Four hours later he experienced sudden left hand numbness lasting two hours . There were no other associated symptoms except for the generalized weakness and lightheadedness . He denied vertigo .

He had been experiencing falling spells without associated LOC up to several times a month for the past year .

MEDS: procardia SR, Lasix , Ecotrin, KCL , Digoxin , Colace, Coumadin.

PMH: 1)8/92 evaluation for presyncope (Echocardiogram showed: AV fibrosis / calcification , AV stenosis / insufficiency , MV stenosis with annular calcification and regurgitation, moderate TR , Decreased LV systolic function , severe LAE . MRI brain : focal areas of increased T2 signal in the left cerebellum and in the brainstem probably representing microvascular ischemic disease. IVG (MUGA scan) revealed : global hypokinesis of the LV and biventricular dysfunction , RV ejection Fx 45% and LV ejection Fx 39%. He was subsequently placed on coumadin severe valvular heart disease), 2) HTN , 3) Rheumatic fever and heart disease , 4) COPD , 5) ETOH abuse , 6) colonic polyps , 7)CAD, 8) CHF , 9) Appendectomy , 10) Junctional tachycardia .

Pretty Name	Asthenia (finding)
ICD-10 Code	R53.X
OPCS Code	-
Identifier	S-13791008
Type	finding
Confidence Score	0.46955641903483214
Start Index	241
End Index	261
id	10

Figure 3. MEDCAT Annotation results

The various information reflected on the left panel for each tag is defined in the table below:

Table 1: MedCat Tagged element descriptions

Concept Detail	Description
Annotated Text	The text span linked to the concept
Name	The linked concept name from within the MedCAT CDB
Term ID	The higher-level group of concepts that this concept sits under. This may be 'N/A' depending on if your CDB is complete with TUIs.
Concept ID	The unique identifier for this linked concept from the MedCAT CDB.
Accuracy	The MedCAT found the accuracy of the linked concept for this span. Text spans will have an accuracy of 1.0 if they are uniquely identified by that name in the CDB
Description	The MedCAT associated description of the concept.

- **Advantages:**
  - It is a powerful system:
    - From our initial testing of MEDCAT, it was understood that for tagging and extracting relevant clinical text information from free text without any necessary preprocessing of the data necessary.
    - All medical terms are tagged with relevant classifiers attached to them along with additional information about the term derived from the Snomed CT ontology.
- **Disadvantages:**
  - No semantic understanding of the text:
    - This makes it difficult to use for our specific purposes. As we are required to extract the relevant details and terms from a discharge summary of a patient to present the minimum amount of words needed for the doctor to have an overall idea of the patient's history. Without semantic understanding and the context in which words are being used this will be difficult to move forward.

As MEDcat is ultimately a web application rather than an algorithm therefore we think that it will not be possible to integrate its services straightforwardly

3.CliNER

The repository for the CliNER model can be accessed from https://github.com/text-machine-lab/CliNER

The demo of the model can be used at http://text-machine.cs.uml.edu/cliner/

Clinical Named Entity Recognition system (CliNER) is an open-source natural language processing system for named entity recognition in the clinical text of electronic health records. CliNER system was designed to follow the best practices in clinical concept extraction, as established in the i2b2 2010 shared task.

We cloned the CliNER repository and converted the discharge summaries provided by Numeh Health into the i2b2 format, as 12b2 format data is unavailable for external use.

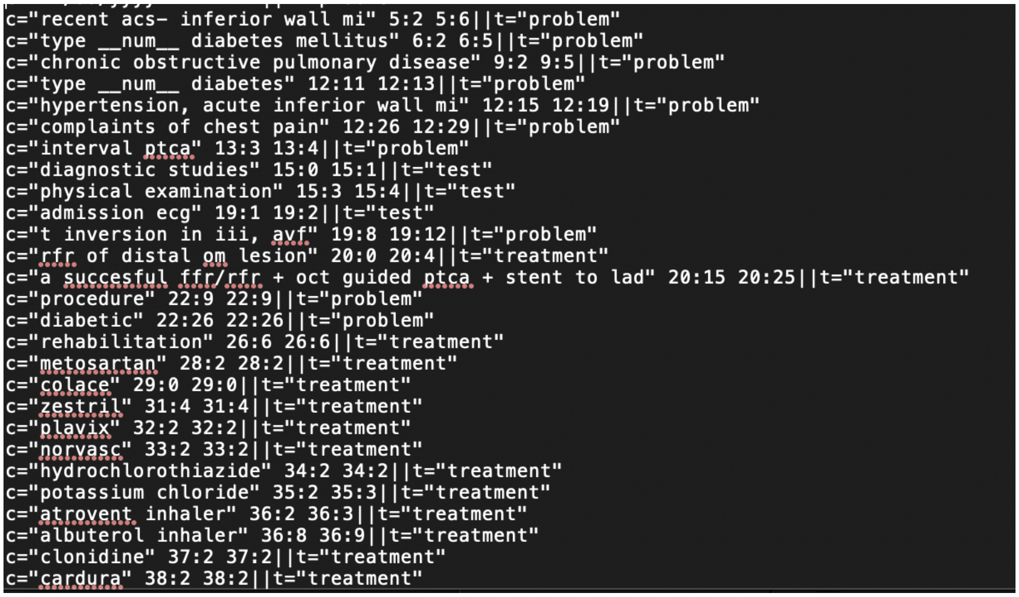
The CliNER model we used was pretrained on i2b2 discharge summaries and when we tested it after preprocessing our available discharge summaries into the i2b2 format, the tags it generated were promising however not satisfactory.

I2B2 Format for Discharge Summary:

```
DATE OF ADMISSION: MM/DD/YYYY

DATE OF DISCHARGE: MM/DD/YYYY

DISCHARGE DIAGNOSES :



CONSULTANTS:

PROCEDURES:

BRIEF HISTORY:

HOSPITAL COURSE:

DISCHARGE DISPOSITION:

ACTIVITY :

DIET :

MEDICATIONS :

FOLLOWUP :
```

Results:

Figure 4.CliNER Model Results



```
c="recent acs- inferior wall mi" 5:2 5:6||t="problem"
c="type __num__ diabetes mellitus" 6:2 6:5||t="problem"
c="chronic obstructive pulmonary disease" 9:2 9:5||t="problem"
c="type __num__ diabetes" 12:11 12:13||t="problem"
c="hypertension, acute inferior wall mi" 12:15 12:19||t="problem"
c="complaints of chest pain" 12:26 12:29||t="problem"
c="interval ptca" 13:3 13:4||t="problem"
c="diagnostic studies" 15:0 15:1||t="test"
c="physical examination" 15:3 15:4||t="test"
c="admission ecg" 19:1 19:2||t="test"
c="t inversion in iii, avf" 19:8 19:12||t="problem"
c="rfr of distal om lesion" 20:0 20:4||t="treatment"
c="a succesful ffr/rfr + oct guided ptca + stent to lad" 20:15 20:25||t="treatment"
c="procedure" 22:9 22:9||t="problem"
c="diabetic" 22:26 22:26||t="problem"
c="rehabilitation" 26:6 26:6||t="treatment"
c="metosartan" 28:2 28:2||t="treatment"
c="colace" 29:0 29:0||t="treatment"
c="zestril" 31:4 31:4||t="treatment"
c="plavix" 32:2 32:2||t="treatment"
c="norvasc" 33:2 33:2||t="treatment"
c="hydrochlorothiazide" 34:2 34:2||t="treatment"
c="potassium chloride" 35:2 35:3||t="treatment"
c="atrovent inhaler" 36:2 36:3||t="treatment"
c="albuterol inhaler" 36:8 36:9||t="treatment"
c="clonidine" 37:2 37:2||t="treatment"
c="cardura" 38:2 38:2||t="treatment"
```

The CliNER model trained using the i2B2 format for datasets and Electronics Health

Records were then tested by converting the available Discharge Summary at Numen into the i2B2 format as shown above as input to the model.

With an increasing number of epochs in our training, it was observed that the CliNER model successfully tagged terms and phrases from the discharge summary into categories of

{ ***Problem*** - an ailment or medical condition;

Treatment - The procedure or action taken by the physician to treat the patient;

Test - Medical tests and procedures are undertaken to understand the ailment. }

- **Advantages:**
    - Ease of use
        - Cliner provides us with a major advantage which is the ease of use, as there exists a web application to test the model and a different model which can be trained and tested for our specific use cases.
    - Comprehensive Tagging
        - CliNER is capable of tagging discharge summary information into relevant and important categories of ***Problem, Treatment,*** and ***Test*** which is extremely useful for our problem statement.
- **Disadvantages:**
    - Poor Re-trainability
        - Even Though we recommend CliNER as a model it does come along with a difficult problem of poor re-trainability, as I2B2 formatted datasets are not freely available.
        - Converting existing datasets into the I2B2 format is a laborious task, which we think is unnecessary.
        - Therefore, Training the CliNER model for our specific use case of cardiology-related EHR tags is not recommended.
    - Certification is required for efficient use of CliNER for industry purposes
        - For optimal performance, CliNER requires the users to obtain a Unified Medical Language System (UMLS) license, since UMLS Metathesaurus is used as one of the knowledge sources for the above classifiers.

4. Word2Vec

After the two pretrained models were tested we decided to move forward with our learning and understanding of Natural Language Processing. We watched various NPTEL videos and youtube resources to understand the algorithm and math behind Word2Vec [9]. Word embedding refers to methods that are used to represent words numerically0]. Word2vec is a neural network structure to generate word embedding by training the model on a supervised classification problem. In Mikolov et al.,2013 [5] the paper which first presented the neural network method to represent words numerically, two model architectures were presented, ***Continuous Bag-of-Words*** model and the ***Skip-gram*** model.

Continuous Bag-of-Words**:** As a model, this predicts the current word based on the context surrounding it.

Skip-gram**:** The skip-gram model predicts the surrounding words given the current word that it has localized.

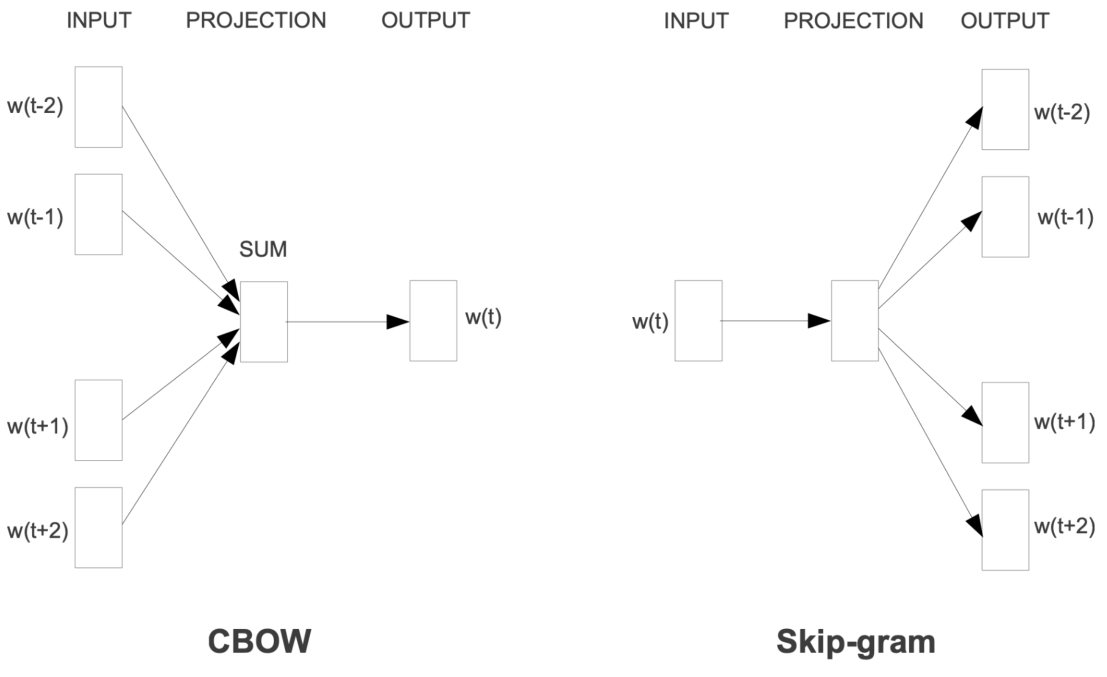


Figure 5. Diagramatic representation of Continuous Bag Of Words and Skip-gram models

They demonstrated that a shallower and more efficient model allows training on much larger amounts of data (Speed increased by 1000 !). Both the models they presented contain only one hidden layer, which leads to more efficient training.

Given below is the objective function used for ***Continuous Bag-of-Words***:

$$J = -log\widehat{P}(w_t|w_{t-n+1}...w_{t-1})$$

- Eqn 1.0

and here, the objective function for the ***Skip-Gram*** :

$$J = \frac{1}{T}\sum_{t=1}^{T}\sum_{-n\leq j\leq n} logp(w_{t+j}|w_t)$$

- Eqn 2.0

The objective of a linear programming problem will be to maximize or to minimize its loss and the loss function is represented as the objective function of the model.

Overall, these models still can be useful. It just depends on our use and needs. Word2Vec is still quite relevant on basic models and can be used to embed sentences or documents by taking the average of word vectors in a sentence.

However, for biomedical tasks, it requires more contextual understanding and comprehension of the sentences which these models on their own cannot accomplish, but it is always a good idea to understand the basics and work our way through.

5. BERT/BIOBERT

BERT (Bidirectional Encoder Representations from Transformers) is a recent paper [11] published by researchers at Google AI Language. It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Question Answering (SQuAD v1.1), Natural Language Inference (MNLI), and others.

BERT is a method of pre-training language representations, meaning that we train a general-purpose "language understanding" model on a large text corpus (like Wikipedia, PubMed), and then use that model for downstream NLP tasks that we require (like question answering, Named Entity Recognition, Classification Tasks). BERT outperforms previous methods because it is the first unsupervised, deeply bidirectional system for pre-training NLP.

BioBERT is a biomedical language representation model designed for biomedical text mining tasks such as biomedical named entity recognition, relation extraction, question answering, etc [12]. This project was undertaken by the engineers over at DMIS-Lab and they have provided weighted models trained over PubMed datasets which we can then exploit and tweak for our purposes.

The GitHub repository to access and run BioBERT is available at https://github.com/dmis-lab/biobert

Tasks Performed:

We successfully cloned the repository and installed all the requirements to run the models in our local system. The requirements.txt had to be edited to meet the specifications of our systems as it used TensorFlow-GPU which was not supported.

We created a virtual environment of python<=3.7 since the latest version of python is 3.9.

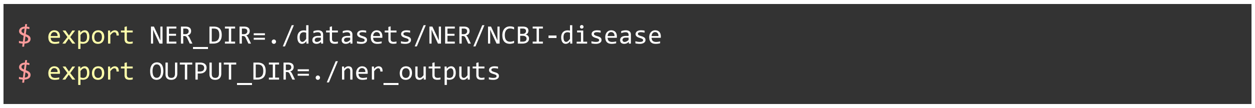
We used a 16Gb Macbook Pro to fine-tune and run the models specific to, **Relation Extraction** using BioBERT and **Named Entity Recognition** using BioBERT.

For our problem statement, we will dive deeper into **Named Entity Recognition** using BioBERT.

For fine-tuning the existing BioBERT model to accomplish **Named Entity Recognition,** the repo provides various datasets about Gene Tagging, NCBI Diseases, JNLPBA, BC5CDR-diseases, etc. We used the NCBI Disease dataset to train the model.

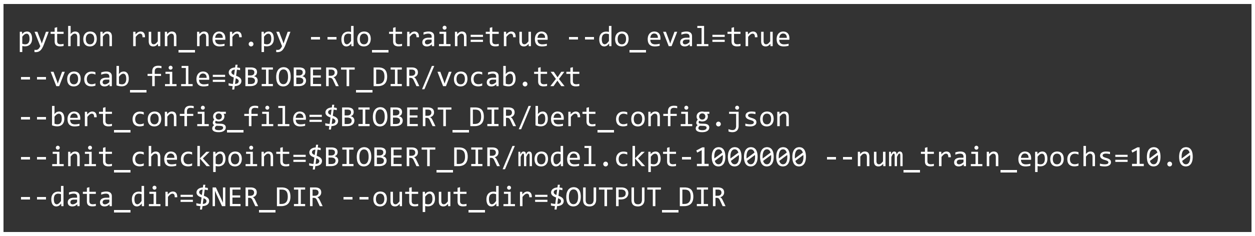
We require Shell script, Perl, and Python to run and use the BioBERT models for evaluation and testing. We set up the local system to train for **Named Entity Recognition** which took about 6 hours to complete.

The $NER_DIR indicates a folder for a single NER dataset which contains train_dev.tsv, train.tsv, devel.tsv and test.tsv. Also, we set $OUTPUT_DIR as a directory for NER outputs (trained models, test predictions, etc). For example, when fine-tuning on the NCBI disease corpus,



```
$ export NER_DIR=./datasets/NER/NCBI-disease
$ export OUTPUT_DIR=./ner_outputs
```

Following command runs fine-tuning code on NER with default arguments.

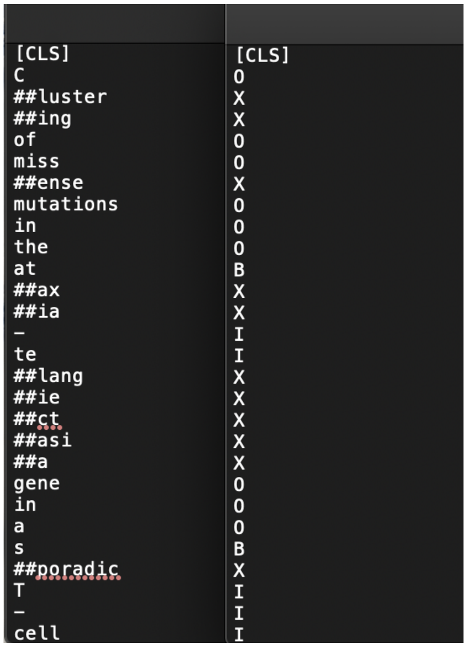


```
python run_ner.py --do_train=true --do_eval=true
--vocab_file=$BIOBERT_DIR/vocab.txt
--bert_config_file=$BIOBERT_DIR/bert_config.json
--init_checkpoint=$BIOBERT_DIR/model.ckpt-1000000 --num_train_epochs=10.0
--data_dir=$NER_DIR --output_dir=$OUTPUT_DIR
```

Once the fine tuning is complete, you can change the arguments as you want. Once you have trained your model, you can use it in inference mode by using --do_train=false --do_predict=true for evaluating test.tsv.

We faced various problems during this process as it took over 6 hours for each model to train, and at certain instances retraining from scratch was necessary as certain files got corrupted or due to the computational power required system crashes occurred frequently.

Note that this result is the token-level evaluation measure while the official evaluation should use the entity-level evaluation measure. The results of python run_ner.py will be recorded as two files: token_test.txt and label_test.txt in $OUTPUT_DIR.



```
[CLS]         [CLS]
C             O
##luster      X
##ing         X
of            O
miss          O
##ense        X
mutations     O
in            O
the           O
at            B
##ax          X
##ia          X
-             I
te            I
##lang        X
##ie          X
##ct          X
##asi         X
##a           X
gene          O
in            O
a             O
s             B
##poradic     X
T             I
-             I
cell          I
```

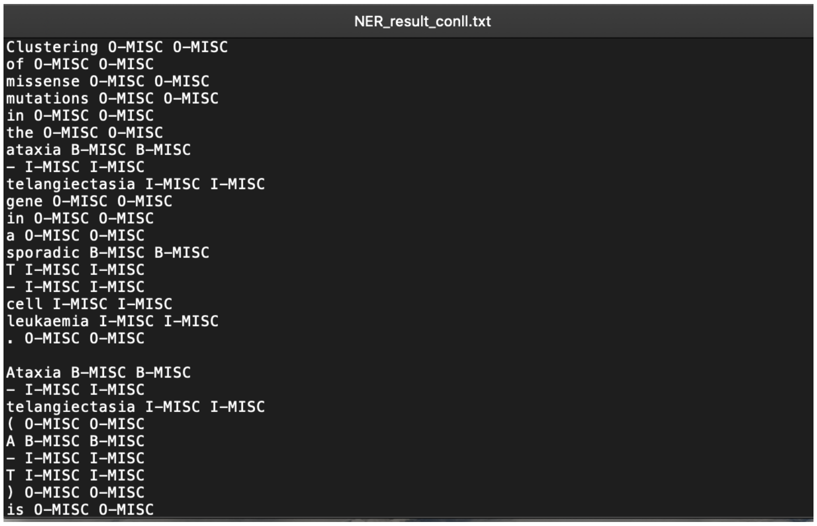
Figure 6. Token-level evaluation of BioBERT NER, with labe_testl.txt and token_test.txt

Use ./biocodes/ner_detokenize.py to obtain word level prediction file.

```
python biocodes/ner_detokenize.py --token_test_path=$OUTPUT_DIR/token_test.txt
--label_test_path=$OUTPUT_DIR/label_test.txt --answer_path=$NER_DIR/test.tsv
--output_dir=$OUTPUT_DIR
```

This will generate NER_result_conll.txt in $OUTPUT_DIR.

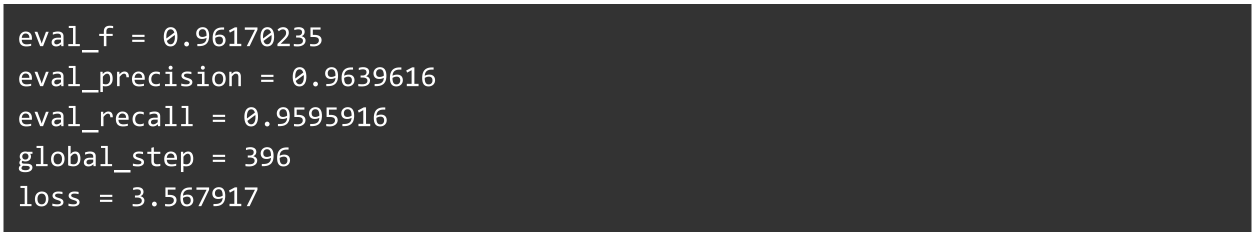
Figure 7. Entity level evaluation using BioBERT NER, with BIO tagging.



```
Clustering O-MISC O-MISC
of O-MISC O-MISC
missense O-MISC O-MISC
mutations O-MISC O-MISC
in O-MISC O-MISC
the O-MISC O-MISC
ataxia B-MISC B-MISC
- I-MISC I-MISC
telangiectasia I-MISC I-MISC
gene O-MISC O-MISC
in O-MISC O-MISC
a O-MISC O-MISC
sporadic B-MISC B-MISC
T I-MISC I-MISC
- I-MISC I-MISC
cell I-MISC I-MISC
leukaemia I-MISC I-MISC
. O-MISC O-MISC

Ataxia B-MISC B-MISC
- I-MISC I-MISC
telangiectasia I-MISC I-MISC
( O-MISC O-MISC
A B-MISC B-MISC
- I-MISC I-MISC
T I-MISC I-MISC
) O-MISC O-MISC
is O-MISC O-MISC
```

From the above results, we were able to infer that BioBERT is capable of tagging relevant medical text data in the BIO format. The BIO / IOB format (short for inside, outside, beginning) is a common tagging format for tagging tokens in a chunking task in computational linguistics (ex. named-entity recognition). The B- prefix before a tag indicates that the tag is the beginning of a chunk, and an I- prefix before a tag indicates that the tag is inside a chunk. The B- tag is used only when a tag is followed by a tag of the same type without O tokens between them. An O tag indicates that a token belongs to no entity/chunk [13].

The accuracy and precision of our trained model was obtained as follows:



```
eval_f = 0.96170235
eval_precision = 0.9639616
eval_recall = 0.9595916
global_step = 396
loss = 3.567917
```

We believe with a more powerful system and specific data points, our results will be better.

- **Advantages:**
  - Robust Clinical Text tagging:
    - As the BioBERT model is trained and maintained by DMIS-Lab using google’s BERT as a base, the clinical text tagging is very robust and we did not find any errors or mistagging, frequent errors we experienced from the previous models
  - Comprehensive documentation:
    - As there exists comprehensive documentation and relevant issue resolutions promptly, the Github repository and community are very active and hence the integration of BioBERT will be a simpler task.
- **Disadvantages:**
  - Datasets are not freely available for Cardiology related matters:
    - Even Though BioBERT is fine-tunable, it is not easy to collect datasets related to Cardiology and train the model as they are not freely available for our purposes.
  - Requires powerful systems for training purposes

    - Since training is a CPU-intensive task, it would require higher performance systems to train and evaluate regularly.
  - Difficulty in understanding the results of the evaluation
    - As this is the first time we are being acquainted with Natural Language Processing, it was a difficult task to understand the format in which the results were presented to us, hence we were not able to figure out a pipeline to present the extracted entities as highlights within the data as presented by CliNER and MEDCAT.

We recommend BioBERT:

BioBERT for this particular problem statement as it is robust and comprehensive in its tasks however, due to the lapses in our understanding we are unable to build upon this at the moment and would like to revisit this model at a later stage.

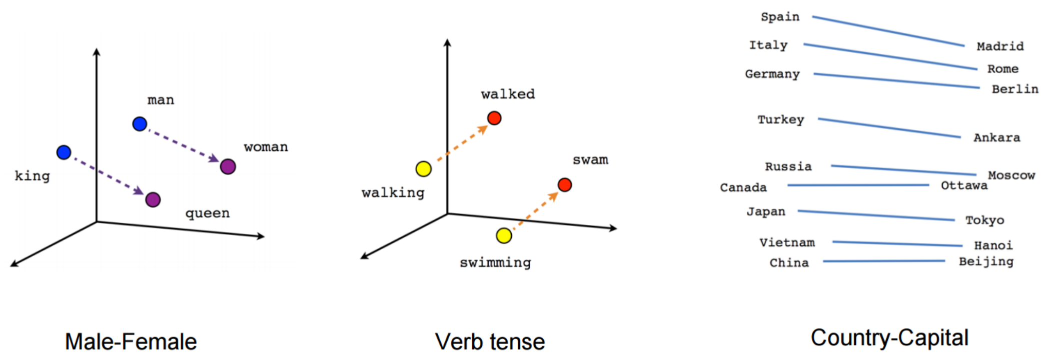
6. Sentence Embedding

Tasks Performed:

With BioBERT as an option, we moved on to try and figure out our pipeline to tag relevant medical terms from free text EHR's. We started researching **Sentence Embedding.** Sentence embedding techniques represent entire sentences and their semantic information as vectors. This helps the machine in understanding the context, intention, and other nuances in the entire text [14].

The initial embedding techniques dealt with only words. Given a set of words, you would generate an embedding for each word in the set.

Figure 8. Word embedding vector representation



Male-Female

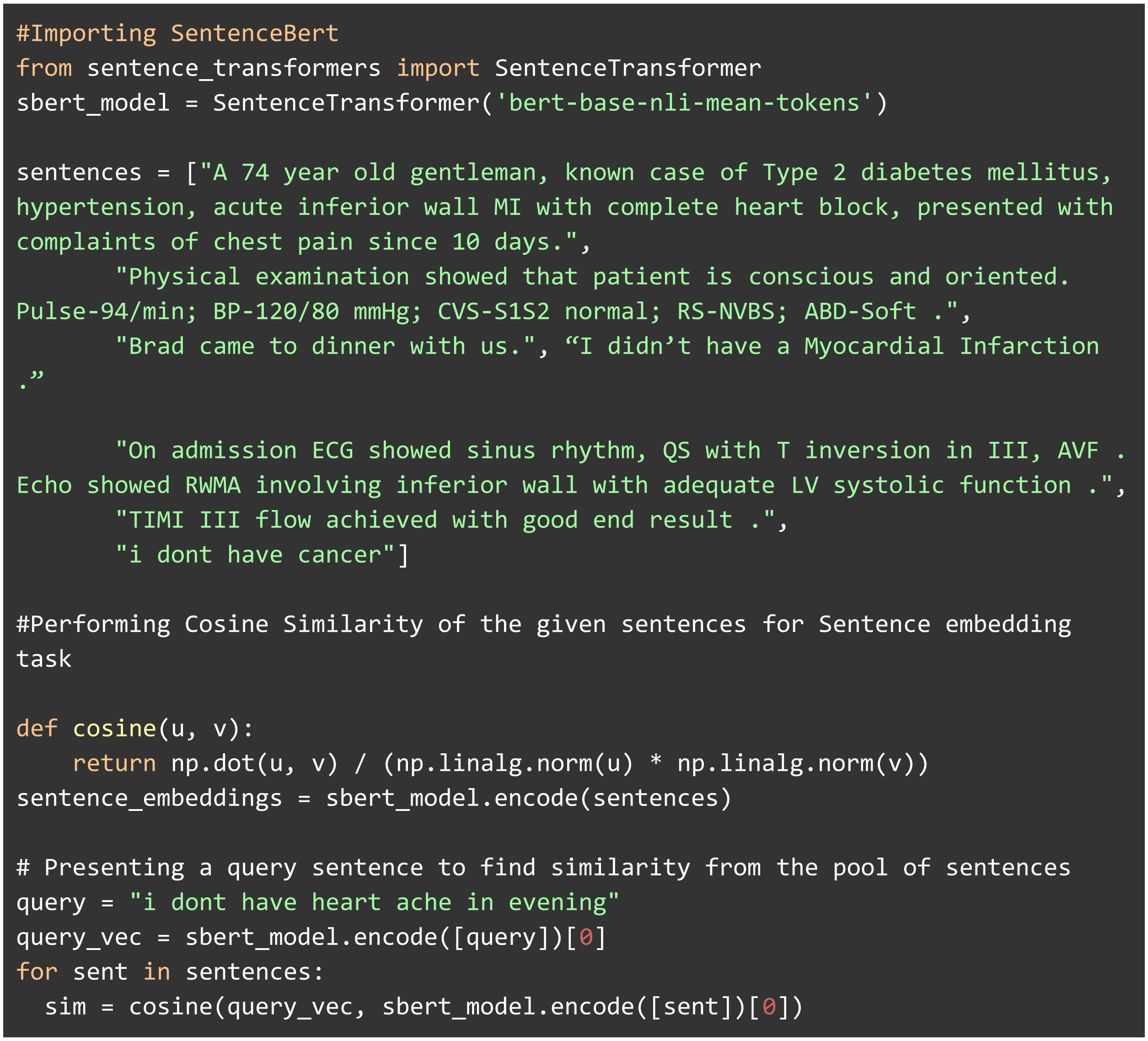
Verb tense

Country-Capital

In the case of large text, using only words would be very tedious and we would be limited by the information we can extract from the word embeddings.

We used python and Jupyter notebooks to run and implement pieces of code to develop an understanding of sentence embedding and how it can be used to solve our problem statement.

We tried to implement sentenceBERT. Sentence-BERT uses a Siamese network-like architecture to provide 2 sentences as input. These 2 sentences are then passed to BERT models and a pooling layer to generate their embeddings. Then use the embeddings for the pair of sentences as inputs to calculate the cosine similarity [15].



```
#Importing SentenceBert
from sentence_transformers import SentenceTransformer
sbert_model = SentenceTransformer('bert-base-nli-mean-tokens')

sentences = ["A 74 year old gentleman, known case of Type 2 diabetes mellitus,
hypertension, acute inferior wall MI with complete heart block, presented with
complaints of chest pain since 10 days.",
       "Physical examination showed that patient is conscious and oriented.
Pulse-94/min; BP-120/80 mmHg; CVS-S1S2 normal; RS-NVBS; ABD-Soft .",
       "Brad came to dinner with us.", “I didn’t have a Myocardial Infarction
.”

       "On admission ECG showed sinus rhythm, QS with T inversion in III, AVF .
Echo showed RWMA involving inferior wall with adequate LV systolic function .",
       "TIMI III flow achieved with good end result .",
       "i dont have cancer"]

#Performing Cosine Similarity of the given sentences for Sentence embedding
task

def cosine(u, v):
    return np.dot(u, v) / (np.linalg.norm(u) * np.linalg.norm(v))
sentence_embeddings = sbert_model.encode(sentences)

# Presenting a query sentence to find similarity from the pool of sentences
query = "i dont have heart ache in evening"
query_vec = sbert_model.encode([query])[0]
for sent in sentences:
  sim = cosine(query_vec, sbert_model.encode([sent])[0])
  print("Sentence = ", sent, "; similarity = ", sim)

# Results of similarity with the given sentence
Sentence =  A 74 year old gentleman, known case of Type 2 diabetes mellitus,
hypertension, acute inferior wall MI with complete heart block, presented with
complaints of chest pain since 10 days. ; similarity =  -0.009621797
Sentence =  Physical examination showed that patient is conscious and oriented.
Pulse-94/min; BP-120/80 mmHg; CVS-S1S2 normal; RS-NVBS; ABD-Soft . ; similarity
=  0.34176022
Sentence =  Brad came to dinner with us. ; similarity =  0.18913847
Sentence =  On admission ECG showed sinus rhythm, QS with T inversion in III,
AVF . Echo showed RWMA involving inferior wall with adequate LV systolic
function . ; similarity =  0.29188272
Sentence =  TIMI III flow achieved with the good end result. ; similarity =
0.4672234
Sentence = I didn’t have a Myocardial Infarction . ; similarity = 0.892899
Sentence =  i don’t have have cancer ; similarity =  0.7527088
```

Our Learnings:

We were able to build and implement a simple model that found the similarity of a target sentence with a group of sentences, this demonstrated semantic and contextual understanding of the free text by the model. This gave us some confidence to experiment with building our own models.

7. Datasets

A common problem we faced throughout all of our implementations was the lack of necessary datasets that we could use to train subsequent models for our specific use case. We explored various repositories within kaggle, 12b2 challenge datasets, MTsamples, SnomedCT, PubMed, MIMIC Datasets, etc. Most of our searches proved futile as Electronic Health Records are hard to come by and access is only granted to specific use cases which did not coincide with ours. However, we were able to extract some relevant terms with regards to Cardiac Health Care and Electronic Health Records related to cardiology, and in this section, we will discuss what we were able to achieve concerning relevant datasets.

SNOMED CT

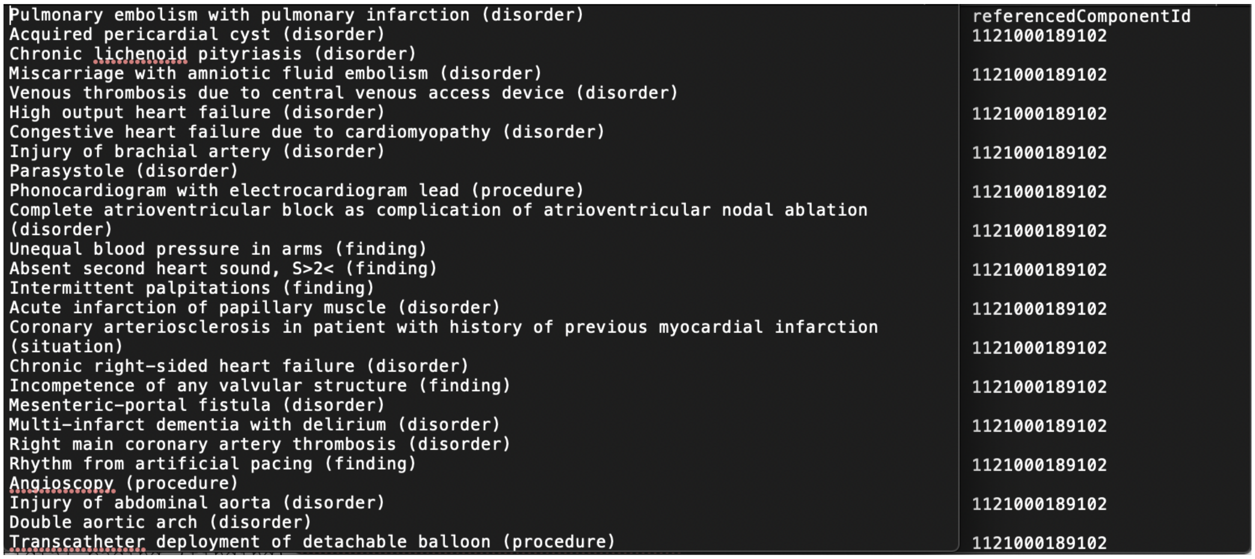
We generated a script that is capable of extracting information from the SNOMED CT browser directly as JSON files and then giving us relevant information such as:

- Concept By ID
- Description By ID
- Concepts ID By String
- Descriptions By String From Procedure

We received three refset files containing ten thousand Id's each pertaining to Cardiology procedures, Patient Instructions related to cardiothoracic health, and cardiology-related terminologies. We passed these refset IDs through our program and received three sets of large datasets consisting of words related to cardiology.

```
from urllib.request import urlopen
from urllib.parse import quote
import json
baseUrl = 'https://browser.ihtsdotools.org/snowstorm/snomed-ct'
edition = 'MAIN'
version = '2019-07-31'

#Prints fsn of a concept
def getConceptById(id):
    url = baseUrl + '/browser/' + edition + '/' + version + '/concepts/' +
id
```

```
    response = urlopen(url).read()
    data = json.loads(response.decode('utf-8'))

    return (data['fsn']['term'])

#Prints description by id
def getDescriptionById(id):
    url = baseUrl + '/' + edition + '/' + version + '/descriptions/' + id
    response = urlopen(url).read()
    data = json.loads(response.decode('utf-8'))

    print (data['term'])

#Prints number of concepts with descriptions containing the search term
def getConceptsByString(searchTerm):
    url = baseUrl + '/browser/' + edition + '/' + version +
'/concepts?term=' + quote(searchTerm) +
'&activeFilter=true&offset=0&limit=50'
    response = urlopen(url).read()
    data = json.loads(response.decode('utf-8'))

    print (data['total'])

#Prints number of descriptions containing the search term with a specific
semantic tag
def getDescriptionsByStringFromProcedure(searchTerm, semanticTag):
    url = baseUrl + '/browser/' + edition + '/' + version +
'/descriptions?term=' + quote(searchTerm) +
'&conceptActive=true&semanticTag=' + quote(semanticTag) +
'&groupByConcept=false&searchMode=STANDARD&offset=0&limit=50'
    response = urlopen(url).read()
    data = json.loads(response.decode('utf-8'))

    print (data['totalElements'])

getConceptById('100231000119101')
getDescriptionById('679406011')
getConceptsByString('attack')
getDescriptionsByStringFromProcedure('heart', 'procedure')
```



```
Pulmonary embolism with pulmonary infarction (disorder)
Acquired pericardial cyst (disorder)
Chronic lichenoid pityriasis (disorder)
Miscarriage with amniotic fluid embolism (disorder)
Venous thrombosis due to central venous access device (disorder)
High output heart failure (disorder)
Congestive heart failure due to cardiomyopathy (disorder)
Injury of brachial artery (disorder)
Parasystole (disorder)
Phonocardiogram with electrocardiogram lead (procedure)
Complete atrioventricular block as complication of atrioventricular nodal ablation
(disorder)
Unequal blood pressure in arms (finding)
Absent second heart sound, S>2< (finding)
Intermittent palpitations (finding)
Acute infarction of papillary muscle (disorder)
Coronary arteriosclerosis in patient with history of previous myocardial infarction
(situation)
Chronic right-sided heart failure (disorder)
Incompetence of any valvular structure (finding)
Mesenteric-portal fistula (disorder)
Multi-infarct dementia with delirium (disorder)
Right main coronary artery thrombosis (disorder)
Rhythm from artificial pacing (finding)
Angioscopy (procedure)
Injury of abdominal aorta (disorder)
Double aortic arch (disorder)
Transcatheter deployment of detachable balloon (procedure)
```

```
referencedComponentId
1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102

1121000189102
```

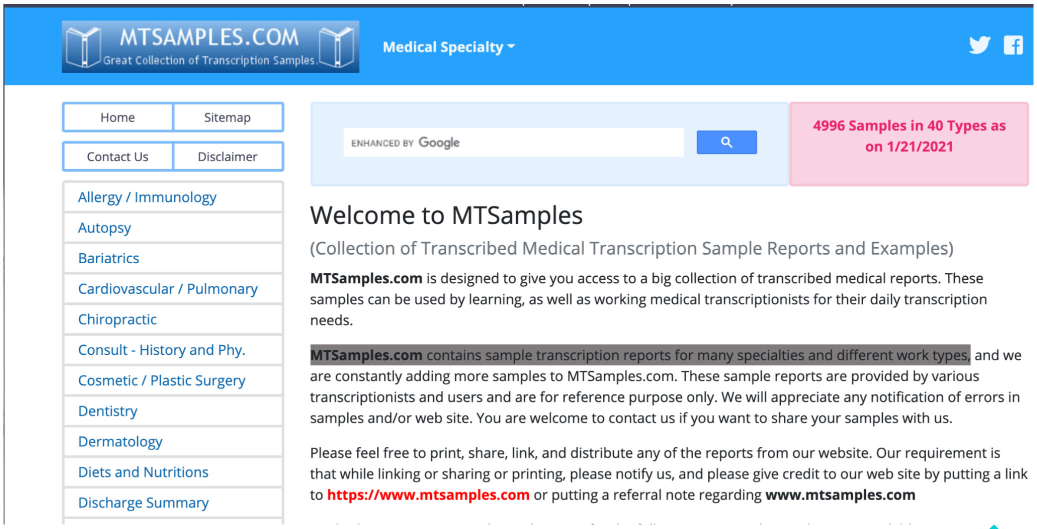
Figure 9. Extracting the concepts referenced by Id in the refsets

MTSamples

We soon came across the website MTSamples. MTSamples.com is designed to give access to a big collection of transcribed medical reports. These samples can be used by learning, as well as working medical transcriptionists for their daily transcription needs [16].

MTSamples.com contains sample transcription reports for many specialties and different work types, we made use of the Cardiovascular/Pulmonary transcriptions.

Figure 10. MTSamples Website Screenshot



4996 Samples in 40 Types as on 1/21/2021

Welcome to MTSamples

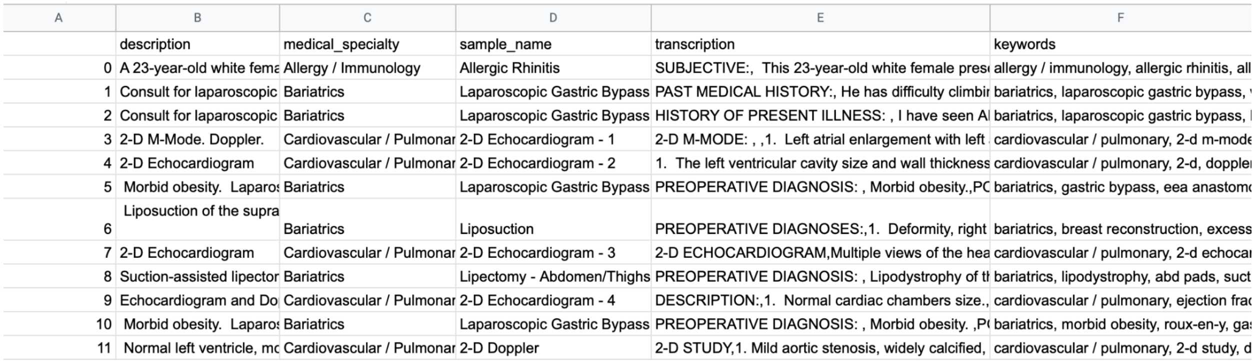
(Collection of Transcribed Medical Transcription Sample Reports and Examples)

MTSamples.com is designed to give you access to a big collection of transcribed medical reports. These samples can be used by learning, as well as working medical transcriptionists for their daily transcription needs.

MTSamples.com contains sample transcription reports for many specialties and different work types, and we are constantly adding more samples to MTSamples.com. These sample reports are provided by various transcriptionists and users and are for reference purpose only. We will appreciate any notification of errors in samples and/or web site. You are welcome to contact us if you want to share your samples with us.

Please feel free to print, share, link, and distribute any of the reports from our website. Our requirement is that while linking or sharing or printing, please notify us, and please give credit to our web site by putting a link to **https://www.mtsamples.com** or putting a referral note regarding **www.mtsamples.com**

Furthermore, we found a CSV file that contained all the discharge summaries uploaded onto MTSamples.com and we wrote a python script to extract only the cardiovascular/pulmonary related discharge summaries from the consolidated CSV. The information from those discharge summaries was then pre-processed and used to train our most recent model. The benefit of this particular CSV file was the presence of keywords concerning each discharge summary.



A	B	C	D	E	F
	description	medical_specialty	sample_name	transcription	keywords
0	A 23-year-old white fema	Allergy / Immunology	Allergic Rhinitis	SUBJECTIVE:, This 23-year-old white female pres	allergy / immunology, allergic rhinitis, all
1	Consult for laparoscopic	Bariatrics	Laparoscopic Gastric Bypass	PAST MEDICAL HISTORY:, He has difficulty climbi	bariatrics, laparoscopic gastric bypass,
2	Consult for laparoscopic	Bariatrics	Laparoscopic Gastric Bypass	HISTORY OF PRESENT ILLNESS: , I have seen A	bariatrics, laparoscopic gastric bypass,
3	2-D M-Mode. Doppler.	Cardiovascular / Pulmonar	2-D Echocardiogram - 1	2-D M-MODE: , ,1. Left atrial enlargement with left	cardiovascular / pulmonary, 2-d m-mode
4	2-D Echocardiogram	Cardiovascular / Pulmonar	2-D Echocardiogram - 2	1. The left ventricular cavity size and wall thickness	cardiovascular / pulmonary, 2-d, dopple
5	Morbid obesity. Laparo	Bariatrics	Laparoscopic Gastric Bypass	PREOPERATIVE DIAGNOSIS: , Morbid obesity.,PC	bariatrics, gastric bypass, eea anastom
6	Liposuction of the supra	Bariatrics	Liposuction	PREOPERATIVE DIAGNOSES:,1. Deformity, right	bariatrics, breast reconstruction, excess
7	2-D Echocardiogram	Cardiovascular / Pulmonar	2-D Echocardiogram - 3	2-D ECHOCARDIOGRAM,Multiple views of the hea	cardiovascular / pulmonary, 2-d echoca
8	Suction-assisted lipector	Bariatrics	Lipectomy - Abdomen/Thighs	PREOPERATIVE DIAGNOSIS: , Lipodystrophy of t	bariatrics, lipodystrophy, abd pads, suct
9	Echocardiogram and Do	Cardiovascular / Pulmonar	2-D Echocardiogram - 4	DESCRIPTION:,1. Normal cardiac chambers size.,	cardiovascular / pulmonary, ejection fra
10	Morbid obesity. Laparo	Bariatrics	Laparoscopic Gastric Bypass	PREOPERATIVE DIAGNOSIS: , Morbid obesity. ,P	bariatrics, morbid obesity, roux-en-y, ga
11	Normal left ventricle, mc	Cardiovascular / Pulmonar	2-D Doppler	2-D STUDY,1. Mild aortic stenosis, widely calcified,	cardiovascular / pulmonary, 2-d study, d

Figure 11.Excerpt of MTSamples CSV

8. Spark NLP [John Snow Lab]

We received contact information for an NLP expert from Bosch to try and get in touch with him for a professional opinion on what our next possible steps may be to solve this given problem statement. After a long conversation with him he was able to provide us with relevant links and information regarding medical text mining and NLP in the medical domain, he proceeded to also share information regarding John Snow labs and their various predefined and pre-trained models which are currently being used for NLP in the medical domain.

John Snow Labs' Spark NLP is a paid open-source text processing library for Python, Java, and Scala. It provides production-grade, scalable, and trainable versions of the latest research in natural language processing [17]. It is the most widely used NLP library in the Enterprise, by far [18]. They have a dedicated program called Spark NLP for healthcare whose integration will greatly benefit Numen Health's application. Furthermore, **we recommend the further investment of resources to Spark NLP for health care** [19].

This is a collab notebook that Automatically identifies Signs and Symptoms in clinical documents using two of their pretrained Spark NLP clinical models: Collab

9. spaCy

Tasks Performed:

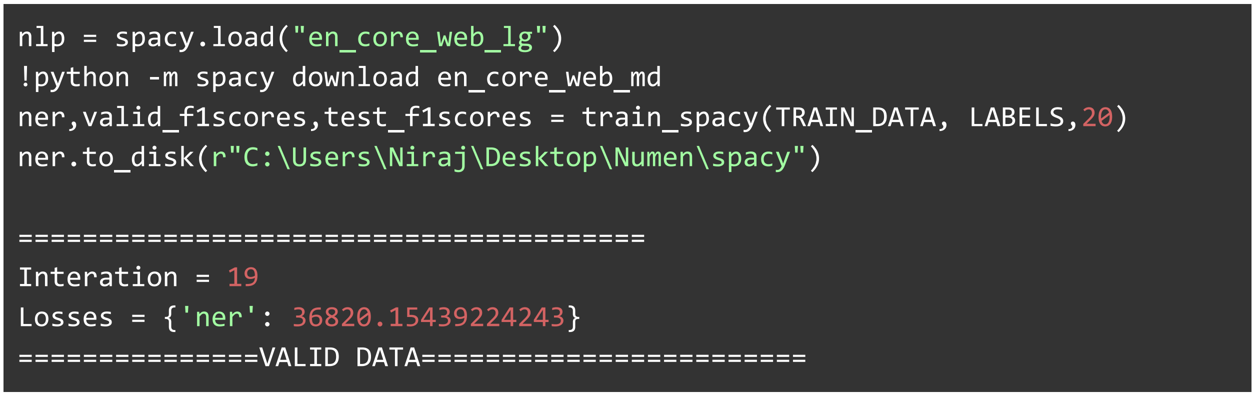
This is the last model we implemented during our time at Numen Health. spaCy is a free, open-source library for advanced Natural Language Processing (NLP) in Python. spaCy is designed specifically for production use and helps you build applications that process and “understand” large volumes of text. It can be used to build information extraction or natural language understanding systems or to pre-process text for deep learning [20].

The spaCy NER model we implemented was done with the help of RSREETech a YouTuber whose videos focus on education in the area of Data Science, Artificial Intelligence, Machine Learning, and Natural Language Processing concepts [21].

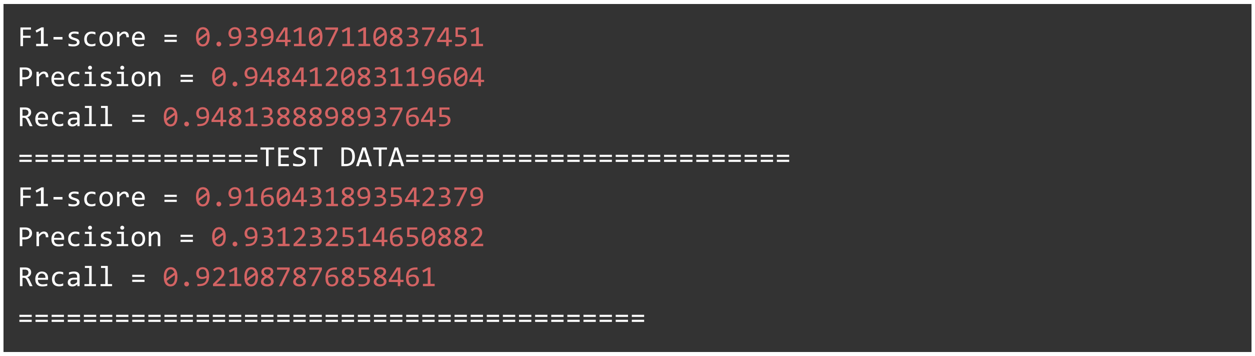
We realized that spaCy models and BioBERT both use the same type and format for their training data, so we proceeded to train the model on BioBERT data that we procured in section 5. With promising results, we added the MTSamples data we collected in section 7 into the training pool.

We used the trained pipelines and models of spaCy, in general, spaCy expects all pipeline packages to follow the naming convention of [lang]_[name]. For spaCy’s pipelines, the names are divided into three components:

1. **Type:** Capabilities (e.g. core for general-purpose pipeline with vocabulary, syntax, entities, and word vectors, or dep for only vocab and syntax).
2. **Genre:** Type of text the pipeline is trained on, e.g. web or news.
3. **Size:** Package size indicator, sm, md or lg.



```
nlp = spacy.load("en_core_web_lg")
!python -m spacy download en_core_web_md
ner,valid_f1scores,test_f1scores = train_spacy(TRAIN_DATA, LABELS,20)
ner.to_disk(r"C:\Users\Niraj\Desktop\Numen\spacy")

=======================================
Interation = 19
Losses = {'ner': 36820.15439224243}
===============VALID DATA========================
```



```
F1-score = 0.9394107110837451
Precision = 0.948412083119604
Recall = 0.9481388898937645
===============TEST DATA========================
F1-score = 0.9160431893542379
Precision = 0.931232514650882
Recall = 0.921087876858461
=======================================
```

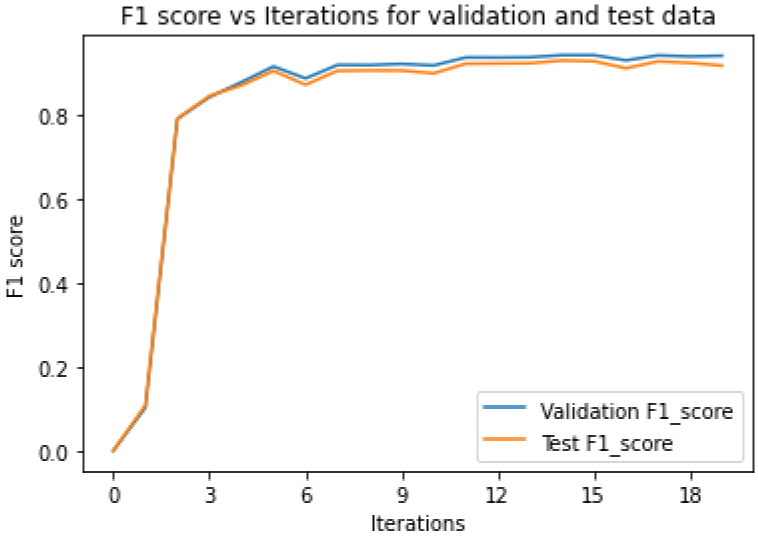
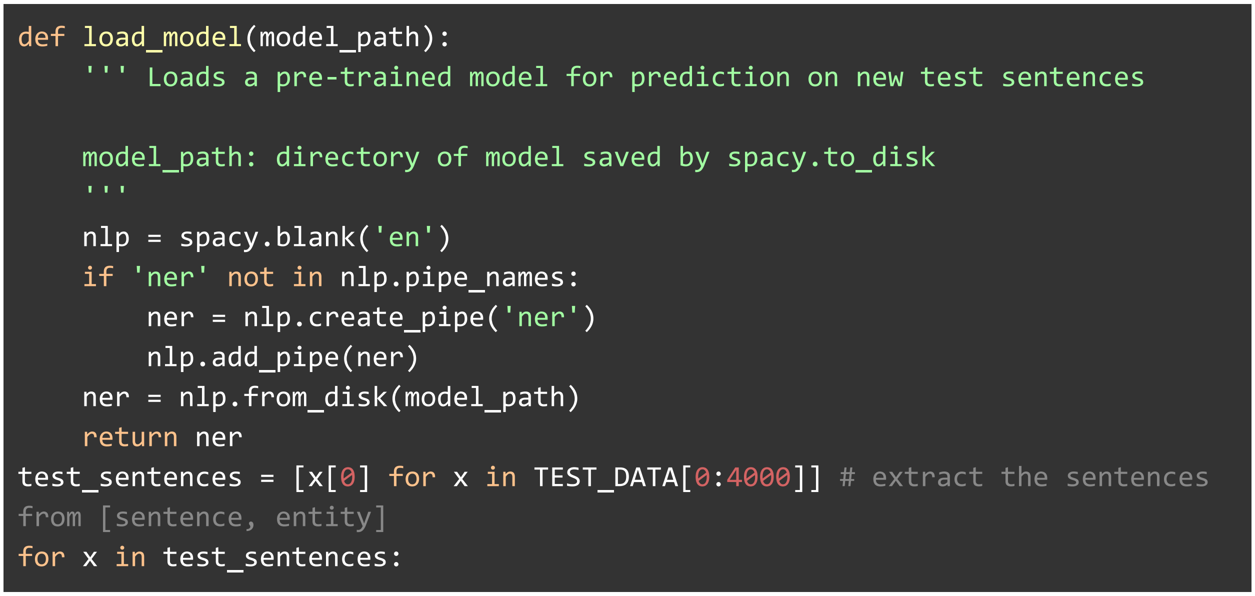
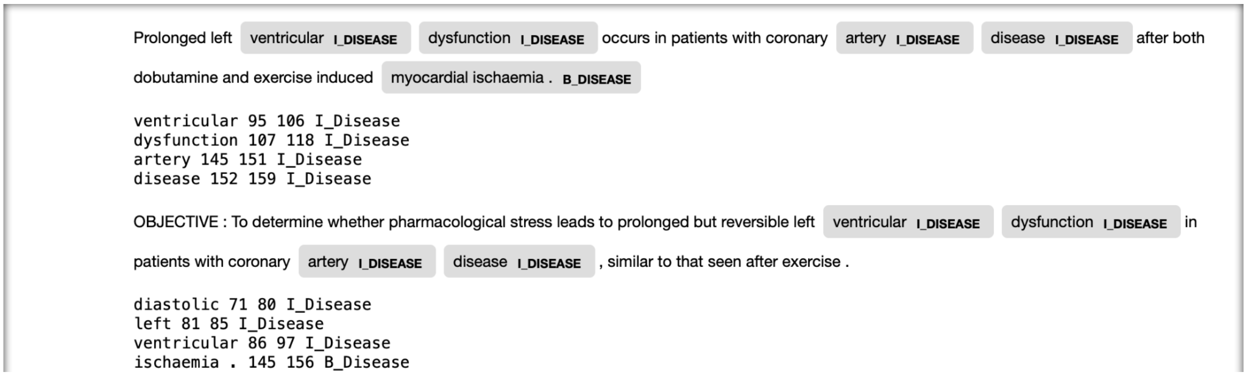


Figure 12 The efficiency of Training Graph



```
def load_model(model_path):
    ''' Loads a pre-trained model for prediction on new test sentences

    model_path: directory of model saved by spacy.to_disk
    '''
    nlp = spacy.blank('en')
    if 'ner' not in nlp.pipe_names:
        ner = nlp.create_pipe('ner')
        nlp.add_pipe(ner)
    ner = nlp.from_disk(model_path)
    return ner
test_sentences = [x[0] for x in TEST_DATA[0:4000]] # extract the sentences
from [sentence, entity]
for x in test_sentences:
```

```
doc = ner(x)
for ent in doc.ents:
    print(ent.text, ent.start_char, ent.end_char, ent.label_)
displacy.render(doc,jupyter=True, style = "ent")
```



Prolonged left ventricular I_DISEASE dysfunction I_DISEASE occurs in patients with coronary artery I_DISEASE disease I_DISEASE after both dobutamine and exercise induced myocardial ischaemia . B_DISEASE

```
ventricular 95 106 I_Disease
dysfunction 107 118 I_Disease
artery 145 151 I_Disease
disease 152 159 I_Disease
```

OBJECTIVE : To determine whether pharmacological stress leads to prolonged but reversible left ventricular I_DISEASE dysfunction I_DISEASE in patients with coronary artery I_DISEASE disease I_DISEASE , similar to that seen after exercise .

```
diastolic 71 80 I_Disease
left 81 85 I_Disease
ventricular 86 97 I_Disease
ischaemia . 145 156 B_Disease
```

Figure 13 .NER annotated results

As you can see from the above results the free text is tagged and annotated using BIO tags as introduced in section 5 (BioBERT). In the context of this model let us assume that the keywords to be tagged are “heart attack”, we trained this specific spaCy model to first tag ‘heart’ as B and ‘attack’ I, and all miscellaneous words surrounding the keywords ‘O’. As mentioned in section 5, B refers to the beginning of a chunk and I inside.

After successfully annotating the free text we realized that the capabilities of this particular model can be enhanced with the addition of POS [Parts Of Speech] tagging, wherein we get the position of each tagged keyword and search for 2-3 words around the tagged keyword to look for parts of speech that describe the relationship between various keywords and find the relevance of the tagged keyword from a doctor’s perspective. For example “'Congestive heart failure due to cardiomyopathy”, the model would just tag Congestive Heart Failure and Cardiomyopathy, with the addition of POS tagging this phrase in its entirety will be tagged and annotated.

For the model to be able to identify and tag Parts of speech, we made use of NLTK. NLTK, the Natural Language Toolkit, is a suite of open source program modules, tutorials, and problem sets, providing ready-to-use computational linguistics courseware. NLTK covers symbolic and statistical natural language processing and is interfaced with annotated corpora [22].

The POS tagger in the NLTK library outputs specific tags for certain words. The list of POS tags is as follows, with examples of what each POS stands for.

- CC coordinating conjunction
- CD cardinal digit
- DT determiner
- EX existential there (like: “there is” … think of it like “there exists”)
- FW foreign word
- IN preposition/subordinating conjunction
- JJ adjective ‘big’
- JJR adjective, comparative ‘bigger’
- JJS adjective, superlative ‘biggest’
- LS list marker 1)
- MD modal could, will
- NN noun, singular ‘desk’
- NNS noun plural ‘desks’
- NNP proper noun, singular ‘Harrison’
- NNPS proper noun, plural ‘Americans’
- PDT predeterminer ‘all the kids’
- POS possessive ending parent’s
- PRP personal pronoun I, he, she
- PRP$ possessive pronoun my, his, hers
- RB adverb very, silently,
- RBR adverb, comparative better
- RBS adverb, superlative best
- RP particle give up
- TO, to go ‘to the store.
- UH interjection, errrrrrrrm
- VB verb, base form take
- VBD verb, past tense, took
- VBG verb, gerund/present participle taking
- VBN verb, the past participle is taken
- VBP verb, sing. present, known-3d take
- VBZ verb, 3rd person sing. present takes
- WDT wh-determiner which
- WP wh-pronoun who, what
- WP$ possessive wh-pronoun whose
- WRB wh-adverb where, when

[23]

```
import re
import nltk
import pickle
Sentence =”The patient suffered from a mild myocardial infarction in the hospital and has diabetes mellitus.”
str1 =  sentence
for k in range (len(final_keys)):
    sub = '\w*\W*\w*\W*'+final_keys[k]
    for i in re.findall(sub, str1, re.I):
        i=i.strip(" .")

    x=i.split()
    x_tag=nltk.pos_tag(x)
    #print(x_tag)

    comparison=['VBP','VBN','VBG','VB','JJ','JJR','JJS','DT','VBZ'] #these are POS tags for different types of verbs, adjectives and determiner
    finkey=[]
    count=0
    fkey=""
    for i in range (len(x_tag)):
        if(count==2):
            fkey=fkey+final_keys[k].strip()+" "
            break
        if(x_tag[i][1]in comparison):
            fkey=fkey+x_tag[i][0]+" "
        count=count+1
    print(fkey)

#RESULTS
['a mild myocardial infarction ', 'has diabetes mellitus ']
```

- **Advantages:**
    - It is a simple model:

        - With all the knowledge that we gained over one month, we were able to design this pipeline from scratch and enhance its capabilities
    - Results look promising
        - With our initial testing, we found that it was successfully tagging relevant clinical terms and extracting phrases that ultimately provide a rudimentary representation for the solution to the initial problem statement.
- **Disadvantages:**
    - Lack of time to improve:
        - As our internship has come to an end, we did not have sufficient time to explore the bounds of this model and further improve it to create a robust pipeline fit for integration.
    - Very Sensitive to the dataset we provide :
        - Since we have not been able to acquire a substantial amount of datasets specifically related to cardiology this model was not able to pick up on certain keywords and we believe improving this model is synonymous with acquiring a large dataset that is cardiology specific.
    - Other Models are more powerful:
        - With the extensive studies that we have undertaken for each model so far, we cannot confidently recommend the spaCy model as the best fit for implementation. We have used this model only to describe and present a proof of concept that solves the problem statement presented to us at the beginning of this internship.

CLOSING REMARKS

The research and implementation undertaken by the internship team for one month resulted in the understanding of Natural Language Processing from its basics to applications in the medical domain. We were able to explore various industry-standard models and libraries that aid and improve Data and text mining on Electronic Health Records, a task that comes along with its own set of discrepancies and unavoidable roadblocks. We discovered the various resources that make medical and clinical data including discharge summaries available for research and testing purposes. Concerning adding the text mining feature to the Numen Health application, we insist that the engineering team look into section 5 and section 8 of this document and refer to section 7 for Dataset-related queries.

REFERENCES

1. https://www.nrces.in/standards/snomed-ct

2. https://youtu.be/hdYXuNbPLFo

3. https://www.snomed.org/snomed-ct/five-step-briefing

4. Ignacio Martinez Soriano, Juan Luis Castro Peña. (2017) Automatic Medical Concept Extraction from Free Text Clinical Reports, a New Named Entity Recognition Approach. International Journal of Computers, 2, 38-46

5. Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey, Efficient estimation of word representations in vectorspace. arXiv:1301.3781, 2013a

6. ShaodianZhang,NóemieElhadad,Unsupervised Biomedical Named Entity Recognition:Experiments with Clinical and Biological Texts, J Biomed Inform. 2013.

7. Chen Y, Lasko TA, Mei Q, Denny JC, Xu H. A Study of Active Learning Methods for Named Entity Recognition in Clinical Text. Journal of biomedical informatics. 58:11-18. 2015.

8. Multi-domain Clinical Natural Language Processing with MedCAT: the Medical Concept Annotation Toolkit Zeljko Kraljevic, Thomas Searle, Anthony Shek, Lukasz Roguski, Kawsar Noor, Daniel Bean, Aurelie Mascio, Leilei Zhu, Amos A Folarin, Angus Roberts, Rebecca Bendayan, Mark P Richardson, Robert Stewart, Anoop D Shah, Wai Keong Wong, Zina Ibrahim, James T Teo, Richard JB Dobson

9. https://www.youtube.com/watch?v=5hKxvh4RAsY&list=PLyqSpQzTE6M_EcNgdZ2qOtTZe7YI4Eedb

10. https://towardsdatascience.com/word2vec-from-scratch-with-numpy-8786ddd49e72

11. Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." *arXiv preprint arXiv:1810.04805* (2018).

12. Lee, Jinhyuk, et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining." *Bioinformatics* 36.4 (2020): 1234-1240.

13. BIO / IOB Tagged Text to Original Text Medium Article

14. Sentence Embedding Techniques Analytics Vidhya Article

15. Reimers, Nils, and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks." *arXiv preprint arXiv:1908.10084* (2019).

16. https://www.mtsamples.com/

17. https://www.johnsnowlabs.com/spark-nlp

18. Gradientflow.com

19. https://www.johnsnowlabs.com/spark-nlp-health/

20. Neumann, Mark, et al. "Scispacy: Fast and robust models for biomedical natural language processing." *arXiv preprint arXiv:1902.07669* (2019).

21. https://www.youtube.com/watch?v=DxLcMI-EMYI

22. ETMTNLP '02: Proceedings of the ACL-02 Workshop on Effective Tools and methodologies for teaching natural language processing and computational linguistics - Volume 1 July 2002 Pages 63–70https://doi.org/10.3115/1118108.1118117

23. https://medium.com/@muddaprince456/categorizing-and-pos-tagging-with-nltk-python-28f2bc9312c3