Deep Learning on Clinical Notes

Adam Lieberman, Ravish Chawla, & Garrett Mallory

Abstract— **There are over 5,500 registered hospitals in the United States. When a patient visits a hospital, clinical notes are taken based off on patient symptoms, lab results, and surgical notes. These notes help doctors actively diagnose their patients. However, clinical notes vary from hospital to hospital and interoperability standards are not preserved in the sharing of these electronic health records. Additionally, clinical records can be messy from a data stand point and be difficult to interpret. Thus, there are often incorrect diagnoses. This project aims to leverage the power of the MIMIC III database, deep learning, and big data technologies like Spark, Scala, Python, and Microsoft Azure to build a system to assist doctors in accurately diagnosing patients.**

I. OVERVIEW

In this paper, we detail the construction of Dr. ANN, our web-based deep learning system. The system inputs text data as represented by a clinical note and outputs a diagnosis for that clinical note in the form of an ICD-9 code. Additionally, keywords in the clinical note are extracted, detailing which words are most relevant for the predicted diagnosis. This system delivers a few main advantages:

1) The system can serve as a second opinion for a doctor who has a diagnosis in mind.
2) Our system can review thousands of clinical notes and diagnose a patient before a doctor can review and diagnose a single patient.
3) New data can constantly be trained on and we can continuously update the system to achieve more accurate diagnoses.
4) Key concepts relating to the predicted diagnosis from a clinical note query can be extracted.

Currently, there has been a great deal of research on using deep learning, particularly using recurrent neural networks (RNNs) to diagnose patients, but no system has been deployed in a hospital or doctor's office. This is partly due to machine error. Computers are not perfect and do make mistakes. Diagnosing a patient can be a life-threatening event and to put this fate in a computer's hands can be quite dangerous. However, a computer trained to recommend diagnoses like a doctor can serve as a second opinion. The deep learning algorithm could potentially discover diagnoses the doctor never considered and could serve as a great point of reference to assist the doctor in diagnosing patients. We explore this ongoing effort through two model pipelines:

1) Using a term frequency—inverse document frequency (tf-idf) feature vector as input to a Long Short Term Memory (LSTM) model to obtain a diagnosis code. Then, using a conditional random field (CRF) on the clinical note to extract the most relevant words from the note relating to the diagnosis.
2) Using a tf-idf feature vector as input to an an LSTM model with an attention mechanism to obtain a diagnosis code and the most relevant words in the clinical note pertaining to the diagnosis.

II. DATA

We will be utilizing data from MIMIC-III, a large publicly-available database comprising of de-identified health-related data associated with approximately 60,000 patients from the critical care units of Beth Israel Deaconess Medical Center over the time period from 2001 to 2012. The database is comprised of static and dynamic data ranging from birth dates to demographics to laboratory test results to diagnoses to imaging reports to clinical notes. For our purposes, we will be analyzing clinical notes and predicting associated diagnoses. Thus, we will particularly make use of the following tables:

- diagnoses_icd - contains patient diagnoses
- noteevents - contains the following types of clinical notes:
  - Discharge summary - A summary written by the physician (or possibly a team of physicians) at the time of discharge from the ICU.
  - Radiology reports - Reports from imaging procedures such as MRI, CT, and Ultrasounds.
  - Nursing progress reports - Daily notes from the nursing staff.

III. ARCHITECTURE

A. Technology Stack

Our application is based on the traditional server-client model, with a client hosting a front-end application for users to interact with, and the server to enable backend pre-processing and computations on the dataset. The server consists of many different technologies. The main services we use are Python 2.7 and Apache Spark. With Python, we use several libraries and frameworks:

- TensorFlow - A deep learning framework that enables us to build scalable neural networks
- Keras - A high-level wrapper for TensorFlow which allows us to create deep learning models in a more transparent manner
- Numpy - A library that allows us to perform linear algebra in a vectorized and hence less expensive fashion.
- Scipy - A scientific library that provides us sparse data structures.
- Pandas - A library that allows us to store data in an efficient dataframe structure

- CliNER - A natural language processing library similar to CTakes that allows feature extraction from clinical notes

Deep Learning is computationally expensive and thus we will make use of Microsoft Azure's GPU enabled machines to speed up our development time.

Our web application will be built with HTML, CSS, Javascript and JQuery. We use the Flask microframework and JInja2 to bridge the frontend and backend together. Both services will be hosted separately on Heroku. The user will interact with the client application, which will communicate with the backend service to obtain the results of the user's query and provide them on the frontend for visual display.

B. Preprocessing

Our MIMIC III clinical notes are originally very messy and hardly machine readable. There are mis-formatted and irrelevant dates and non alphanumeric characters like asterisks, dashes, slashes, and special characters such as newline characters. Additionally, the notes contain uneven whitespace. A synthetic clinical note, not directly from the MIMIC database, looks as follows:

> Discharge Instructions: General Instructions ?????? Have a friend/family member check your incision daily for signs of infection.
> ?????? Take your pain medicine as prescribed.
> ?????? Exercise should be limited to walking; no lifting, straining, or excessive bending.
> ?????? You may wash your hair only after sutures and/or staples have been removed.
> ?????? Unless directed by your doctor, do not take any anti-inflammatory medicines such as Motrin, Aspirin, Advil, and Ibuprofen etc.
> ?????? Clearance to drive and return to work will be addressed at your post-operative office visit.
> CALL YOUR SURGEON IMMEDIATELY IF YOU EXPERIENCE ANY OF THE FOLLOWING
> ?????? New onset of tremors or seizures.
> ?????? Any confusion or change in mental status.
> ?????? Any numbness, tingling, weakness in your extremities. Followup Instructions:
> You will need to see the nurse practitioner 14 days post-operatively for suture removal. Please call [**Telephone/Fax (1) 1669**]
> for the appointment.
> You will need to follow up with Dr. [**Last Name (STitle) **] in 4 weeks with a Head CT of the brain.
> Completed by:[**2118-12-9**]"
> 184,28063,121936,2125-02-16,,,"Discharge summary","Report",,,"Admission Date: [**2125-2-9**] Discharge Date: [**2125-2-16**]

Fig. 1. Original Clinical Note Input Sample

Our first step in preprocessing this data is to remove the dates. These dates do not contribute as clinical terms so they are not needed. Next, we remove the non alphanumeric characters, special codes, and extraneous white space. Lastly, we lowercase all characters. Our cleaned text sample now looks as follows:

> discharge instructions general instructions friend family member check incision daily signs infection. take pain medicine prescribed. exercise limited walking lifting straining excessive bending. may wash hair sutures staples removed. unless directed doctor take anti inflammatory medicines motrin aspirin advil ibuprofen etc. clearance drive return work addressed post operative office visit. call surgeon immediately experience following new onset tremors seizures. confusion change mental status. numbness tingling weakness extremities. followup instructions need see nurse practitioner 14 days post operatively suture removal. please call telephone fax 1 1669 appointment. need follow dr. last name stitle 4 weeks head ct brain. completed

Fig. 2. Cleaned Clinical Note Sample

We see that the above cleaned sample is much more machine readable and is better suited to construct features from.

C. Feature Construction

CliNER, an open-source natural language processing system for named entity recognition in clinical text of electronic health records, offers an out-of-the box silver model trained on MIMIC data. This model identifies clinical concepts from text data. We can pass each clinical note into this model to extract clinical concepts and phrases, which allows us to build a repository of the medical terminology present in our clinical notes data. After our medical repository is built, we need to clean it. We remove stop words from all clinical phrases and split all phrases on whitespace to give us individual clinical words. We count the number of occurrences for each word and select the top 40,000 words to represent our vocabulary. Below we find an example of the top 15 words in our vocabulary along with their respective counts.

Vocabulary Word	Number of Occurrences
pt	2987186
left	2179431
name	2044487
right	1997474
mg	1941351
ml	1891773
patient	1686424
last	1556429
chest	1287864
plan	1261318
normal	1258367
reason	1198831
clip	1156675
pain	1136394

TABLE I

TOP 15 WORDS FROM VOCABULARY

The table below contains the bottom 15 words from our vocabulary with their respective counts:

Vocabulary Word	Number of Occurrences
codac	1
holdover	1
highlighter	1
dermatography	1
amish	1
surgicals	1
fiexible	1
protoptic	1
nucleoplasm	1
carpial	1
racer	1
dynamical	1
fajitas	1
particiation	1

TABLE II
BOTTOM 15 WORDS FROM VOCABULARY

We see that our top 15 words in our vocabulary are fairly common clinical terms, areas of the body, and conditions. The bottom 15 words seem to be misspellings and non common words that one would not find in a clinical note.

With a present vocabulary, we can create tf-idf features for our set of clinical notes. Tf-idf allows us to reflect how important a word is to a document in a corpus. Some words are very frequent like "the", "a", "is", but carry very little meaningful information about the actual content of the document. If we were to look at a frequency count in a document that has many of these non-meaningful words our model might perform worse as it is not giving importance to the rarer yet more interesting terms. We can re-weight the count features and use this as our feature vector for input into our deep learning models.

D. Long Short Term Memory & Conditional Random Fields

Long short term memory networks are a special type of recurrent neural network capable of learning long-term dependencies. Recurrent neural networks have loops which allow the network to use information from the previous passes, acting as finite memory. Here the older the information, the less usable it is. An LSTM is able to actively maintain self-connecting loops with memory output to prevent memory degrading. The LSTM is trained to select what gets passed into this memory output. Now, the networks does not have to worry about new information degrading the prior information.

Using Tensorflow and Keras we create a sequential model with an LSTM layer consisting of 128 neurons. Our tf-idf feature vector will be passed into this LSTM layer and connected to a dense layer that will output an icd-9 diagnosis code.

Conditional Random Fields (CRFs) are a statistical sequential modeling method frequently used as an alternative to Hidden Markov Models (HMMs) in fields like bioinformatics because they can model a much richer set of label distributions, define a much larger set of features, and have arbitrary weights. In a CRF each feature function is a function that takes in:

- A sentence, S
- The position i of a word in S
- The label l_i of the current word
- The label l_{i-1} of the previous word

The CRF outputs a real valued number. Each feature function f_i is assigned a weight λ_i. Given a sentence S we can score a labeling l of S by summing the weighted features over all words in S. Our score can be computed as follows:

$$score\,(l|s) = \sum_{j=1}^{m}\sum_{i=1}^{n} \lambda_j f_j(S, i, l_i, l_{i-1})$$

We can then take the scores and transform them into probabilities $p(l|s)$:

$$p(l|s) = \frac{exp\,[score(l|s)]}{\sum_{l'} exp\,[score(l'|s)]}$$

$$= \frac{exp\left[\sum_{j=1}^{m}\sum_{i=1}^{n} \lambda_j f_j(S, i, l_i, l_{i-1})\right]}{\sum_{l'} exp\left[\sum_{j=1}^{m}\sum_{i=1}^{n} \lambda_j f_j(S, i, l'_i, l'_{i-1})\right]}$$

The CliNER module has a pre-trained CRF model using MIMIC data. We make use of this pre-trained CRF model to pass in a clinical note and identify the key clinical problems and treatment categories of terms within this note. For instance, we can pass the following clinical note into our conditional random field model:

> Patient has been suffering from headaches and a sore throat and has been prescribed advil.

Fig. 3. Sample Sentence for CRF

and obtain the following patient problems and treatments labeling:

> Patient has been suffering from <problem> headaches <problem> and <problem> a sore throat <problem> and has been prescribed <treatment> advil <treatment>.

Fig. 4. CRF Clinical Sample Sentence Labeling

We see that the model has been able to label headaches and a sore throat as a problem and has labeled advil as the treatment.

E. Long Short Term Memory & Attention Mechanism

LSTMs can be augmented with another feature called an attention mechanism. Human brains are capable of deducing a large amount of information from sensory signal inputs. We are able to filter out irrelevant information from the useful information we are processing. Neural networks can capture this sequence processing task through an attention mechanism. This mechanism holds onto all states from the encoder and gives the decoder a weighted average of the

encoder states for each element in the decoder sequence. The decoder can peek into the encoder sequence to now figure out which element should be outputted next. Specifically, for each encoded input from the LSTM, the attention mechanism calculates the importance:

$$importance_{i,j} = V * tanh\left(E_i W_1 + D_j W_2\right)$$

of each input vector where E_i denotes the encoded input vector at step i, D_j denotes the decoding step j, and W_1, W_2, and V are learned parameters. After the importance is calculated for each encoded vector, the vectors are normalized with a softmax and the weight of each encoded vector. This results in a time dependent encoding, which is inputted to each step of our decoder LSTM. This attention mechanism, just like our conditional random field implementation, will show us what features contributed most to the predicted diagnosis code our LSTM will output. We utilize tensorflow to build an attention mechanism for our LSTM.

F. User Interaction

Once our deep learning algorithm has been finely tuned, it is ready for predictions. To make our system user friendly, we are creating a python-backed web application where doctors and patients can input their clinical record and view the model's predicted diagnoses along with the words most responsible for the diagnosis. In our system, they will find information regarding the top 3 predicted diagnoses and sample clinical reports for each predicted diagnosis. This will allow them to evaluate and gain insight into the condition. We host the system on the Heroku platform at the following link: https://doctorann.herokuapp.com.

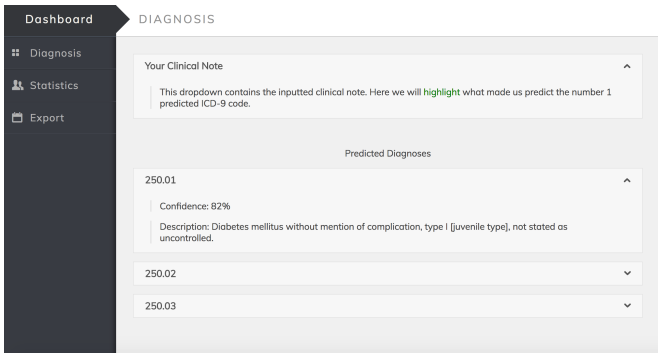

Fig. 5. Dr. ANN Dashboard

G. Visual Pipeline

The entire pipeline of our system is as follows:

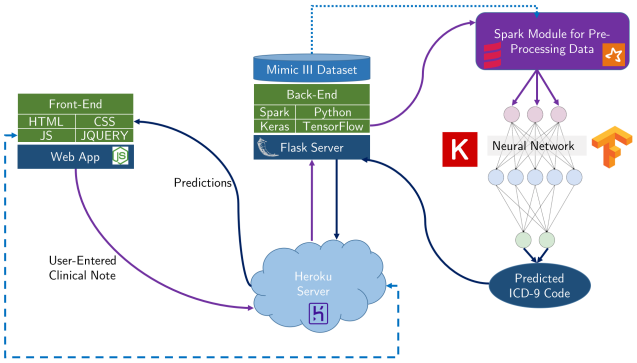

Fig. 6. Dr. ANN system pipeline

IV. Performance Measurements

There are several types of sample-based metrics that we can use in multi-label classification to measure performance. To define our metrics, we will let Y_i denote the set of predicted labels, Z_i denote the ground truth labels, and n to denote the number of samples. We will use the following metrics with their respective definitions:

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|}$$

$$Precision = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Z_i|}$$

$$Recall = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i|}$$

$$F1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2|Y_i \cap Z_i|}{|Y_i| + |Z_i|}$$

We utilize spark to calculate the performance of our respective models.

V. Results

[Results will be included in the final paper draft]

VI. Conclusions

[Conclusions will be included in the final paper draft]

References

[1] Weston J, Chopra S, Bordes A. Memory networks. arXiv preprint arXiv:1410.3916. 2014 Oct 15.

[2] Fries JA. Brundlefly at SemEval-2016 Task 12: Recurrent neural networks vs. joint inference for clinical temporal information extraction. arXiv preprint arXiv:1606.01433. 2016 Jun 4.

[3] Nigam P. Applying Deep Learning to ICD-9 Multi-label Classification from Medical Records.

[4] Miotto R, Li L, Kidd BA, Dudley JT. Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. Scientific reports. 2016;6.

[5] Kim Y, Denton C, Hoang L, Rush AM. Structured Attention Networks. arXiv preprint arXiv:1702.00887. 2017 Feb 3.

[6] Perotte A, Pivovarov R, Natarajan K, Weiskopf N, Wood F, Elhadad N. Diagnosis code assignment: models and evaluation metrics. Journal of the American Medical Informatics Association. 2014 Mar 1;21(2):231-7.

[7] Subotin M, Davis AR. A system for predicting ICD-10-PCS codes from electronic health records. InProc BioNLP 2014 Jun 27 (pp. 59-67).

[8] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473. 2014 Sep 1.

[9] Choi E, Bahadori MT, Song L, Stewart WF, Sun J. GRAM: Graph-based Attention Model for Healthcare Representation Learning. arXiv preprint arXiv:1611.07012. 2016 Nov 21.

[10] Choi E, Bahadori MT, Sun J, Kulas J, Schuetz A, Stewart W. RETAIN: An Interpretable Predictive Model for Healthcare using Reverse Time Attention Mechanism. InAdvances in Neural Information Processing Systems 2016 (pp. 3504-3512).