Improving Readmission Prediction by Extracting Relevant Information from Clinical Notes

Stan Arefjev, Luis Fernandez-Rocha, Justin Skycak, Devan Stormont
Georgia Tech, Atlanta, GA

Abstract *The purpose of this project is to develop a model that will predict patient readmission from the MIMIC-III database using coded features and clinical notes. The goal is to predict unplanned patient readmission with a higher ROC AUC score than prior published work [RAJKOMAR][1]. The approach taken to accomplish this is three fold. First we build a data pipeline to find all readmission events for labeling, as well as processing the coded features from the dataset. Next, the clinical notes are processed using Natural Language Processing (NLP) in order to build additional predictive features. Finally, a Recurrent Neural Network (RNN) will be used to train and test the combined processed data in order to make final predictions and calculate the ROC AUC score.*

1 Introduction and Background

Patient readmission prediction involves predicting the probability that a discharged patient will be readmitted in the near future, with the goal of guiding discharge decisions to prevent avoidable readmissions. Erroneous discharges not only increase risk for patients, but also incur increasing financial cost for hospitals due to readmission penalty fines [BOCCUTI][6].

Conventional readmission prediction models compute unplanned readmission probabilities based on coded fields in patient medical records, such as admissions dates, diagnoses, lab tests, and medications. These fields are already structured, even in their raw form, which makes them easy to featurize, and they contain some degree of information relevant to readmission prediction ([RAJKOMAR][1] 2018 achieved ROC AUC of 0.77 using only coded fields).

However, coded fields contain only a small portion of the total information about a patient: even more personalized, detailed information about patients is contained in clinical notes written by doctors. Despite containing vast amounts of rich information, unfortunately, clinical notes are often left out of readmission prediction models because they are difficult to featurize. Unlike coded fields, clinical notes have no predefined structure, and exist as free-text narratives rather than clear-cut (field, value) pairs.

Several groups have used NLP techniques [KANG][3] to featurize clinical notes and incorporate them into medical prediction models, successfully improving predictive performance ([FORD][2], [CURTO][7]). In this project, we aim to provide another example of leveraging clinical notes to improve predictive performance, in the case of unplanned readmission prediction. Specifically, we reproduce the same baseline results (ROC AUC of 0.77) achieved in [RAJKOMAR][1] by training solely on coded features, and then attempt to improve the predictive model by supplementing it with features derived from clinical notes using NLP techniques.

Unfortunately, [RAJKOMAR][1] does not give much guidance about how they structured their model. This inherently leads to difficulty in reproduction. They do briefly mention the use of a deep learning model. They also describe the size of the test and training sets. However, they do not describe their training methodology: Did they use training, validation, and a heldout test set, or did their test set act as a validation set? Did they use cross-validation? What did their deep learning model look like? How much training did they put their model through? Many questions remain unanswered.

2 Problem Formulation

The patient readmission prediction problem was formulated as follows: given a patient who has been discharged from a hospital, what is the probability that the patient will have an unplanned readmission to a hospital within a 30-day window after being discharged? As such, the target variable for a patients discharge was chosen as 1 if the patient was readmitted to a hospital within a 30-day window after being discharged, and 0 otherwise ([CRONIN][9], [RAFI][10]). As a case study, we applied our readmission prediction pipeline to patient data obtained from MIMIC-III, a publicly available dataset which contains both coded readmission data as well as uncoded physician notes related to readmission ([JOHNSON][4]). We considered readmissions only for unplanned events, filtering out newborn and

death events. Newborn events introduce significant variance and often contain incomplete clinical notes. Meanwhile, mortality events are invalid for readmission events, due to the death of the patient. Also, elective admissions were removed, as these do not typically qualify as ”unplanned” readmissions.

3 Approach and Implementation

3.1 Infrastructure

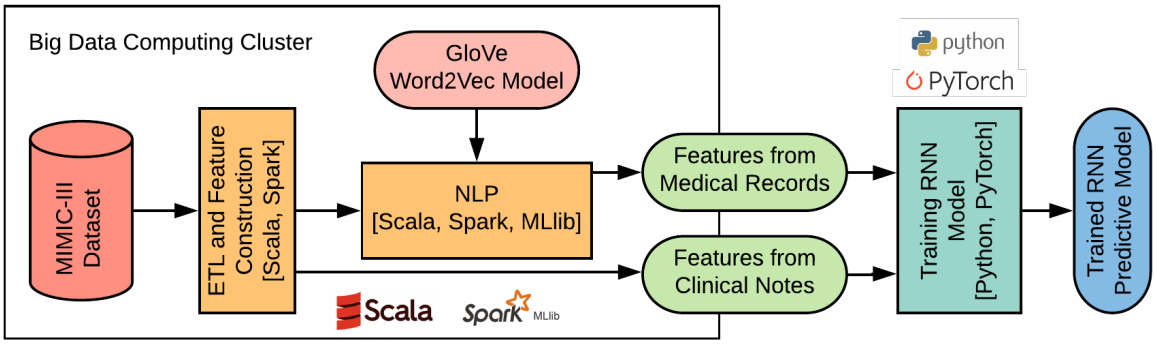


Figure 1: System architecture.

Typical medical records are very massive datasets, which require the use of big data tools for processing, cleaning, and extracting features in a timely manner. A Docker environment was used to leverage Scala with Apache Spark and MLlib to load and process the data from clinical records Figure 1. To process clinical notes, Spark with MLlib was used to tokenize, filter and vectorize notes with GloVe, a pretrained embedding model. This Docker environment and associated data pipelines can be used with a single host (if provided enough resources), or distributed across multiple hosts to process the data. The resulting featurized dataset was stored in CSV files, and Python and PyTorch were used to build and train the model on the featurized dataset. This featurized dataset is small enough for a single host to be able to perform model training, allowing non-distributed machine learning methods to be used.

For our experiment, the Docker environment was run on a host system for data processing and model training. The host system consists of a 4-core hyperthreaded host with 16 GB of RAM and GPU acceleration. The coded features from MIMIC-III tables can be processed in roughly 30 minutes using the Docker environment on this host system. The NLP clinical notes from MIMIC-III tables can be processed in roughly 30 minutes with the same environment. Model training using the featurized dataset in the same environment can run 100 epochs in roughly 10 minutes.

3.2 Dataset

We used the publicly available MIMIC-III dataset, which contains 55000 admission records for 33000 patients, as well as information about diagnoses, lab results, drugs, and raw clinical notes [JOHNSON][4]. All records within MIMIC-III are indexed by patient and admission ID.

3.3 Data Cleaning and Feature Extraction

Within the MIMIC-III dataset, five CSV tables were used: ADMISSIONS, LABEVENTS, DRGCODES, DIAGNOSES_ICD and NOTEEVENTS. These tables were linked by the field HADM_ID which represented a unique visit. ADMISSIONS and NOTEEVENTS were filtered as follows:

- ADMISSIONS: To enable a prediction window of N days, admissions which were fewer than N days before the last recorded admission in the dataset were removed, since no target variable could be computed for these

admissions. Also, admissions with ADMISSION_TYPE equal to NEWBORN or ELECTIVE were removed, to ensure that only EMERGENCY or URGENT readmissions were counted as true readmissions.

- NOTEEVENTS: Notes which were labeled as being erroneous (according to ISERROR) were removed.

In the ADMISSIONS table, events were sorted and grouped by SUBJECT_ID, so that each SUBJECT_ID was paired with a list of admission events in chronological order. For each admission event, the duration from the patients current admission discharge date to the next admission date was calculated, and the admission event was labeled with a target variable: '1' if the duration to the next unplanned readmission falls within a 30-day window, and '0' otherwise.

For each admission, a structured feature sample of diagnoses, lab events, and drug codes for the particular patient and visit was generated by joining the admission with DIAGNOSES_ICD, LABEVENTS, and DRGCODES on HADM_ID. To reduce the dimensionality of diagnoses, only the part of the ICD9 code before the decimal part was used. Then, the features were normalized using min-max normalization, which rescales each feature into the range from -1.0 to +1.0. As a result, each admission was paired with a single normalized feature vector consisting of 2692 numerical features between 0 and 1.

As expected, the processed dataset was very imbalanced, with only 2.9% of samples representing patients who were readmitted within a 30-day window. This was taken into account when training and analyzing the model by weighting prediction thresholds to reflect this imbalance.

3.4 NLP

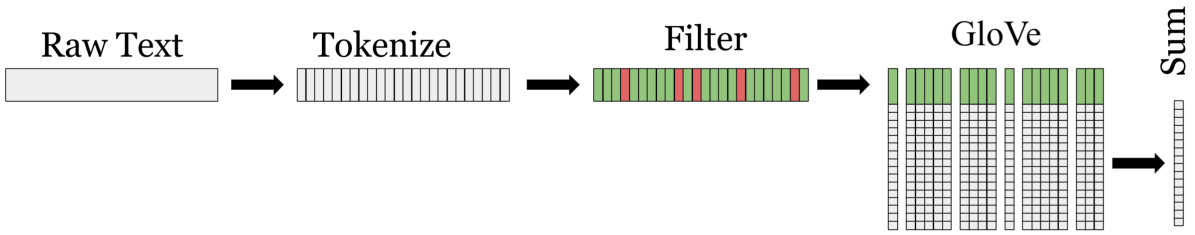


Figure 2: NLP pipeline.

NLP methods were used to extract structured, featurized information from unstructured clinical notes. For each admission, an unstructured feature sample of clinical notes was generated by joining the admission with NOTEEVENTS on HADM_ID. To featurize the notes, the notes were first tokenized (using the MLlib library) into individual words and filtered to exclude stopper words (common, irrelevant words such as the and that, as well as medication measurements). Stopper words were determined through manual analysis of the most frequent tokens. Remaining words were mapped to 300-dimensional feature vectors using a GloVe[11] embedding model, and then summed into a single 300-dimensional feature vector encoding the notes of the particular visit, Figure 2. After conversion NLP features are rescaled into the range from -1.0 to +1.0.

The processed features for diagnoses, lab events, drug codes, and clinical notes were grouped by SUBJECT_ID and, a separate CSV file containing a table with features as columns and admissions as rows was created for each SUBJECT_ID. These CSV files were used as training inputs into a Recurrent Neural Network (RNN).

3.5 Predictive Model

The most consistently significant predictor of readmission is previous hospitalizations [DONISI][5]. Therefore, to make use of temporal dependencies in the admission sequences, a Recurrent Neural Network (RNN) was chosen for predictive modeling ([LIN][8], [RAFI][10]). Not only can RNNs learn patterns in the temporal domain, they can also handle sequences of variable length, making them an ideal match for the natural structure of the admissions data.

Input to neural network consists of a batch of admission sequences. Each admission sequence has a corresponding

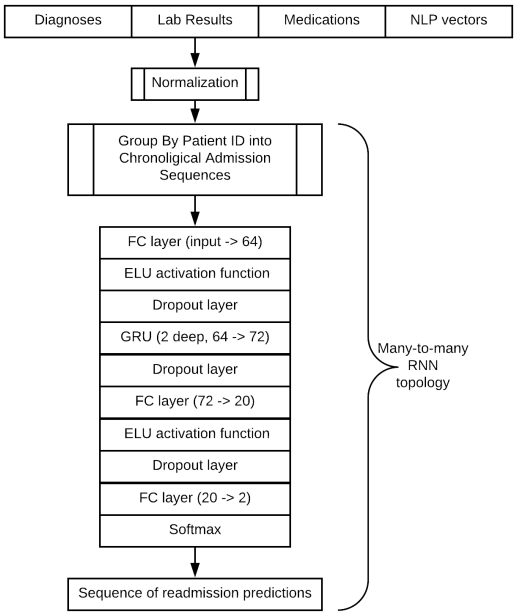


Figure 3: RNN topology.

list of labels. Output is a set of sequences with predictions for each admission. This leads to an RNN implementation known as many-to-many (sequence on input generates a sequence on output). It mapped each sequence of admission samples for a particular patient, to a sequence of prediction labels of that patient being readmitted within a 30-day window after discharge.

RNN was implemented using PyTorch with the structure shown on the Figure 3. The first layer is a fully connected layer with 64 hidden units, followed by an ELU activation layer. The next layers are two Gated Recurrent Unit (GRU) layers, each with 72 hidden units. And 2 final fully connected layers with 20 and 2 output units with ELU activation function on the first one and the last one followed by softmax activation function to produce predictions of readmissions. Dropout of 50% was applied after all fully connected layers except the last and to the GRU layers (the choice of 50% maximizes dropout variance), to act as a feature regularization method. This helps to prevent the model from overfitting on the the exact input values, helps to generalize better to unseen test sets, and to be more robust to noise in input features.

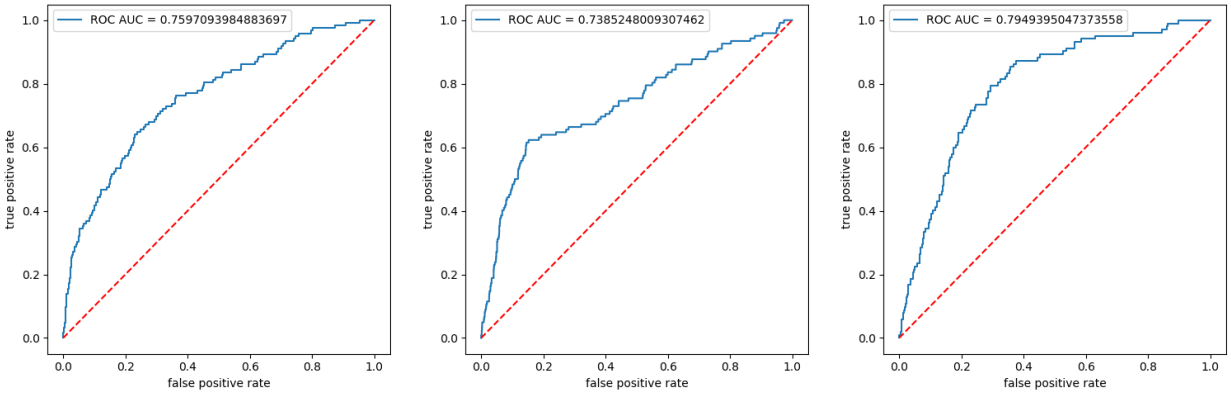
The RNN model was trained using binary cross-entropy loss $l = y_n \log x_n + (1 - y_n) \log(1 - x_n)$ as we are training a model to predict probabilities based on binary labels. To account for the imbalance (2.9% readmission rate) in the dataset, thresholds for readmission probability were adjusted proportional to imbalance ratio.

4 Experimental Evaluation

Model	ROC AUC (validation)	ROC AUC (test)	
Coded features	0.7928	0.8015	
NLP features	0.7791	0.7464	
Combined features	0.8111	0.8001	
Baseline[1]		0.77	

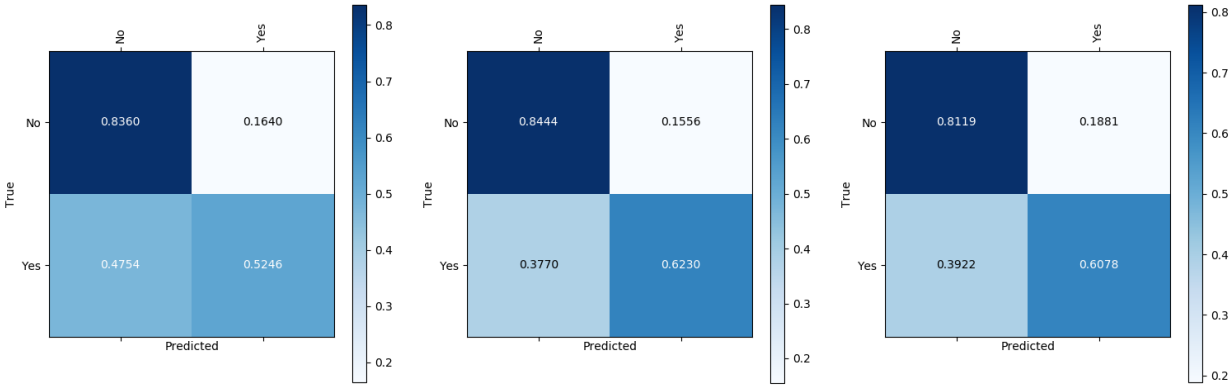
Table 1: ROC AUC score for model trained on different combinations of features (with and without NLP features). The result is an average of 10 runs on test and validation datasets.

The dataset was split into training (90%) and test (10%) sets. The training set was used to train the model and test set



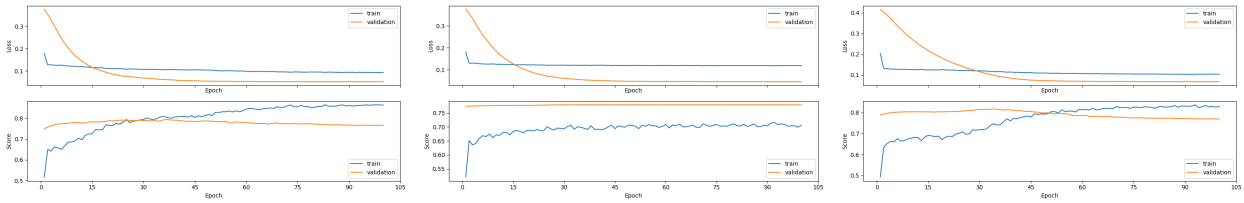
(a) Features form clinical notes **(b)** NLP features from clinical notes **(c)** Combined features

Figure 4: ROC curve for test dataset.



(a) Features form clinical notes **(b)** NLP features from clinical notes **(c)** Combined features

Figure 5: Confusion matrix for test dataset.



(a) Features form clinical notes **(b)** NLP features from clinical notes **(c)** Combined features

Figure 6: Learning curves with loss and ROC AUC score for training and validation datasets.

to measure the performance, which is reported below.

To train the neural network and calculate more statistically significant scores, we used randomized cross validation with 10 folds that split training dataset into training and validation datasets (of a 90/10 split). We trained our model on 100 epochs and picked the model that has the highest ROC AUC score on the validation dataset in order to prevent overfitting. Due to the use of dropout in the model, in some cases the score on the training set did not exceed the score on the validation sets, during which scoring dropout is disabled.

The predictive performance of the model was measured using the ROC AUC score, comparable to other research. Using coded features only, the model achieved ROC AUC of 0.801 (with standard deviation 0.014), Table 1 and Figure 4, which exceeds the benchmark of 0.77 in [RAJKOMAR][1]. This provides a good baseline for comparison with the improvement that could be achieved by augmenting the model with features extracted from clinical notes

using NLP methods. Using only features extracted form clinical notes, the model achieved ROC AUC of 0.746. Finally, combining coded features with clinical notes features, the model achieved ROC AUC of 0.800 (with standard deviation 0.008), remaining at the same quality as that with coded features only. While the ROC AUC scores were roughly the same when including notes, the standard deviation was smaller, indicating a more confident model.

It is a bit surprising that combined features haven't resulted in an improved performance. It could be because, in the current implementation, features form clinical notes weren't normalized while coded features from medical records were normalized, or because features from medical records are very sparse and input from clinical notes overwhelmed the input of the Neural Network.

In the current implementation we use a simple Bag of Words NLP technique to vectorize a sequence of words in notes. This technique is powerful and as we can see in Table 1 that by using features produced from that method we achieved ROC AUC score of 0.746. However, Bag of Words is limited in that it can't capture semantic information that is dependent on positional interaction of words in sentences. Using more sophisticated NLP methods [KANG][3], we could potentially improve the performance further. Further, we used a pretrained Word2Vec model (GloVe) on Wikipedia articles; training an embedding model on clinical documents could produce a slightly better results.

We also tried to use a pretrained gensim Doc2Vec model to generate clinical note embeddings, which is known to better capture document similarities, but we achieved a result very similar to using the Word2Vec model. However, that model also fails to capture semantic information inside the notes.

Our RNN has a relatively simple structure but as we can see form learning curves, Figure 6, the model fits training dataset with very little loss and the gap between training and validation losses are very large that indicates that increasing model complexity would only result in overfitting and won't help with generalization.

5 Conclusion

The goal of this project was to build a readmission prediction model which leverages clinical notes to achieve a higher ROC AUC than prior published work. To accomplish this goal, admission events in the MIMC-III were first labeled with a readmission target variable which indicated whether the patient was readmitted to a hospital within a 30-day window after the current admissions discharge date. Then, patient admissions were featurized by aggregating coded lab events, diagnoses, and drug codes, and using Natural Language Processing techniques (bag of word embeddings) to vectorize clinical notes. Finally, admissions for individual patients were grouped into sequences, which were used to train a Recurrent Neural Network model.

Using only coded features, the model achieved ROC AUC of 0.80, exceeding the benchmark of 0.77 achieved in [RAJKOMAR][1], Table 1. Using only clinical notes, the model achieved 0.746 AUC ROC. Combining coded features and clinical notes, the model also achieved AUC ROC of 0.80, but this did not improve performance.

Using just coded features we achieved better performance to that reported in [RAJKOMAR][1], which we used as our baseline comparison. But adding features extracted form clinical notes using Word2Vec model doesn't produce in very significant improvement in performance. We did find that the inclusion of notes reduced the standard deviation of the trained model.

There are several issues that we think could cause the lack of improvement in the model from the inclusion of clinical notes. It could be the case that our problem domain, the dataset, or both, have an inherently high noise and our model has already reached that maximum performance. It should be noted that our model has a low false positive rate of 18%, Figure 5,which is considered to be a good result from a medical standpoint, as only 18% of patients will be erroneously predicted to need readmission in 30 day window. From a clinical perspective, this means both less stress on the patients for improper readmission, as well as cost savings for the hospital. However, our model could still use more improvement to reduce false negatives (with a 40% rate).

One issue is that training samples are limited. Even though our dataset has 40k samples, the dataset is very imbalanced, only having around 1k samples with unplanned readmission. Increasing the size of the dataset should give a significant improvement in performance.

Most of the performance in our model was achieved by using coded features, but they only contain a fraction of

information that could be found in clinical notes. Though we tried to extract useful information using Word2Vec model but that method isn't powerful enough to extract semantic information that could differentiate minute details in patient conditions.

We tried to experiment with Doc2Vec models, but they performed similar to Word2Vec models. This may be because they're suited to calculating similarity between documents and not extracting minute semantic information. We also experimented with the Apache cTAKES[12] NLP model that is specifically designed for extracting semantic information form clinical notes. However, we weren't unable to conduct experiments using it in our limited timeframe; it requires a lot of manual engineering and medical domain knowledge to extract information for implementation. Therefore, this avenue of exploration was scrapped.

In conclusion, we were able to implement a distributed system to extract features from medical records and trained a RNN model to achieve results comparable to one of the leading papers [RAJKOMAR][1]. Given more time, we believe it's possible to improve performance even further by using larger datasets and more sophisticated NLP methods.

References

1. Rajkomar A, Oren E, et al. Scalable and accurate deep learning for electronic health records. npj Digital Medicine 2018; 1:18
2. Ford E, Carrol JA, Smith HE, Scott D, Cassel JA. Extracting information from the text of electronic medical records to improve case detection: a systematic review. Journal of the American Medical Informatics Association 2016; 23.5:1007-1015
3. Kang T, Zhang S, Tang Y, Hruby GW, Rusanov A, Elhadad N, Weng C. ElilE: An open-source information extraction system for clinical trial eligibility criteria. Journal of the American Medical Informatics Association 2017; 24.6:1062-1071
4. Johnson, A. E. W. et al. MIMIC-III, a freely accessible critical care database. Sci. Data 3, 160035 (2016)
5. Donisi, V., Tedeschi, F., Wahlbeck, K., Haaramo, P., & Amaddeo, F. (2016). Pre-discharge factors predicting readmissions of psychiatric patients: a systematic review of the literature. BMC psychiatry, 16(1), 449.
6. Boccuti, C., & Casillas, G. (2015). Aiming for fewer hospital U-turns: the Medicare hospital readmission reduction program. Policy Brief.
7. Curto, S., Carvalho, J. P., Salgado, C., Vieira, S. M., & Sousa, J. M. (2016, July). Predicting ICU readmissions based on bedside medical text notes. In 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE) (pp. 2144-a). IEEE.
8. Lin, Y. W., Zhou, Y., Faghri, F., Shaw, M. J., & Campbell, R. H. (2018). Analysis and Prediction of Unplanned Intensive Care Unit Readmission using Recurrent Neural Networks with Long Short-Term Memory. bioRxiv, 385518.
9. Cronin, P. R., Greenwald, J. L., Crevensten, G. C., Chueh, H. C., & Zai, A. H. (2014). Development and implementation of a real-time 30-day readmission predictive model. In AMIA Annual Symposium Proceedings (Vol. 2014, p. 424). American Medical Informatics Association.
10. Rafi, P., Pakbin, A., & Pentyala, S. K. Interpretable Deep Learning Framework for Predicting all-cause 30-day ICU Readmissions.
11. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.
12. Savova, Guergana; Masanz, James; Ogren, Philip; Zheng, Jiaping; Sohn, Sunghwan; Kipper-Schuler, Karin and Chute, Christopher. 2010. Mayo Clinic Clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. JAMIA 2010;17:507-513 doi:10.1136/jamia.2009.001560