Applying Deep Learning to ICD-9 Multi-label Classification from Medical Records

Priyanka Nigam
Department of Computer Science
Stanford University
Stanford, CA 94305
pnigam@stanford.edu

Abstract

Medical records contain detailed notes written by medical care providers about a patient's physical and mental health, analysis of lab tests and radiology results, treatment courses, and more. This information may be valuable in improving medical care. In this project, we apply deep learning models to the multi-label classification task of assigning ICD-9 labels from these medical notes. Previous works have applied machine learning methods, like logistic regression and hierarchical SVM, using bag-of-words features to this task. On a dataset of around 40,000 critical care unit patients with 10 labels and with 100 labels, we find that a Recurrent Neural Network (RNN) and a RNN with Long Short-term Memory (LSTM) units show an improvement over the Binary Relevance Logistic Regression model.

1 Introduction

Medical records contain a variety of patient data including demographics, medical history, vital signs, lab test results, prescriptions issued, procedures performed, etc. Medical records also contain detailed notes and reports written by members of the medical staff. The vast array of information in these records may provide medical insights including identification of early symptoms, recognition of patterns in disease progression, and a more detailed understanding of treatment outcomes [1]-[2].

These insights could be used to improve medical care. For example, understanding treatment outcomes in individuals with different demographics, vital signs, lab results, etc. could help in developing more specific and targeted treatments for each patient [2]. This is already done in medicine for large classifications like ethnicity and gender, but analysis of a large set of patient records may allow an even more specific approach. Identification of early symptoms may help flag high-risk patients or those in early-stages of a disease progression [3]. Other applications include identifying causes of post-operative complications [4] and identifying patients for medical studies and clinical trials [1].

2 Problem Statement and Background

This is a broad area of research, but there has been previous work done particularly in the area of predicting diagnosis and clinical phenotypes from medical records. Specifically, most of this previous work has looked at how to automatically assign ICD-9 labels based on patient records [2]-[6]. Therefore in this project, we will focus on the task of automatically assigning ICD-9 labels from medical records, focusing in particular on the notes written by medical staff members.

2.1 ICD-9 Classification System

ICD-9 is a system of about 15000 numerical codes representing diagnoses and procedures. These are used to standardize medical records to aid in statistical analysis (condition frequency, mortality rates, etc.) and in insurance billing. ICD-9 has been superseded by ICD-10, which has around 70,000 codes, but many medical records, and in particularly those available through public datasets, still use the ICD-9 classification system. The system is hierarchical in natural, with categories for larger sets of similar health conditions that encompass labels for more specific classifications that take into account causes, specific locations in the body, etc. [6].

2.2 Multi-label classification

Patients can have multiple conditions, or in other words multiple ICD-9 labels, thus this is a multi-label classification task, a generalization of the multi-class task where each instance no longer has a single label. Multi-label classification has an additional degree difficulty because the number of correct labels for each instance is unknown, and one must avoid simply learning label frequencies [7]. However this task is unique in that ICD-9 labels for different conditions are not independent, for example patients with diabetes are at higher risk for cardiac problems. Furthermore due to its hierarchical structure, some ICD-9 labels are mutually exclusive and some ICD-9 labels encompass others. Also certain diseases are incredibly rare whereas others are very common, so label distribution is highly skewed.

There are a few standard approaches to multi-label classification. The first is Binary Relevance in which a separate binary single-class classifier for each label is trained on the dataset. This method simplifies the problem but also makes the assumption of label independence. The second is the Label Powerset method, which transforms every possible set of labels into a class and then attempts to predict one of these classes using multi-class techniques. This method results in a large set of classes, 2^l where l is the number of labels, and often in unbalanced classes. There are also ranking based methods that attempt to rank more relevant labels higher [7]-[8]. Previous work has also been done in using neural networks for multi-label classification of textual data, with the development of a new neural network algorithm called BP-MLL with a novel loss function [9].

2.3 Previous Work

The previous work done in automatically assigning ICD-9 codes has focused on rule-based systems designed by experts, traditional machine learning methods including SVM and Logistic Regression (LR), as well as more complicated methods like hierarchical SVMs [10]. Most of these systems have used bag-of-words features. Rule based systems in many cases have out-performed other simple machine learning methods [5]. This is not unexpected as rule-based systems resemble how medical professionals diagnose conditions. Furthermore many of these previous techniques have employed simple sets of features that likely fail to capture the degree of complexity that can be found in medical records. As a result, some methods have been shown to work well on specific subsets of labels, but fail to scale to larger or more general sets of labels [10].

3 Dataset and Metrics

3.1 Dataset

In this project, we use the MIMIC III dataset, released December 2015, which contains de-identified medical records from 46250 critical care unit patients of all ages at the Beth Israel Deaconess Medical Center between 2001 and 2012 [11]. Because these medical records are from ICU/CCU patients, more debilitating conditions and patients in more advanced stages of the disease are likely to be overrepresented. The records in this dataset have a variety of information including medications prescribed, test results, procedures performed, etc. Much of this data, like the caregiver information or admission time, is not as relevant and may lead to an overly complex model. We also want to focus on the NLP task of semantic parsing and understanding of the doctor's notes. Therefore we will look at only the text notes and diagnosis (not procedure) ICD-9 labels.

Before training any models, it is important to understand the label distribution. There are an average of 13.9 labels per patient. This dataset contains a subset of 6985 ICD-9 codes, with a highly skewed distribution; the top 105 codes make up 50% of the total labels in the set, 1503 labels have only one example in the dataset, and 3110 labels have fewer than 5 examples in the dataset. A cumulative distribution function of the labels can be found in Figure 1.

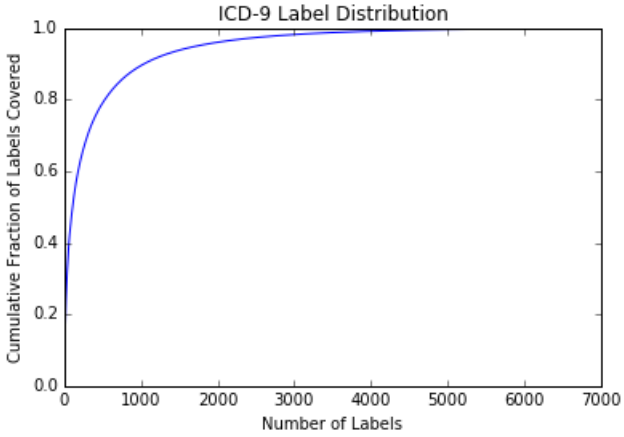

Figure 1: Cumulative Distribution of ICD-9 labels in MIMIC III

After inspecting the information available in the dataset and the distribution of ICD-9 labels, we perform a series of preprocessing steps to generate a filtered dataset. First we identify the top ten labels based on number of patients with that label. We then remove all patients who don't have at least one of these labels, which leaves 31865 patients, and then filter the set of labels for each patient to only include these labels. We create a list of at most 20 notes per patient in the filtered list of patients, ordered by the date that note was written. Finally to measure performance of the models, we split the filtered and pruned dataset into training, development, and test sets with a 50-25-25 split. We use the train set to train our models, the development set to routinely validate that we are not overfitting the train set, and the test set to measure model performance in the end. We repeat this preprocessing procedure to generate a second filtered dataset using the top 100 ICD-9 labels; this set has 44591 patients.

Table 1: Top 10 ICD-9 Labels

ICD-9 Label	**Fraction of Patients with Label**
4019: Hypertension	0.445
4280: Congestive Heart Failure	0.282
42731: Atrial Fibrillation	0.277
41401: Coronary atherosclerosis	0.267
5849: Acute Kidney Failure	0.196
25000: Diabetes, Type II	0.195
2724: Hyperlipidemia	0.187
51881: Acute Respiratory Failure	0.161
5990: Urinary Tract Infection	0.141
53081: Esophageal Reflux	0.136

3.2 Metrics

There are several types of metrics used in multi-label classification: label-based, sample-based, and ranking-based metrics. Label based metrics measure the binary classification performance of that specific label, so accuracy, precision, recall, etc. are measured for each label separately.

Sample-based metrics evaluate metrics on the entire set of labels for each instance. Ranking metrics measure how well the classifier ranks relevant and irrelevant labels prior to binarization.

We will use the following sample based metrics [7]:

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

$$F_1 = \frac{1}{n}\sum_{i=1}^{n}\frac{2|Y_i \cap Z_i|}{|Y_i| + |Z_i|} \qquad Accuracy = \frac{1}{n}\sum_{i=1}^{n}\frac{|Y_i \cap Z_i|}{|Y_i \cup Z_i|}$$

where: Y_i = set of predicted labels
Z_i = set of ground truth labels
n = number of samples

Another valuable metrics is the ranking loss, which measures how often irrelevant labels are ranked higher than relevant labels [7]:

$$RL = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|Y_i||\overline{Y_i}|}\left|\left\{(y_a, y_b) : r_i(y_a) > r_i(y_b), (y_a, y_b) \in Y_i \times \overline{Y_i}\right\}\right|$$

where: r_i = ranking function

4 Technical Approaches and Models

4.1 Feature Selection

The first step in feature selection is vocabulary generation. By looking over some of the notes, we notice that clinical text has various idiosyncrasies, abbreviations, common misspellings, important phrases etc. that may be important to account for (Table 2). Some examples are presented in the table below. For the most part, we ignore these issues when generating the vocabulary. To construct the vocabulary, we tokenize all notes in the filtered training dataset and keep a word-document matrix with the counts of each term in each note. To tokenize, we split on spaces and certain types of punctuation like periods and semicolons and remove stop words and numerical tokens. Then only terms that appear in at least 3 documents are kept in the word-document matrix. The TF-IDF scores are calculated for each term, and the vocabulary is selected as the top 40,000 terms ordered by TF-IDF scores.

Example misspellings		**Example Phrases**
zaroxalyn	hypercholesteremia	central line
zaroxlyn	hypercholesterinemia	subcortical white matter
zaroxolyn	hypercholestermia	ulcerative colitis
zaroxylin	hypercholesteroemia	small cell carcinoma
zaroxyln	hypercholesterolaemia	
zaroxylyn	hypercholesterolinemia	
	hypercholestolemia	
	hypercholestremia	
	hypercholestreolemia	
	hypercholestrolemia	
	hypercholeterolemia	
	hypercholexterolemia	
	hypercholsterolemia	

Table 2: Examples of misspellings and important phrases found in the dataset

After constructing the vocabulary, we construct a bag of words feature vector for each patient note. This vector is a 40,000 dimensional vector, $v^{(j)}$, where $v_i^{(j)}$ is the count of the number of times the i^{th} vocabulary word appears in note j. This results in a sparse vector representation of each note that ignores word order. We chose a bag of words representation because the medical notes don't really have a particular grammatical structure and while the word order within the small segments of text matters, the order of each segment doesn't matter too much.

4.2 Baseline Model: Logistic Regression

As a baseline, we implement a Binary Relevance Logistic Regression model. In this model, we train a separate logistic regression model for each label, and each model independently predicts the value (0 or 1) of that label. As input features to this model, we take the sum of all of the bag-of-words note vectors for each patient and normalize them.

4.3 Feed-Forward Neural Network

We then implement a basic feed-forward neural network, to understand the baseline performance of deep learning models. Once again, we have similar input features: the unnormalized sum of all of the bag-of-words note vectors for each patient. We used ReLU activation functions in the hidden units and a learning rate of 0.001. For the 10 label task, we experimented with the number of hidden layers and the size of these layers, and found that we acheived the best performance on the development set with two hidden layers of size 300 and 100 and no dropout. We used the sigmoid cross entropy as the loss function to optimize. To predict labels, we apply the sigmoid function on the output and predict 1 for all output labels with a value greater than 0.5 and 0 otherwise. For the 100 label task, the model was the same except the two hidden layer sizes were 1000 and 1000.

4.4 Recurrent Neural Network

After establishing baseline models on this dataset, we implement a recurrent neural network (RNN) with a single layer and 20 time steps (Figure 2). The input features to this network are slightly different. Instead of summing all of the bag-of-words note vectors, we keep each vector separate and input a normalized vector at every time step; this is $\hat{x}^{(i)}$ in Figure 2. The vectors are input in order with the oldest record input first and the most recent record input last. We limit the input to the most recent 20 notes; if a patient has fewer than 20 notes, we pad with vectors of all zeros at the beginning. We ignore predictions, $\hat{y}^{(i)}$, at every time step except the last time step. We use the same sigmoid output layer and thresholding to predict labels from the final output, $\hat{y}^{(20)}$.

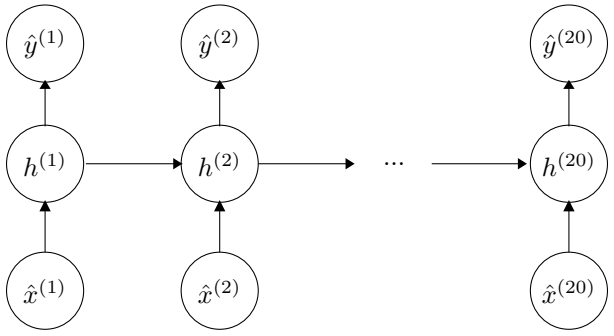

Figure 2: Recurrent neural network with 20 time steps

We use tanh activation units, a dropout rate of 0.1, learning rate of 0.001, and sigmoid cross-entropy for the loss function. Each hidden layer node in Figure 2 corresponds to a hidden layer of size 100 for the 10 label case and 1000 for the 100 label case.

4.5 Recurrent Neural Network with LSTMs and GRUs

We then extended the RNN to use LSTM units as information from previous notes may be very important and we do not want to lose earlier information. We simply replace the nodes labelled $h^{(i)}$ with LSTM units, so it is still a recurrent network with a single hidden layer. Other aspects of the network remain the same; we use a tanh activation function, sigmoid cross-entropy for the loss function, and sigmoid on the output layer to predict positive and negative labels.

Although we expect the performance to be similar with GRUs and LSTMs, out of interest in better understanding the performance differences between the two, we also substituted the LSTM units for GRUs. The rest of the network remains the same.

5 Results

The results on the 10 label task are presented in Table 3. All of the NN models perform better than the baseline, logistic regression, with the RNN with GRUs performing the best. The F1 score on the test set for the feed-forward neural network model is 31.2% higher than that of the baseline, and the F1 score on the test set for the RNN with GRUs model is 40.1% higher than that of the baseline. The baseline, logistic regression, overfits the training set, with an F1 score of 0.8939 on the training set but only 0.3 on the test set. All models train at similar speeds, although the RNNs with LSTMs and GRUs do train slightly slower than the simpler models, but also have better performance.

For the RNN models, we initially found the model simply learned term frequencies and predicted labels based on that. In an attempt to fix this, we added batch normalization that had been missing and dropout. The batch normalization was particularly important to avoid always predicting term frequencies. This is expected as the activation function, tanh, saturates at large values.

Model	**Training Set**			**Test Set**		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Logistic Regression	0.9447	0.8718	0.8939	0.4381	0.2678	0.3000
Basic NN	0.6629	0.3058	0.3925	0.6671	0.3062	0.3937
RNN	0.7932	0.3143	0.4286	0.5406	0.3610	0.4035
LSTM RNN	0.8310	0.3224	0.4469	0.7488	0.3199	0.4168
GRU RNN	0.8500	0.3074	0.4333	0.8505	0.3005	0.4203

Table 3: Model performance of the 10 label task on the training and test sets. Dev set excluded, but results are similar to the test set.

We also looked at the results for the 100 label case, which are presented in Table 4. As expected, performance is much lower, but using neural network models definitely gives an improvement in performance. We did not have as much time to tune parameters for the 100 label task, because the models take much longer to train due to the both the larger dataset and larger output layer. However we see that based on F1 score, the feed-forward neural network model performs 28.9% better than the baseline and the RNN performs 34.4% better than the baseline.

Model	**Training Set**			**Test Set**		
	Precision	**Recall**	**F1**	**Precision**	**Recall**	**F1**
Logistic Regression	0.7734	0.5154	0.5708	0.3322	0.1598	0.1815
Basic NN	0.3539	0.1994	0.2320	0.3528	0.2022	0.234
RNN	0.2807	0.2667	0.2360	0.3410	0.2776	0.2439
GRU RNN (no dropout)	0.3734	0.3125	0.3066	0.1882	0.1956	0.1691
GRU RNN (0.1 dropout)	0.4648	0.3668	0.3725	0.2177	0.2335	0.2001

Table 4: Model performance of the 100 label task on the training and test sets. Dev set excluded, but results are similar to the test set.

Model	Training Ranking Loss	Test Ranking Loss
GRU RNN (no dropout)	0.1911	0.3874
GRU RNN (0.1 dropout)	0.1606	0.3019

Table 5: Ranking loss of the 100 label task on the training and test sets

Interestingly the RNN model with GRUs performs worse than the feed-forward network and the basic RNN model. The model appears to be overfitting the training data, and likely a higher dropout value is required.

6 Conclusions and Future Work

Previous work has explored the application of traditional machine learning methods, including LR, hierarchical SVM, and rule-based methods, to the task of ICD-9 auto-assignment from medical records. But there has not been much work in applying deep learning models to this task. Because the MIMIC III dataset is relatively new, there also has been little work done on this dataset. Therefore, we presented the results of a baseline model as well as several basic neural network models as it is important to understand the performance of simpler architectures before developing more complex and specialized architectures. Neural network models outperformed the baseline model, but there was not as much benefit from using LSTMs to capture long-term dependencies as we had hoped. Furthermore, none of the models worked well enough to actually be used for this task.

Due to time constraints, there are several extensions left to explore. Certainly, we could look at utilizing more complicated networks, like deep recurrent networks with LSTMs or GRUs to better capture the information embedded in the vector representations. A recursive neural network may better represent the fact that each of the smaller components of clinical notes combine to form a diagnosis, and these components are not necessarily dependent on the previous context. But more improvement is likely to be gained by improving the features. Medical vocabulary is unique in that often phrases, like "arterial line" or "right medial frontal lobe", better capture information than individual words. Furthermore, there are many abbreviations and shorthand. Developing a better vocabulary, that includes these facets, perhaps by using medical lexicons, like the Specialist Lexicon, could aid in developing a better bag-of-words representation. Alternatively, we could use dense embeddings, by using word2vec to train word vectors on medical notes, which should capture meanings of abbreviations. We trained the embeddings, but could not try this approach, because notes are very long and thus feeding in each word embedding in order had too many time-steps. But a more advanced model could be developed; notes have many fields, so each field could be represented as the average of word embeddings. The field features could be fed into each time-step of the RNN.

We choose to tackle this task in particular, because there were clear metrics that did not require expert analysis to understand the results. However the task of diagnosising common diseases is something that doctors excel at already, and we believe that deep learning can be applied to more interesting areas of medical record understanding. These applications include: (1) large scale analysis of treatment outcomes in individuals with different demographics, vital signs, lab results, etc. to develop targeted treatments, (2) identification of patients at higher risk for certain diseases and disorders, (3) identification of patients for clinical trials, and (4) a greater understanding of genotype-phenotype correlations as genomic data become more readily available.

References

[1] Jensen, P.B. & Jensen, L.J. & Brunak S. (2012) Mining electronic health records: towards better research applications and clinical care. *Nature Reviews Genetics* **13**(6):395-405.

[2] Chen, P. (2010) Semantic Analysis of Free Text and its Application on Automatically Assigning ICD-9-CM Codes to Patient Records. *Proc. 9th IEEE Int. Conf. on Cognitive Informatics*, 68-74.

[3] Stubbs, A. & Uzuner, O. (2015) Annotating risk factors for heart disease in clinical narratives for diabetic patients. *Journal of Biomedical Informatics* **58**:S78-91.

[4] Murff H.J. et Al. (2011) Automated identification of postoperative complications within an electronic medical record using natural language processing. *JAMA* **306**(8):848-855.

[5] Goldstein, I. & Arzumtsyan A. & Uzuner, O. (2007) Three Approaches to Automatic Assignment of ICD-9-CM Codes to Radiology Reports. *AMIA Annu Symp Proc*, 279-283.

[6] National Center for Health Statistics, International Classification of Diseases, ninth revision, Clinical Modification (ICD-9-CM) cdc.gov/nchs/about/otheract/icd9/abticd9.htm

[7] Zhang, M. & Zhi-Hua Z. (2014) A review on multi-label learning algorithms. *Knowledge and Data Engineering, IEEE Transactions* **26.8**: 1819-1837.

[8] Nam J. & Kim J. & Mencia E. & Gurevych I. & Furnkranz J. (2014) Large-scale Multi-label Text Classification Revisiting Neural Networks. *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 437-452.

[9] Zhang, M. & Zhi-Hua Z. (2006) Multilabel neural networks with applications to functional genomics and text categorization. *Knowledge and Data Engineering, IEEE Transactions* **18.10**: 1338-1351.

[10] Perotte A. & Pivovarov R. & Natarajan K. & Weiskopf N. & Wood F. & Elhadad N. (2014) Diagnosis code assignment: models and evaluation metrics. *J Am Med Inform Assoc.* **21**(2): 231-237.

[11] Johnson A.E.W. & Pollard T.J. & Shen L. & Lehman L. & Feng M. & Ghassemi M. & Moody B. & Szolovits P. & Celi L.A. & Mark R.G. (2016) MIMIC-III, a freely accessible critical care database. *Scientific Data*. DOI: 10.1038/sdata.2016.35.