2020

Pseudo-data Generation For Improving Clinical Named Entity Recognition

Jeffrey T. Smith
Virginia Commonwealth University

THESIS PSEUDO-DATA GENERATION FOR IMPROVING CLINICAL NAMED ENTITY RECOGNITION

A Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

by

JEFFREY SMITH

Bachelor of Science, Biochemistry - September 2000 to May 2004

Director: Thesis Dr. Bridget McInnes,

Assistant Professor, Department of Computer Science

Virginia Commonwealth University

Richmond, Virginia

July, 2020

Acknowledgements

I would like to thank Jorge Vagras for the use of his LSTM cell image. I would like to thank Bill Cramer and Evan French for their help in the initial part of the project. I would like to thank Dr. Arodz and Dr. Özlem for taking the time to be on my committee and participating in my defense. I would like to thank my wife for dealing with me both working full time and working on my masters full time. And I would like to thank Dr. McInnes for guiding me along the last four years and shaping this thesis into what it is.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Abstract

THESIS PSEUDO-DATA GENERATION FOR IMPROVING CLINICAL NAMED ENTITY RECOGNITION

By Jeffrey Smith

A Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at Virginia Commonwealth University.

Virginia Commonwealth University, 2020.

Director: Thesis Dr. Bridget McInnes,
Assistant Professor, Department of Computer Science

One of the primary challenges for clinical Named Entity Recognition (NER) is the availability of annotated training data. Technical and legal hurdles prevent the creation and release of corpora related to electronic health records (EHRs). In this work, we look at the imapct of pseudo-data generation on clinical NER using gazetteering and thresholding utilizing a neural network model. We report that gazetteers can result in the inclusion of proper terms with the exclusion of determiners and pronouns in preceding and middle positions. Gazetteers that had higher numbers of terms inclusive to the original dataset had a higher impact. We also report that thresholding results in clear trend lines across the thresholds with some values oscillating around a fixed point at the most confidence points.

CHAPTER 1

INTRODUCTION

Named Entity Recognition (NER) is a subtask within the Information Extraction (IE) a branch of Natural Language Processing (NLP). The goal of NER is to identify and extract entities fitting a predetermined category from text. NER typically sits in the middle of NLP pipelines as the information extracted is sent to downstream tasks such as question answering[1], word sense disambiguation[2], and automatic text summarization[3].

Classical NER sought to identify entities from categories such as person, place, and location. Many early datasets utilized news sources as their primary corpora[4][5][6]. Traditional machine learning techniques for identifying sequences, such as conditional random fields (CRF) and hidden markov models (HMM) were used as classifiers. Modern NER now compromises of many domains, are domain specific, and utilize neural network architectures with contextual word or character embeddings to label sequences of entities[7][8]. In this thesis, we examine clinical NER focusing on the extraction of medical entities from clinical records.

In the medical domain, NER systems identify and label entities ranging from diseases and gene sequences to pharmaceuticals and medical equipment. Documents used are often referred to as electronic health records (EHR) or electronic medical records (EMR). Proper identification of these entities is critical for downstream tasks that rely on them such as developing a clinical knowledge extraction system[9]. Unfortunately, NER models have shown that they do not generalize well and require domain specific adaptation[10].

In 2001, Poibeau and Kosseim[11] were one of the first groups to document the issues of performing NER across different domains. Changes in the structure of text, the vocabulary used, and the types of entities being extracted require differing approaches which do not easily carry over between one another. Clinical NER is no exception to this. EHRs and EMRs tend to be heavily unstructured, absent of complete sentences, and make heavy use of domain specific abbreviations and jargon. Each domain also requires annotated datasets built from related corpora.

One of the primary challenges in any supervised machine learning task is training data. Large amounts of annotated training data are required to be efficient and generalize the problem space well. However, these datasets have to be annotated by hand and require significant man-hours to assemble and normalize between annotators[12]. The clinical domain exposes additional challenges due to legal and ethical barriers. Patient confidentiality requires that records be de-identified. There are risks that data can be re-identified with modern machine learning techniques[13]. Care must be taken such that the risk of re-identification is minimized[14].

A potential method of bypassing some of the challenges of generating datasets is the use of pseudo-data, also known as artificial training data. Pseudo-data has been used historically to generate additional samples in cases where there is a large class imbalance. There are two main types of pseudo-data, synthetic and sampled. Synthetic pseudo-data is data that has been produced through statistical or rule based methods that attempts to mimic samples already available in the dataset. A popular algorithm for generating synthetic data is SMOTE, or synthetic minority over-sampling technique[15], which uses a K-nearest neighbors algorithm to create new samples between existing ones. Sampled pseudo-data is data that is extracted from corpora or datasets and is labeled through semi-supervised[16] or unsupervised[17] methods.

Bootstrapping as a technique for machine learning (ML) has been around since the foundations of early machine learning models. One of the first major references to the concept was detailed in 1966, in the paper ”Learning without a teacher.” [18] where the concept of thresholding with statistical inferences from previous data is detailed. Another 1966 paper[19] also detailed similar processes and is one of the first papers to use the term bootstrapping in a machine learning setting. Use of the technique was examined through the 1960s and into the 70s [20][21][22][23] and was found to be useful for improving algorithms in low-information settings.

Use of the technique in NER can be found as far back as 2001[24], and is typically used to add additional training data by selecting unlabeled entities that the initial model is highly confident of. Vlachos and Gasperin[25] were able to demonstrate that bootstrapping training data for biomedical entities was able to outperform manually annotated data in their model.

In this work, we look at the impact of pseudo-data generation on clinical NER using gazetteering and thresholding utilizing a neural network model. We examine precision, recall, and F_1 score on a variety of gazetteers and threshold confidence values using the MIMIC-III Clinical Care Database as a pseudo-data source. The contributions of this thesis are:

1. Examining the effect of gazetteering for clinical NER.

2. Examining the effect of thresholding for clinical NER.

3. Examining the trade-off between precision and recall when either of these techniques are used.

CHAPTER 2

LITERATURE REVIEW

In this section, we describe previous research in NER. We subdivide the previous work into three categories: 1) Dictionary Methods, 2) Rule-Based Methods, 3) Supervised Learning Methods. The remainder of this section describes each of these categories in more detail.

2.1 Dictionary Methods

In NER, dictionary methods take a set of known terms and map them to entities. While precision is often high, the recall in these systems is known to be low[26]. To help solve this issue, partial matching techniques are employed to increase the number of matches to the dictionary[26]. As the number of domains increase, dictionary methods become less feasible to utilize as dictionaries for these domains are often manually compiled. In the modern setting, dictionary methods are sometimes used in tandem with supervised learning methods within the clinical domain [27] but rarely on their own.

2.2 Rule-Based Methods

Rule-based methods take hand-crafted features based on known knowledge of the domain and corpus structure, and use statistical models to predict labels. These systems work on smaller data sets but have trouble when encountering larger or diverse sets[28]. As an example, work on identifying proteins in a small corpus carried over to MEDLINE had a significant drop in precision[29]. Unlike dictionary methods, rule-based methods still have opportunities to produce good results in the modern setting[30].

2.3 Supervised Learning Methods

Supervised learning methods involve systems that take in data and generate mathematical models that typically apply classification or regression. For NER, many use weights and functions to shape input feature data into output label predictions. Four general methods have seen the most use in NER prior to deep learning methods and can be split into probabilistic and non-probabilistic categories. The probabilistic methods are hidden markov models (HMM) and conditional random fields (CRF). The non-probabilistic methods are support vector machines, basic feed-forward neural networks (NN) and deep neural networks (DNNs). Currently a majority of NER models utilize deep learning methods.

Hidden Markov Models (HMM). HMMs are probabilistic models that take advantage of Markov chains. They utilize past state information to generate a directed graph of predictions based on probabilities gleaned from data. In the most simple case, a HMM predicts the label for x given y, where y is all of the previous information. The moniker hidden comes from the assumption that not every possible state is known by the system and thus some are hidden. Due to the nature of sparsity in the NLP domain, the chain rule and naive assumptions are used to simplify the equation such that only the previous state is needed to predict the current point of the sequence[31]. HMMs may run into issues for NER tasks, however, as a HMM seeks to maximize the observation sequence and not the state sequence[32].

Conditional Random Fields (CRF). CRFs are a developed evolution of HMMs that utilize an undirected graph structure. Unlike HMMs which are considered generative models, CRFs are considered discriminative. That is, HMMs model a joint probability distribution and CRFs model a conditional probability distribution. In the NLP domain, linear chain CRFs are normally used which result in conditionally trained finite state machines[33]. This is in a sequence chain where each entity uses probabilities from the previous, next, and current entity to determine classification. CRFs were one of the stronger options for NER[34].

Support Vector Machines (SVM). In the context of NLP, SVMs are non-probabilistic binary classifiers that attempt to define a hyperplane through a given feature space such that the distinction between entities is maximized. Traditional SVMs were linear classifiers. Over time, SVMs were adapted to support non-linear classification through kernel functions that map a given vector space to a higher dimensional space. Although SVMs work great for classification and regression tasks, they are no longer frequently used in modern NLP tasks. As the feature size of many NLP tasks has exploded, so has the time required to train SVMs. This is due to the SVM model having to calculate the Lagrangian dual of the feature space for proper optimization[35].

Feed-forward Neural Networks (NN). Feed-forward NNs are, in their simplest form, a series of layers processed sequentially to give an output value. Each layer contains a weight matrix which defines how inputs are prioritized and an activation function to shape the output. Layers between the input and output are called hidden layers. The use of different activation functions allows NNs to approximate solutions to both linear and non-linear problems[36][37]. Typically NNs are trained through the use of backpropagation on the network which assigns loss to individual weights and adjusts them based on a user defined learning rate, with recent advances being able to represent both words and characters in an n-dimensional feature vector of floating point values[38].

Deep Neural Networks (DNN). Within the last decade, multi-layer neural networks, referred to as deep neural networks (DNNs) have been commonly used for NER tasks. DNNs utilizing recurrent nodes in networks, also called Recurrent Neural Networks (RNNs), have been used for sequential classification problems[39]. RNNs store past sequence information by feeding layer output from previous network activations into the input of current activations. Researchers have also found that iterating over a sequence in both directions can provide contextual information for layers further in the network[40]. Output from DNNs can also be fed into other machine learning models such as CRFs. Bi-directional Long Short-Term Memory - CRF networks (BiLSTM+CRF) were the state of the art technique for many Biomedical NER tasks before the development of transformer networks[41]. Transformer networks are a new style of non-recurrent neural networks that utilize attention mechanisms in conjunction with query key value (QKV) embeddings[42]. Attention is a scored system that indicates how relevant a given network output is to the currently processing input. In the context of NER, it can indicate how important other words in a sentence are for classifying the given word. One method of assessing attention is the use of cosine similarity. In this work, we use BiLSTM+CRF DNN which is described in more detail in Section 3.4.

Feature Representation As most supervised learning methods require numerical values to run, words (also referred to as tokens) from sentences require processing into features. Features can represent numerous characteristics of a token, such as its placement in a sentence, dependencies, or part of speech.

One of the earliest methods of representing tokens is through the use of one-hot encodings where each word maps to a single position in a vector. This allows for quick representation of n-grams, or a window of nearby words. One of the major downsides of one-hot encodings that have resulted in decreased use for direct word representation is the linear growth in dimensionality with increase of new vocabulary.

In the place of one-hot encodings, encodings that utilize context or dimensionality reduction have appeared. Clustering approaches attempt to take tokens and cluster them into groups where similar features or context are found[43]. One example of dimensionality reduction involves finding a co-occurrence matrix of tokens and performing principal component analysis[44].

With the introduction of neural networks, new encoding methods have appeared that are now heavily used in NER tasks. A basic method of using neural networks for encodings is to develop an auto-encoder network, where a simple feed-forward neural network is trained to map an input index to an output index[45][46]. The hidden layer values for the token are used as the feature for the token. A more modern and contextually aware method of generating embeddings is the Word2Vec algorithm[38]. The Word2Vec network utilizes either bag of words or skip-grams, and attempts to map tokens to other tokens that would be found together in sentences. Mapping to other tokens allows the network to model potential semantic dependencies between words and acts as a more robust way to represent tokens in sentences.

Recently contextualized embeddings have been introduced which incorporates the context surrounding the specific word when formulating its embedding representation. For example, BERT which use transformers to learn a masked language model at both the token and sentence level resulting in different embedding for each word based on its positional information within a sentence[47]; XLNet which also uses transformers, however it learns byte-pair encodings at the token level only[48]; and ELMo which uses a BiLSTM to learn character-level embeddings which allow it to handle out-of-vocabulary words[49].

CHAPTER 3

BACKGROUND

3.1 Tools

3.1.1 SpaCy

SpaCy[50] is an open-source Python NLP library written mostly in C for rapid processing of text. It provides built-in methods and dictionaries for a variety of NLP tasks. Relevant to this study, it contains specific methods for parsing part-of-speech and dependency information from text while maintaining original tokenization.

3.1.2 Tensorflow

Tensorflow[51] is a Python based end-to-end machine learning platform typically used in the creation of advanced neural network models. It was originally designed for internal use at Google before the company open-sourced the project and released it publicly.

3.1.3 Gensim

Gensim[52] is a Python library used for NLP applications. It is being used solely for loading of a trained Word2Vec model and mapping input tokens to vectors.

Table 1. 2010 i2b2 Clinical Dataset Information

i2b2	**Text**			**Annotations**		
Set	**Files**	**Sentences**	**Tokens**	**Problem**	**Test**	**Treatment**
Train	170	16315	149541	7073	4608	4844
Test	256	27625	267249	12592	9225	9344

3.2 Corpora

3.2.1 2010 i2b2 Clinical Dataset

The 2010 i2b2 Clinical Dataset[53] is an annotated set of clinical data comprised of discharge summaries from three healthcare systems, Partners Healthcare, Beth Israel Deaconess Medical Center, and the University of Pittsburgh Medical Center. The dataset was annotated for concept extraction, assertion classification and relation classification, and was created to support the 2010 i2b2/VA shared task. The dataset contains annotations for three concepts, problems, tests, and treatments. This dataset was chosen as the gold standard corpus for our study. It has been used in literature as a benchmark for many Clinical NER tasks and provides a good platform for evaluating information extraction methods. Table 1 lists the exact details for the dataset. In particular, the training set contains 170 annotated documents with 16,525 annotations over 149,541 tokens; and the test data set contains 256 annotated documents with 31,161 annotations over 267,249 tokens.

Problems, tests, and treatments have specific definitions to limit their scope. Problems, or medical problems, are phrases that contain observations made by patients or clinicians about the patient's body or mind that are thought to be abnormal or caused by a disease. Tests are phrases that describe procedures, panels, and mea-

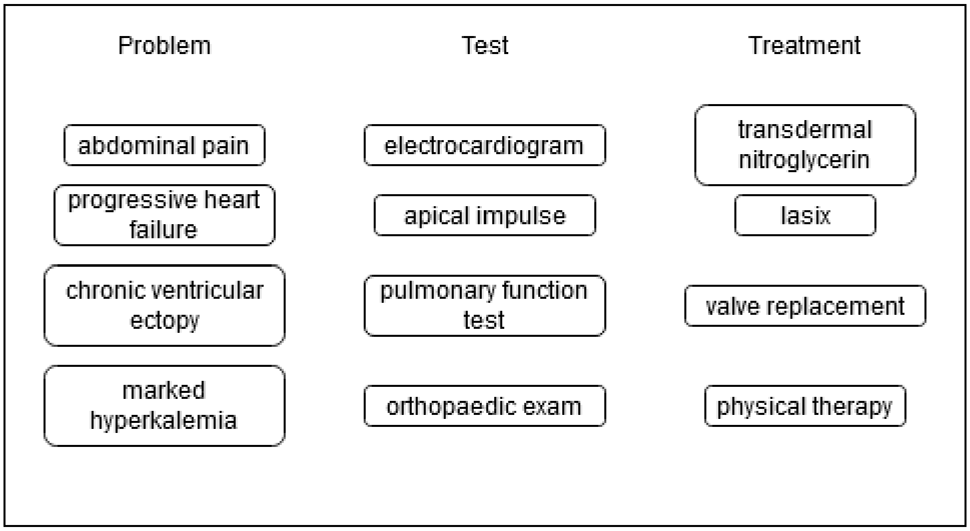


Fig. 1. Examples of Problems, Tests, and Treatments.

sures that are done to a patient or a body fluid or sample in order to discover, rule out, or find more information about a medical problem. Treatments are phrases that describe procedures, interventions, and substances given to a patient in an effort to resolve a medical problem. Examples of each of these types can be seen in Figure 3.2.1.

3.2.2 MIMIC-III

The MIMIC Critical Care Database[54] (MIMIC-III) is a freely available de-identified dataset comprising of electronic health records (EHRs) for over 40,000 critical care patients from the Beth Israel Deaconess Medical Center. It was collected between 2001 and 2012 and includes documents ranging from caregiver notes to imaging reports and prescribed medications. In particular, the caregiver notes section is comprised of 2,083,180 individual documents from a variety of internal departments with a total token count of 487,639,049. Caregiver notes can contain information such as doctor or nurse reviews of patients and potential diagnostic information alongside

Table 2. Gazetteer Annotation Information

Source	Problem	Test	Treatment
ICD10CM	86,168	-	-
ICD10PCS	-	303	59
CPT	-	4,813	8,571
WebMD	-	17,676	-
FDA Drug List	-	-	42,639
Southern Cross	-	-	2,599

treatment guidelines. The caregiver discharge notes were used by this study as the primary data source for pseudo-data generation.

3.3 Gazetteer Annotation Sources

In this section we list the gazetteers used as annotation sources for pseudo-data generation. The number of annotations for the three entities found in each gazetteer is shown in Table 2. Additional information is provided in each sources respective section.

3.3.1 ICD10

The International Statistical Classification of Diseases and Related Health Problems (ICD) is a medical classification list generated by the World Health Organization that standardizes coding for medical terms across the globe[1]. The 10th revision was first put into use in 1994. In the United States, ICD was modified into ICD-CM and ICD-PCS. ICD-CM is the clinical modification of ICD for classifying diagnosis and

[1] https://www.cms.gov/Medicare/Coding/ICD10

reasons for hospital visits. It is split into 21 chapters of differing categories. There was 5,562 entries in the system that were used as a source for problem annotations. ICD-PCS is the modification of ICD for standardizing procedure coding systems. All of the tests and treatments used within the United States are codified in this database. There were 1,065 tests and 9,165 treatments coded in the database that were used for annotation sources.

3.3.2 CPT

Current Procedural Terminology (CPT) is another medical code set used in the United States and maintained by the American Medical Association[2]. It contains information for surgical and diagnostic procedures and is released annually. There were 1,346 tests and 3,958 treatments coded in the database that were used as annotation sources for this study.

3.3.3 FDA Approved Drug List

The Food and Drug Administration (FDA) Approved Drug List is a current listing of all drugs that have been given approval in the United States to be distributed either over the counter or by practicing physicians. Drugs are listed by both their brand name and generic chemical name along with application method and distributing company. The database is freely available online at the FDA website[3]. A list of current drugs was downloaded and used as a source of potential treatment annotations for our system. A total of 8,906 treatments were present in the list.

[2] https://coder.aapc.com/cpt-codes/

[3] https://www.accessdata.fda.gov/scripts/cder/daf/index.cfm

3.3.4 WebMD Medical Tests

WebMD is an American company that provides news and information relating to medical care and diagnosis and was founded in 1996. The company maintains the WebMD website[4] and features glossaries for medical terms. One glossary is a list of medical tests and information about them. The list of terms from the glossary[55] was extracted and contained 625 tests.

3.3.5 Southern Cross Surgery List

Southern Cross is a healthcare system based out of New Zealand that maintains their own coding database for all surgeries performed at their locations. This list of surgeries was published online in 2014[56] and contains 43 pages of procedure codes along with text representations for each which totaled 761 treatments. This list was extracted and used as a source of treatment annotations for our system.

3.4 Named Entity Recognition (NER) System

In this study, we use a Bi-directional Long Short-Term Memory (BiLSTM) Recurrent Neural Network. The feature vectors are split by token and sentences are fed into the network along with a scalar representing the number of words in each sentence. Each sentence is read by two separate LSTM cells. One cell reads the information left to right, and the other reads right to left. The output is concatenated and shaped into a flat vector and fed into a dropout layer and a linear dense layer. The data is then reshaped back into the original token and sentence format and classified using a conditional random field (CRF). Figure 3(a) shows an high-level view of the BiLSTM+CRF architectures. In the remainder of this section, we describe each of

[4] https://www.webmd.com/a-to-z-guides/tests

Table 3. Gazetteer Model Hyperparameters

Parameter	**Value**
Initialization Type	Variance Scaling
Initialization Factor	2
Initialization Mode	FAN_IN
LSTM Layer Size	256
LSTM Initial Forget Bias	1
Training Dropout	0.5
Loss Function	Log-Likelihood
Optimizer	Adam
Learning Rate	0.005

the above components in more detail. Table 3 specifies the hyperparameters used in the network.

Bi-directional LSTM. LSTMs allow for weighted retention of information from previous instances in a specified window. It achieves this through two recurring input/output pairs, H (the LSTM output) and C (the internal memory state). The LSTM output (H) represents a basic view of the cell state between each iteration. In comparison, the internal memory state (C) tries to keep information from multiple iterations back based on trained weights. This difference between gates is where the name ”long short” comes from in LSTM. An example LSTM cell can be seen in Figure 2. Given a feature vector X at point t:

$$LSTM(X_t, H_{t-1}, C_{t-1}) = H_t, C_t$$

The internal memory of the cell is a function of the previous memory state, the previous output, and the current input and is given by the following formulas, where

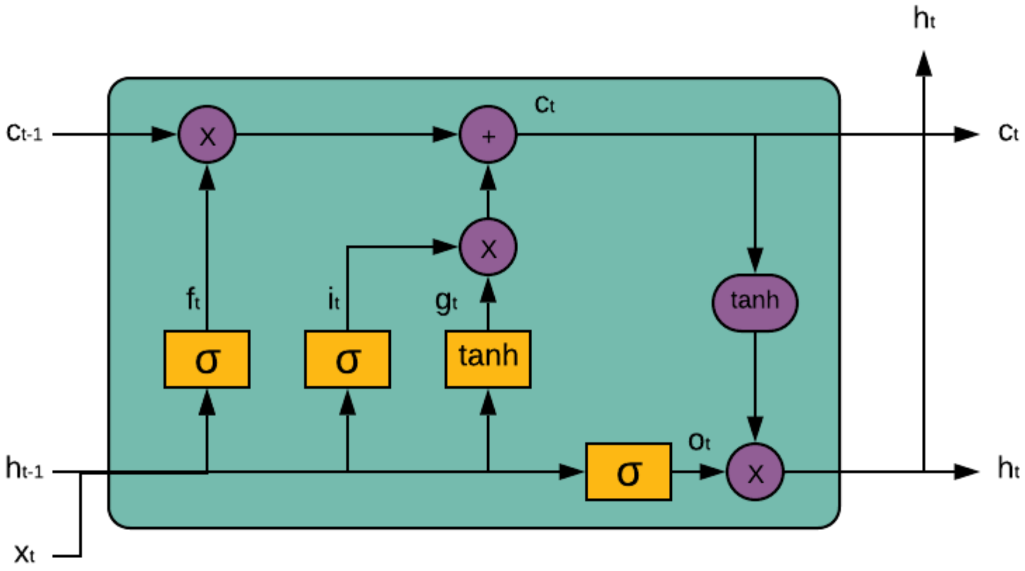


Fig. 2. LSTM Cell

U and W are learned weights:

$$f_t = \sigma(X_t U_f + H_{t-1} W_f)$$

$$i_t = \sigma(X_t U_l i + H_{t-1} W_i)$$

$$O_t = \sigma(X_t U_o + H_{t-1} W_o)$$

The input to the cell and the previous cell state are multiplied by weight matrices, summed, and a sigmoid activation function is applied. This happens for f_t, i_t and O_t, with different learned weight matrices.

$$g_t = tanh(X_t U_g + H_{t-1} W_g)$$

The input to the cell and the previous cell state are multiplied by weight matrices, summed, and a hyperbolic tangent activation function is applied.

$$C_t = (C_{t-1} f_t) + (g_t i_t)$$

The supplementary cell state (C_t), is multiplied by f_t and added to the multiplication of i_t and g_t.

The output H is a function of the internal cell state and the input and is given by the following:

$$H_t = O_t * tanh(C_t)$$

The current cell state is the product of O_t and the hyperbolic tangent of the supplemental cell state.

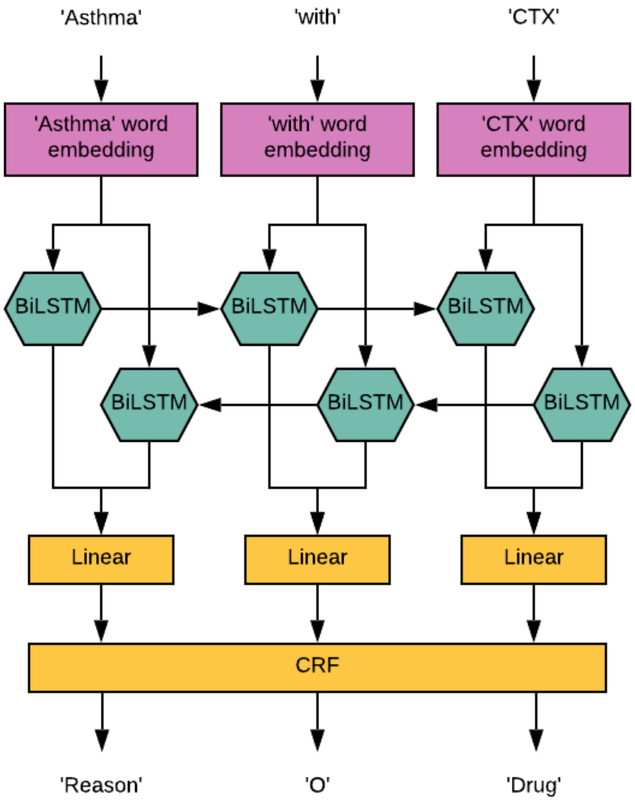


Fig. 3. Example architecture of a BiLSTM+CRF model.

3.4.1 Feature Vector Generation

A simplified representation of the way we generated feature vectors can be seen in Figure 4(a). We loaded text documents containing already separated sentences into memory alongside their respective annotation files and mapped them to one another, marking information for annotation locations and types. Sentences were split by token. We used pre-processing to remove symbols and convert numbers, dates, and times into standardized values. Original sentence positions were marked for reconstruction after network annotation. We ran each sentence through the SpaCy POS/DEP parser to get POS and DEP tags for each token and mapped them to locations in a one-hot vector. We then ran each token through a Word2Vec model that was trained on the MIMIC-III clinical notes database. The output of this model represented the first 200 values of the vector. The POS/DEP tags were appended and represented the rest of the vector for each token. The final shape of the resulting matrix was [sentences, tokens, features].

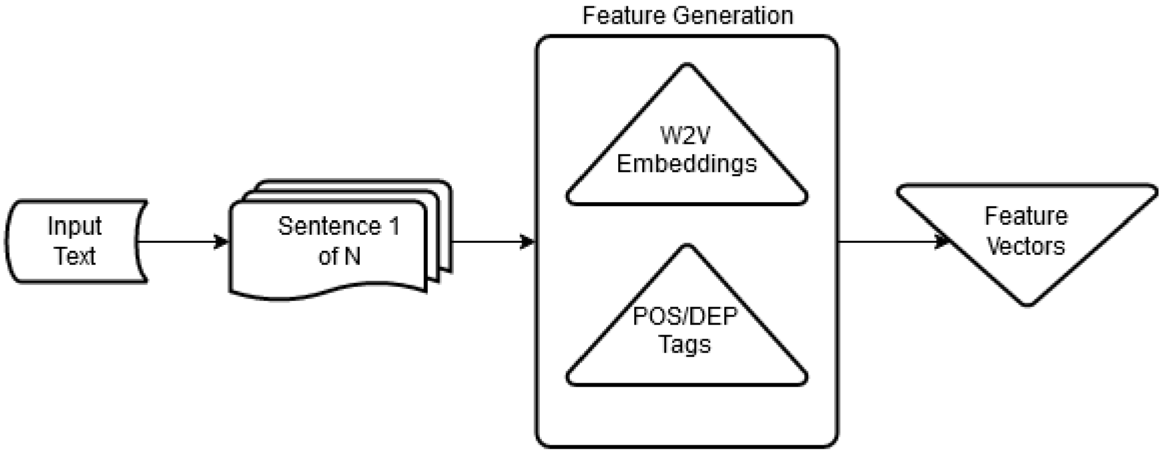


Fig. 4. Simplified Representation of Feature Vector Generation.

3.5 Evaluation

We utilized the F_1 metric for our experiments. F_1 Score is the harmonic mean between precision and recall. Precision is a measure of how accurate each prediction was and utilizes the following formula:

$Precision = \frac{tp}{tp+fp}$

where tp is true positive and fp is false positive. Recall is a measure of how many of the total tokens were predicted and utilizes the following formula:

$Recall = \frac{tp}{tp+fn}$

where tp is true positive and fn is false negative. The formula representing the geometric mean between the two is defined as:

$F_1 = 2 * \frac{Precision*Recall}{Precision+Recall}$

We evaluate our system at both the token and entity level. For the entity level evaluation we use two methods: strict and lenient. Strict requires a true positive to have exact span and tag matches. Lenient requires a partial match of the span with the correct tag. If two or more tags match a phrase, it will only be counted as true positive once, and the additional phrase will not be counted as a false.

CHAPTER 4

GAZETTEER ANNOTATING

In this section, we describe the experiments we conducted on gazetteer annotating and the reasoning behind them. The experiments are split into: 1) single-class experiments and 2) multi-class experiments, where multi-class attempts to label all entities within a single model. A full listing can be found in Table 4. The first set of experiments we performed take each individual gazetteer source and train a model with an inverse epoch structure. That is, a set number of epochs is defined and the gazetteer source is trained on for a given number of epochs. Then for the rest of the epochs, the model is trained on the i2b2 training data. This provides a way to determine what kind of effect each gazetteer is having on the model in an isolated environment. The next experiment takes every gazetteer source and trains them together on a multi-class model. This allowed us to determine if there were cumulative effects from including all of the gazetteers. Finally, we conducted two experiments only taking the best performing gazetteers. The first followed the same pattern as the previous experiments. The second kept the number of epochs for the i2b2 training static and changed how many epochs we pre-trained on the gazetteer data for.

4.1 Methods

4.1.1 Preparatory Steps

We downloaded the MIMIC-III database, extracted discharge summaries, and ran them through pre-processing. All gazetteer sources were downloaded and also pre-processed. Gazetteer annotations were matched to text in the MIMIC-III discharge

Table 4. Gazetteer Annotation Experiments

Name	**Model Type**	**Sources**	**Description**
ICD10CM Problem	Single-Class	ICD10CM	Measure of how much ICD10CM problem pre-training changes precision and recall in comparison to baseline.
CPT Test	Single-Class	CPT	Measure of how much CPT test pre-training changes precision and recall in comparison to baseline.
ICD10PCS Test	Single-Class	ICD10PCS	Measure of how much ICD10PCS test pre-training changes precision and recall in comparison to baseline.
WebMD Test	Single-Class	WebMD Test List	Measure of how much WebMD test pre-training changes precision and recall in comparison to baseline.
CPT Treatment	Single-Class	CPT	Measure of how much CPT treatment pre-training changes precision and recall in comparison to baseline.
ICD10PCS Treatment	Single-Class	ICD10PCS	Measure of how much ICD10PCS treatment pre-training changes precision and recall in comparison to baseline.
FDA Treatment	Single-Class	FDA Drug List	Measure of how much FDA treatment pre-training changes precision and recall in comparison to baseline.
Southern Cross Treatment	Single-Class	Southern Cross Surgery List	Measure of how much Southern Cross treatment pre-training changes precision and recall in comparison to baseline.
Comprehensive	Multi-Class	All	Measure of how much all gazetteers pre-trained together changes precision and recall in comparison to baseline.
Selective	Multi-Class	ICD10CM, WebMD Test List, FDA Drug List, Southern Cross Surgery List	Measure of how much top gazetteers pre-trained together changes precision and recall in comparison to baseline.
Selective Static Epoch	Mutli-Class	ICD10CM, WebMD Test List, FDA Drug List, Southern Cross Surgery List	Measure of how much top gazetteers pre-trained together changes precision and recall in comparison to baseline when baseline has static epochs.

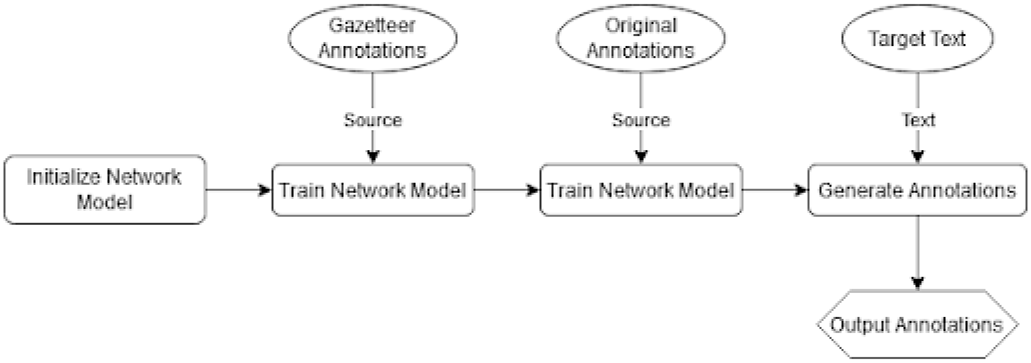


Fig. 5. Visual Pipeline Representing Gazetteer Annotation.

summaries and set aside as annotation files. A visual guide to the pipeline can be seen in Figure 6.

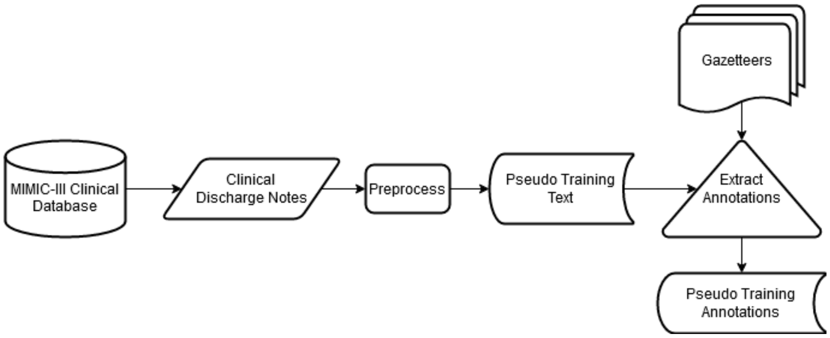


Fig. 6. Visual Pipeline Representing Extraction of Pseudo Annotations from the MIMIC-III corpus.

4.2 Experimental Details

Preprocessing. Text for discharge summaries was extracted from the ’NoteEvents’ table where ’Category’ was set to ’Discharge summary’. Pre-processing started with an initial step of combining all de-identified terms into single terms that could be easily turned into features, including numbers, dates, and times. Punctuation was then modified to match the format that the i2b2 dataset was in. To generate a random sampling of MIMIC data, all pre-processed sentences were loaded into memory.

Sentences less than 8 tokens in length were removed to obtain data similar to i2b2. 400,000 sentences were then randomly selected and sent to files. Annotations from each gazetteer source were scanned for in the extracted MIMIC text and annotation files were generated. For multi-class experiments, annotation files from different sources were merged. In cases where there were overlapping annotations of different classes, each annotation was thrown out.

Feature representation. Generated annotations and associated text were converted into feature vectors with the following features: Word2Vec embeddings (length 200), is a number, date, or time, part of speech tags (length 19), and dependency tags (length 52). The feature vectors were in the form [sentence, word, features] for feeding into the BiLSTM+CRF architecture.

Model Parameters. Training occurred for 15 epochs in each experiment unless otherwise stated. This number was chosen by training the network over a large amount of epochs numerous times and selecting the point where additional training produced negligible results. The BiLSTM+CRF architecture was pre-trained on the MIMIC-III annotations for n epochs; and then fine-tuned over the i2b2 training data for the remaining $15 - n$ epochs. The flow for the training can be seen in Figure 7. Training utilized batch back propagation with an Adam optimizer at a learning rate of 0.005. Every model was initialized and trained independently.

Evaluation. Evaluation was performed against the i2b2 test annotation set. When generating annotations, output from the model went through post-processing to allow for consistent tagging at the phrase level. This included removing tags from specific punctuation and ensuring consistent tags between certain token types. These changes were applied after token level evaluation and before phrase level. Token level metrics included precision, recall, and F_1 score. Phrase level metrics included lenient and strict phrase level F_1 scoring.

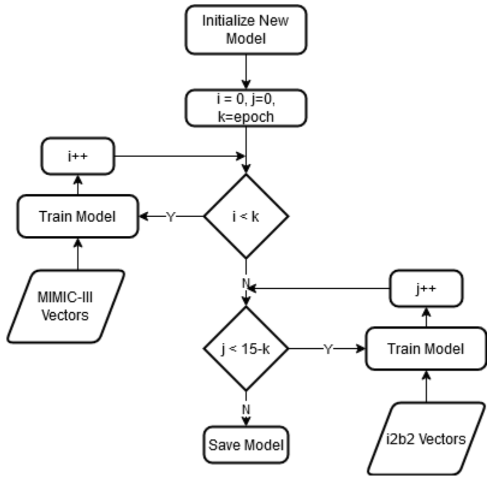


Fig. 7. Gazetteer Model Training.

4.3 Results and Discussion

4.3.1 Single class Gazetteer Pretraining

In this section, we present the results of our inverse epoch structure experiments over each individual gazetteer source. This evaluation provides a way to determine the effect each gazetteer is having on the model in an isolated environment.

4.3.1.1 Problem Entity Type

ICD10CM Problem Gazetteer Pretraining. Figure 8 shows the token level precision, recall, and F_1 scores of the problem entity type when we pretrained a single-class model on annotations from ICD10CM. The precision of the model rises above baseline when trained 1 or 2 epochs then declines until 10 epochs where it remains roughly the same before finally diving 10 points. The recall follows an inverse pattern where it initially declines, then by 5 epochs goes above the baseline. After six epochs, the network has a continuous decline before the final dropoff. When compared to the multi-class model, the precision and recall exhibit the same general pattern but with a larger range in numbers. Out of the 4349 annotations available in ICD10CM, only 620 and 678 of them were found in the i2b2 test and train datasets respectively. This represents the highest percentage found in any of the gazetteers (Table 6).

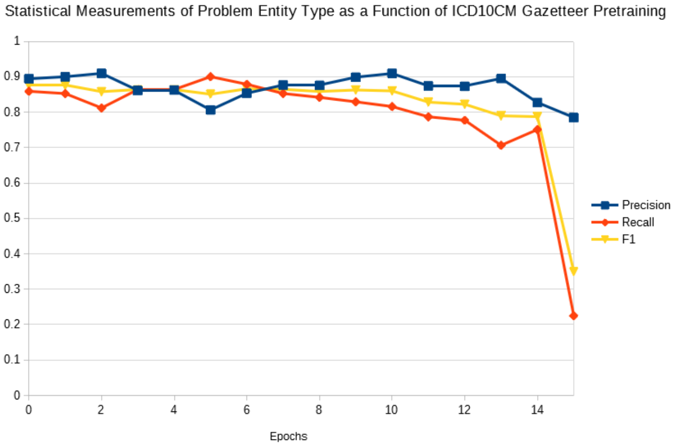


Fig. 8. Statistical Measurements of Problem Entity Type as a Function of ICD10CM Gazetteer Pretraining

4.3.1.2 Test Entity Type

CPT Test Gazetteer Pretraining. Figure 9 shows the token level precision, recall, and F_1 scores of the test entity type when we pretrained a single-class model on annotations from CPT. Precision and recall oscillate below and above baseline as the number of trained epochs increases. The oscillating pattern found here can also be seen in the multi-class model. Out of the 1154 annotations available in CPT for the test annotation group, only 64 and 67 were found in the i2b2 test and train datasets respectively (Table 6).

ICD10PCS Test Gazetteer Pretraining. Figure 10 shows the token level precision, recall, and F_1 scores of the test entity type when we pretrained a single-class model on annotations from ICD10PCS. Like the CPT test results, the precision and

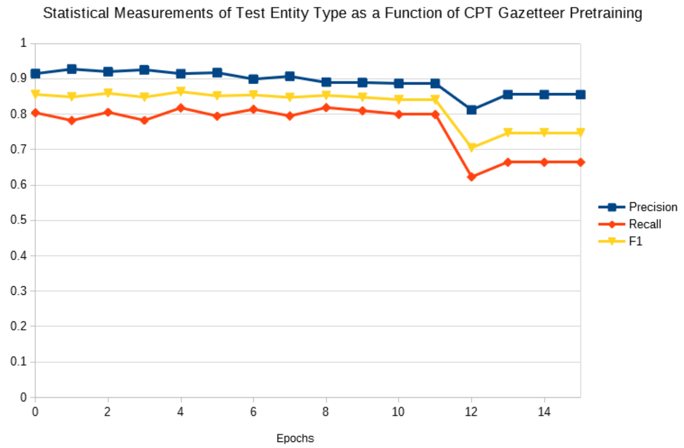


Fig. 9. Statistical Measurements of Test Entity Type as a Function of CPT Gazetteer Pretraining

recall oscillate around baseline. Between 4 and 7 epochs we get an increase in recall at the cost of precision. Out of the 467 annotations available in ICD10PCS for the test annotation group, only 12 were found in both the i2b2 test and train datasets (Table 6).

WebMD Test Gazetteer Pretraining. Figure 11 shows the token level precision, recall, and F_1 scores of the test entity type when we pretrained a single-class model on annotations from WebMD. Initial inclusion of the WebMD annotations from one epoch and onward result in a fairly large (10%) increase in precision. The recall inverses and falls, though not enough to offset the precision gains. Out of the 675 annotations available in WebMD, only 136 and 141 were found in the i2b2 test and train datasets respectively (Table 6).

4.3.1.3 Treatment Entity Type

CPT Treatment Gazetteer Pretraining. Figure 12 shows the token level precision, recall, and F_1 scores of the treatment entity type when we pretrained a single-

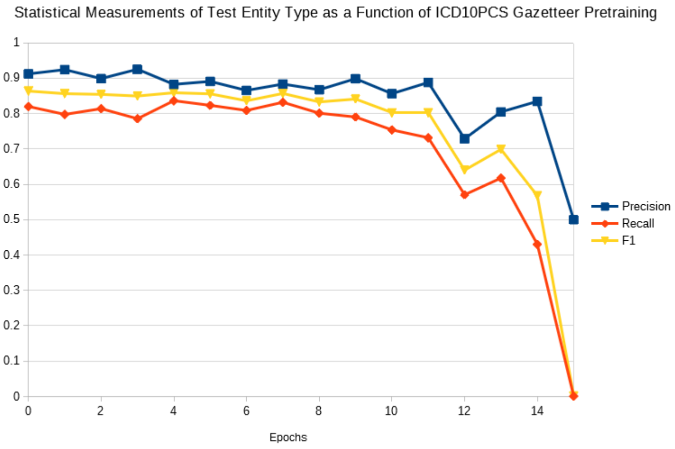


Fig. 10. Statistical Measurements of Test Entity Type as a Function of ICD10PCS Gazetteer Pretraining

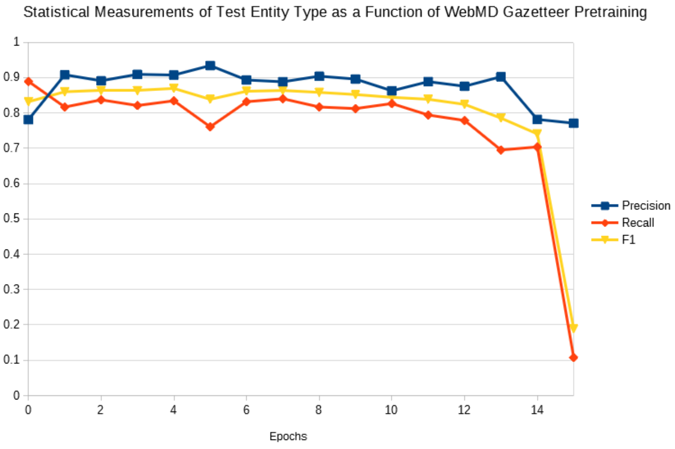


Fig. 11. Statistical Measurements of Test Entity Type as a Function of WebMD Gazetteer Pretraining

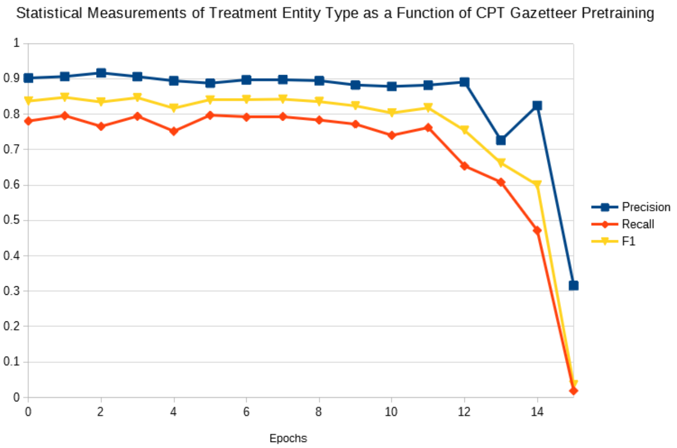


Fig. 12. Statistical Measurements of Treatment Entity Type as a Function of CPT Gazetteer Pretraining

class model on annotations from CPT. Precision and recall oscillate slightly around baseline before suffering a steep dropoff. Out of the 1385 annotations available in CPT for the treatment annotation group, only 50 were found in both the i2b2 test and train datasets (Table 6).

ICD10PCS Treatment Gazetteer Pretraining. Figure 13 shows the token level precision, recall, and F_1 scores of the treatment entity type when we pretrained a single-class model on annotations from ICD10PCS. Precision and recall both hover slightly below baseline when trained for a few epochs. Like previous experiments, the recall tanks at the end. Out of the 790 annotations available in ICD10PCS for the treatment annotation group, only 21 and 23 were found in the i2b2 test and train datasets respectively (Table 6).

FDA Drug Label Treatment Gazetteer Pretraining. Figure 14 shows the token level precision, recall, and F_1 scores of the treatment entity type when we pretrained a single-class model on annotations from the FDA Drug List. The inclusion of the FDA Drug List has a large impact (>10%) on recall when included in a single epoch

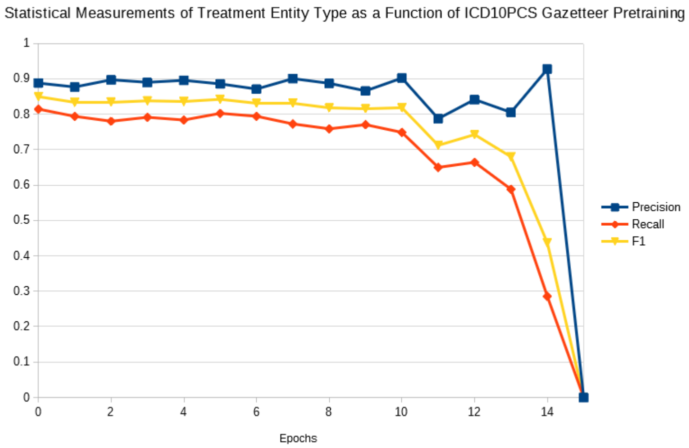


Fig. 13. Statistical Measurements of Treatment Entity Type as a Function of ICD10PCS Gazetteer Pretraining

and onward until past 10 epochs. There is a small hit to precision (2%) initially and a larger hit (5-7%) from 3 epochs to 12. Out of the 7230 annotations available in the FDA Drug List, only 338 and 405 were found in the i2b2 test and train datasets respectively (Table 6).

Southern Cross Treatment Gazetteer Pretraining. Figure 15 shows the token level precision, recall, and F_1 scores of the treatment entity type when we pretrained a single-class model on annotations from the Southern Cross Surgery List. When included for a single epoch or four or more, precision gains a moderate amount (5%) while recall declines a similar amount. This pattern holds until 10 epochs of pre-training where the recall dives towards 0. Out of the 800 annotations available in the Southern Cross Surgery List, only 103 and 111 were found in the i2b2 test and train datasets respectively (Table 6).

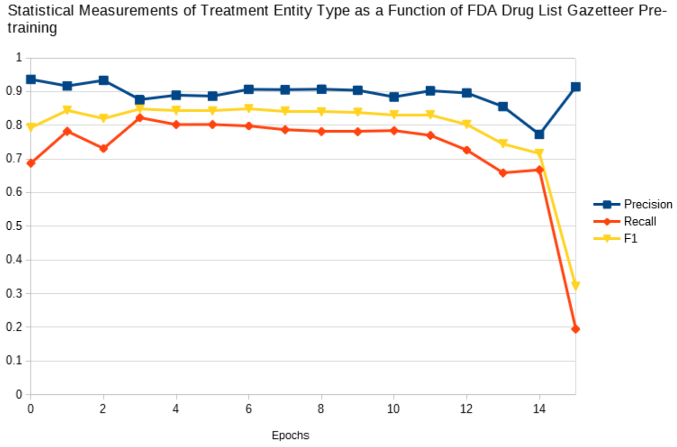


Fig. 14. Statistical Measurements of Treatment Entity Type as a Function of FDA Drug List Gazetteer Pretraining

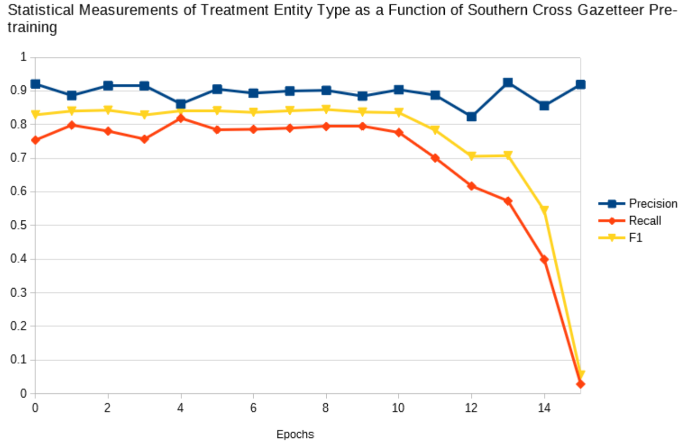


Fig. 15. Statistical Measurements of Treatment Entity Type as a Function of Southern Cross Gazetteer Pretraining

Table 5. Gazetteer Annotation Multi-Class Phrase Strict (Lenient) F_1 Measurement

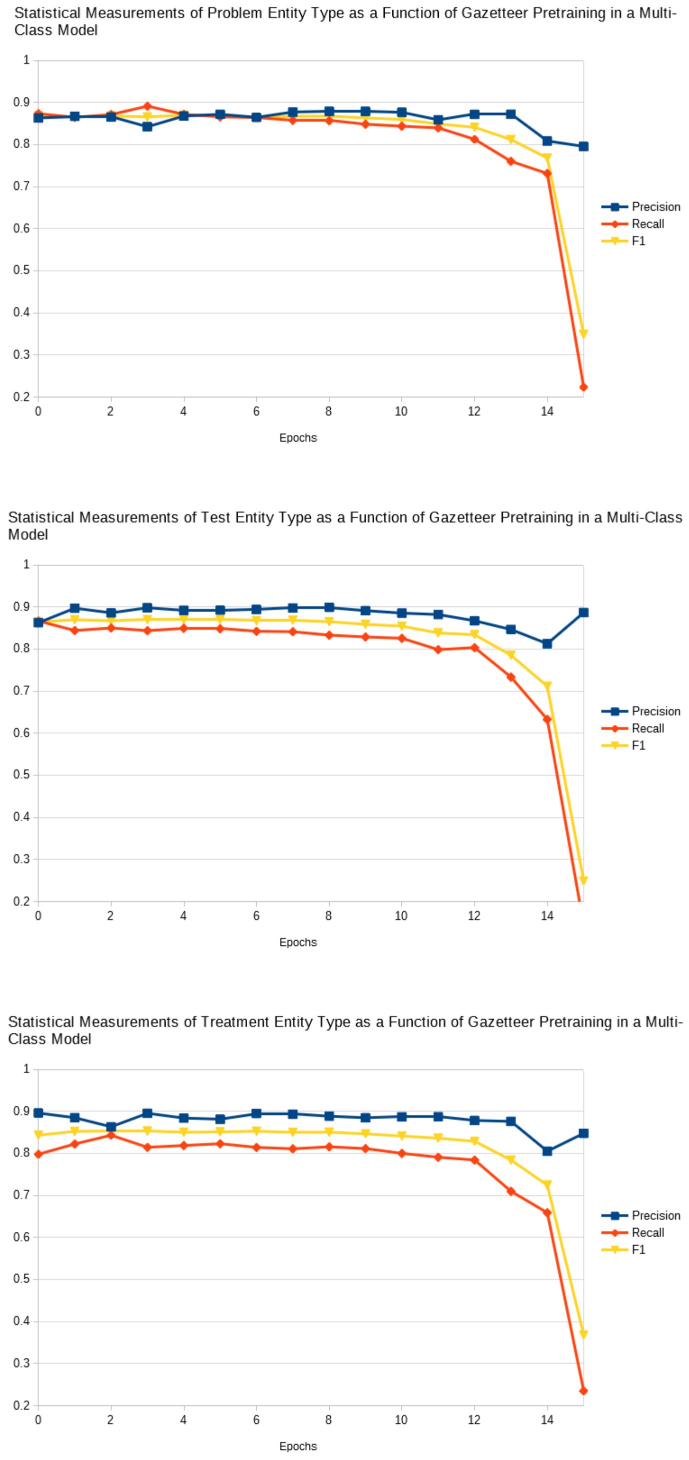
Source	**Class**	**Precision**	**Recall**	F_1
Baseline	Problem	0.696(**0.919**)	0.631(0.815)	0.662(0.864)
	Test	**0.793**(**0.902**)	**0.762**(0.862)	**0.777**(**0.881**)
	Treatment	0.706(0.882)	0.681(**0.834**)	0.693(0.857)
Selective Epoch 5	Problem	**0.718**(0.91)	0.655(0.821)	**0.685**(**0.864**)
	Test	0.787(0.902)	0.754(0.86)	0.77(0.88)
	Treatment	**0.774**(**0.913**)	0.705(0.824)	**0.738**(**0.866**)
Selective Static Epoch 5	Problem	0.702(0.899)	**0.657**(**0.831**)	0.679(0.863)
	Test	0.742(0.885)	0.739(**0.868**)	0.74(0.877)
	Treatment	0.764(0.904)	**0.709**(0.831)	0.735(0.866)

4.3.2 Multi-Class Gazetteer Pretraining

In this section, we evaluate our model by combining the gazeteer sources and training them in a multi-class environment. This allows us to determine the cumulative effects of including all sources.

Comprehensive Multi-Class Gazetteer Pretraining.

Figure 16 shows the token level precision, recall, and F_1 scores of the problem, treatment, and test labels respectively when we merged the annotations from all the gazetteers and trained them on a multi-class model. The experiment was run three times and the results averaged. With the problem and treatment entity types, pretraining for one to five epochs results in a small (2%) increase in recall with a nearly equivalent loss in precision. Training for longer periods of time results in a steep decrease in recall which also results in a steep decline in F_1 score. Precision remains



Statistical Measurements of Problem Entity Type as a Function of Gazetteer Pretraining in a Multi-Class Model

Statistical Measurements of Test Entity Type as a Function of Gazetteer Pretraining in a Multi-Class Model

Statistical Measurements of Treatment Entity Type as a Function of Gazetteer Pretraining in a Multi-Class Model

Fig. 16. Statistical Measurements of Problem (Top), Test (Middle) and Treatment (Bottom) Entity Types as a Function of Gazetteer Pretraining in a Comprehensive Multi-Class Model

fairly high. The test entity type loses recall and gains precision at the same level as the previous entity types when we incorporate the gazetteer pretraining.

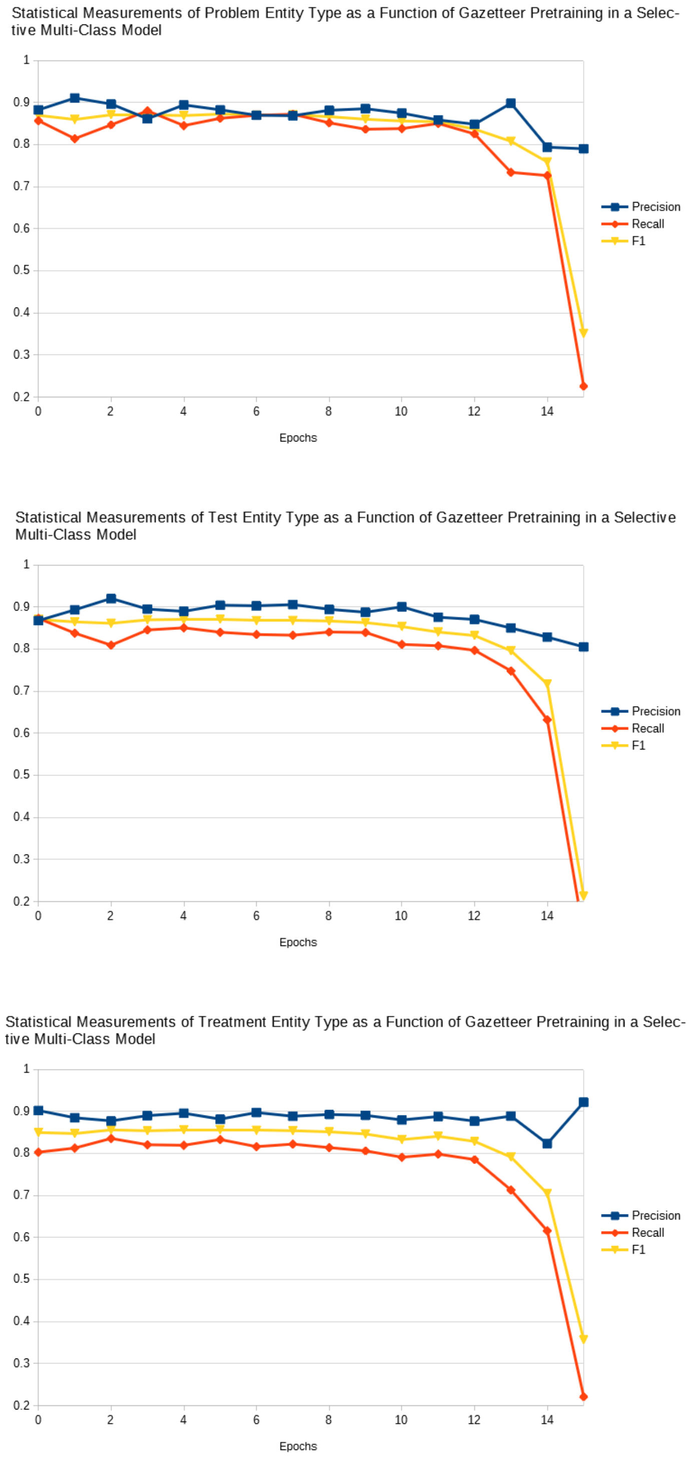
Static Multi-Class Gazetteer Pretraining.

Figure 17 shows the token level precision, recall, and F_1 scores of the problem, treatment, and test labels respectively when we merged the annotations from only the best the gazetteers and trained them on a multi-class model. The specific gazetteers we used were ICD10CM (Test), WebMD Test List, FDA Drug List, and the Southern Cross Surgery List. For problem and test, the precision rises for several epochs when including the gazetteer data before eventually dropping off. The baseline treatment precision remains the highest but doesn't drop off drastically. The recall for the problem and treatment types also rise a couple of points whereas the recall for the test type suffers. In all three instances, the highest F_1 score is observed on the 5/10 ratio.

Selective Static Multi-Class Gazetteer Pretraining.

Figure 18 shows the token level precision, recall, and F_1 scores of the problem, treatment, and test labels respectively when we merged the annotations from only the best the gazetteers and trained them on a multi-class model with a set number of epochs on the i2b2 dataset. The specific gazetteers we used were the same as in the selective multi-class experiment.

For the problem type, precision increases all the way to the max number of epochs on the gazetteer data. Recall stops increasing after the 6th epoch and drops off rapidly afterwards. In most of the chart, an inverse pattern between recall and precision can be observed. For the test type, precision increases for one epoch after the baseline but never again. Recall never increases. The test type benefits most from no gazetteer data. For the treatment type, precision increases slightly on the

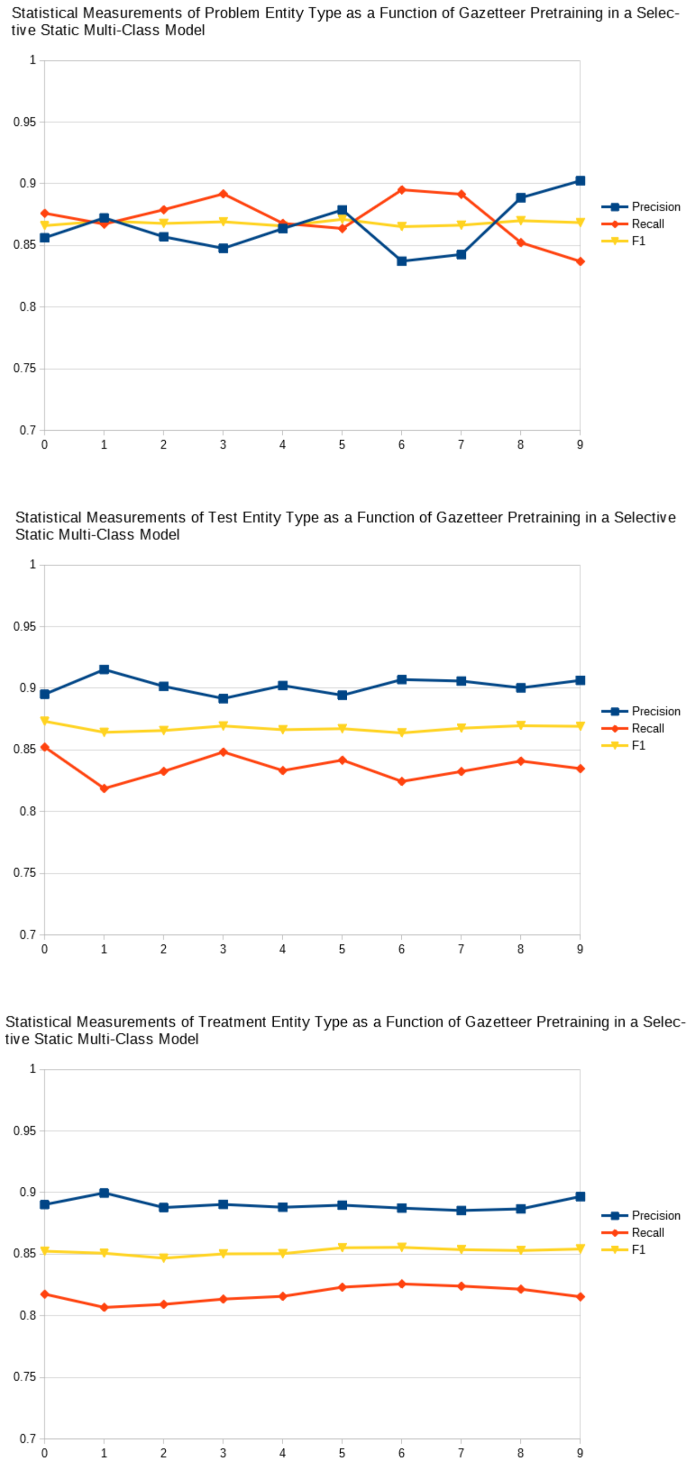


Statistical Measurements of Problem Entity Type as a Function of Gazetteer Pretraining in a Selective Multi-Class Model

Statistical Measurements of Test Entity Type as a Function of Gazetteer Pretraining in a Selective Multi-Class Model

Statistical Measurements of Treatment Entity Type as a Function of Gazetteer Pretraining in a Selective Multi-Class Model

Fig. 17. Statistical Measurements of Problem (Top), Test (Middle) and Treatment (Bottom) Entity Types as a Function of Gazetteer Pretraining in a Selective Multi-Class Model


Statistical Measurements of Problem Entity Type as a Function of Gazetteer Pretraining in a Selective Static Multi-Class Model

Statistical Measurements of Test Entity Type as a Function of Gazetteer Pretraining in a Selective Static Multi-Class Model

Statistical Measurements of Treatment Entity Type as a Function of Gazetteer Pretraining in a Selective Static Multi-Class Model

Fig. 18. Statistical Measurements of Problem (Top), Test (Middle) and Treatment (Bottom) Entity Types as a Function of Gazetteer Pretraining in a Selective Static Multi-Class Model

Table 6. Gazetteer Annotation Term Crossover Analysis

Class	Gazetteer	# Types	# Types in Mimic (%)	# Types Matching i2b2 Train (%)	# Types Matching i2b2 Test (%)
Problem	ICD10CM	4349	1872 (4.30E-01)	620 (1.17E-01)	678 (9.44E-02)
Test	CPT	1154	127 (1.10E-01)	64 (1.21E-02)	67 (9.33E-03)
	ICD10PCS	467	31 (6.64E-02)	12 (2.27E-03)	12 (1.67E-03)
	WebMD	675	345 (5.11E-01)	136 (2.57E-02)	141 (1.96E-02)
Treatment	CPT	1385	98 (7.08E-02)	50 (9.46E-03)	50 (6.96E-03)
	ICD10PCS	790	29 (3.67E-02)	21 (3.98E-03)	23 (3.20E-03)
	FDA	7230	2213 (3.06E-01)	338 (6.40E-02)	405 (5.64E-02)
	Southern	860	199 (2.31E-01)	103 (1.95E-02)	111 (1.55E-02)

first epoch then hovers around baseline. Recall increases towards the latter epochs and falls back down at max epochs. The highest F_1 scores were observed around 5/6 epochs except for test.

4.3.3 Overall Results

Throughout all of the gazetteering experiments, a common trend we observed was a trade-off between precision and recall. As precision increased, recall decreased, and vice versa. In most cases, the increase was high enough to make up for the decrease resulting in an increase of the F_1 score. When examining the strict and lenient phrase F_1 scores, problem and treatment were able to exceed the baseline while test was not.

Some gazetteers outperformed others by several points. To try and understand the reason why, we analyzed the number of terms that overlapped between the gazetteers and the datasets (Table 6). What we found is that gazetteers that had a higher number of terms in the datasets had a higher impact. This is a logical conclusion, as more relevant data will results in better outcomes. What was surprising is the amount of crossover some gazetteers had. ICD10PCS and CPT codes had a large number of unique terms but very little crossover. This indicates that the vocabulary used in billing codes does not match what is used by medical professionals. Public lists, such as the WebMD test list and the Southern Cross surgery list, had a high level of crossover. The FDA drug list also had a high level of crossover. This makes sense as drugs are a common treatment for almost any condition.

Precision and recall being reciprocal of each other is not a new phenomenon with respect to machine learning. Many algorithms struggle to find a balance between the two. In the context of our experiments, one potential reason for the trade-off is the use of determiners and pronouns. In the i2b2 dataset, many of the annotations include determiners and pronouns in the beginning or middle. Some examples are: "an outpatient holter monitor", "his chest x-ray", and "a few fine crackles at the left base". With the annotations generated by gazetteering through MIMIC, these pronouns and determiners are not selected. This could result in a case where training too much on the gazetteering data results in the inclusion of proper terms with the exclusion of pronouns and determiners.

CHAPTER 5

THRESHOLDING

In this section, we describe the experiments we conducted on thresholding and the reasoning behind them. The experiments are split into: 1) single-class experiments and 2) multi-class experiments, where multi-class attempts to label all entities within a single model. A full listing can be found in Table 7. The first set of experiments we performed take each individual type and attempt to perform training based on thresholded annotations for a given confidence score. Each experiment uses a set number of epochs. The multi-class experiment does the same as the single-class experiment except with all classes applied at the same time.

5.1 Methods

5.1.1 Preparatory Steps

MIMIC-III database was downloaded. Discharge summaries were extracted and run through pre-processing.

5.2 Experimental Details

Text for discharge summaries was extracted from the 'NoteEvents' table where 'Category' was set to 'Discharge summary'. Pre-processing started with an initial step of combining all de-identified terms into single terms that could be easily turned into features, including numbers, dates, and times. Punctuation was then modified to match the format that the i2b2 dataset was in.

To generate a random sampling of MIMIC data, all pre-processed sentences were

Table 7. Thresholding Experiments

Name	**Model Type**	**Description**
Problem Thresholding	Single-Class	Experiment determining the impact on precision and recall in comparison to baseline for the problem type when including thresholded pseudo-data in the training process.
Test Thresholding	Single-Class	Experiment determining the impact on precision and recall in comparison to baseline for the test type when including thresholded pseudo-data in the training process.
Treatment Thresholding	Single-Class	Experiment determining the impact on precision and recall in comparison to baseline for the treatment type when including thresholded pseudo-data in the training process.
Multi-Class Thresholding	Multi-Class	Experiment determining the impact on precision and recall in comparison to baseline for all types when including thresholded pseudo-data in the training process.

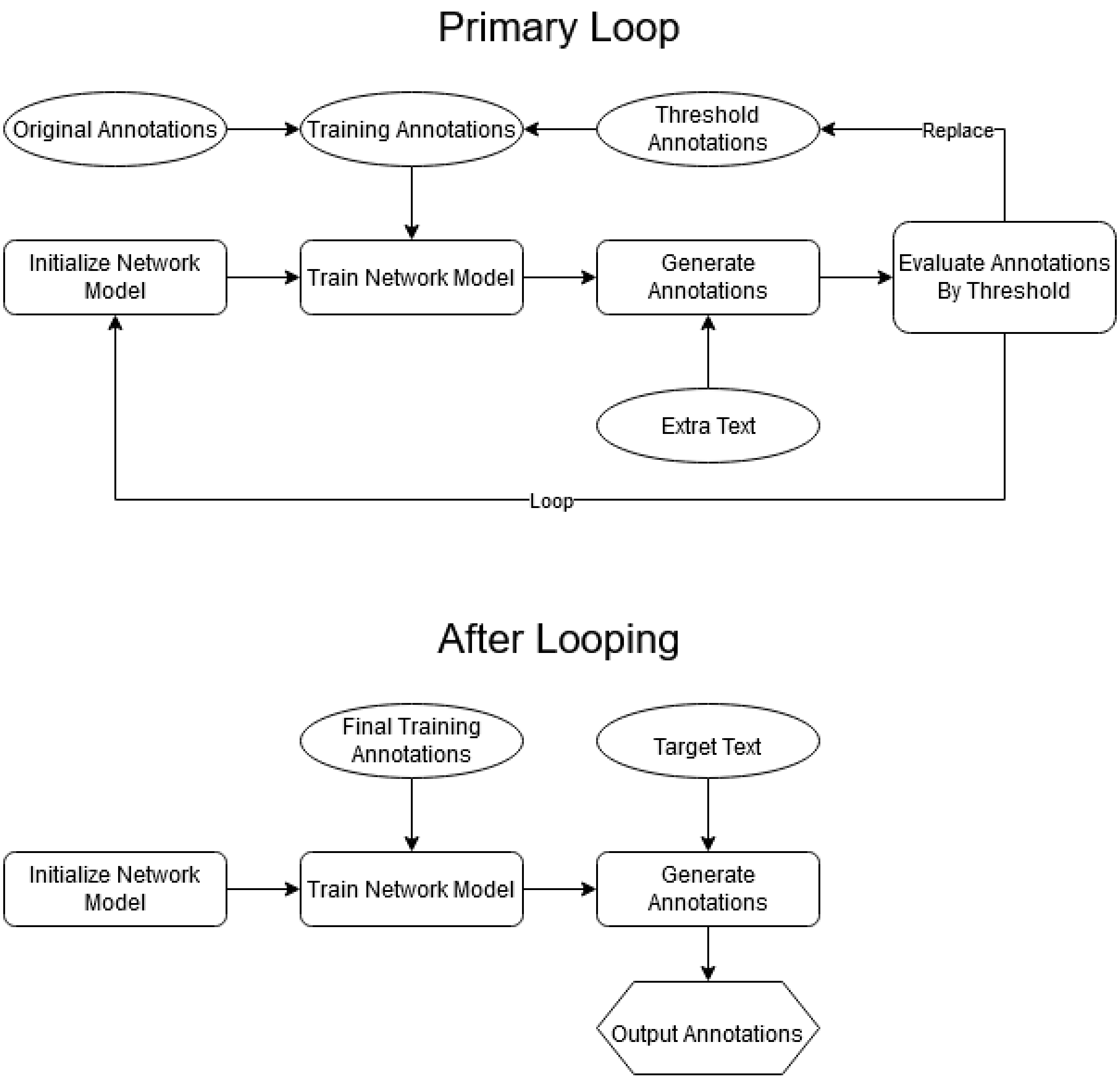


Fig. 19. Visual Pipeline Representing Thresholding.

loaded into memory. Sentences less than 8 tokens in length were removed to obtain data similar to i2b2. 400,000 sentences were then randomly selected and sent to files.

We began the thresholding process by loading the i2b2 vectors into memory and training a baseline network model. This model is then used to annotate the MIMIC-

III dataset. All annotations above a set confidence level are kept for the next cycle and added to a pool. We then train a new model on the i2b2 data and the pooled annotations from the previous iteration. We repeat this process until the percent difference between generated annotations is less than 5% or until 10 iterations have been run. Analysis is then performed on the final generated model. We completed this process for each annotation type in a single class model and in a combined multi-class model with all three labels. All training occured over 15 epochs. This number was chosen by training the network over a large amount of epochs numerous times and selecting the point where additional training produced negligible results.

5.3 Results and Discussion

5.3.1 Problem Type Thresholding

Figure 20 shows the token level precision, recall, and F_1 scores of the problem entity type when we utilized a thresholding method and trained a single-class model on both i2b2 annotations and pseudo-annotations from sampled MIMIC-III clinical notes after 10 epochs of thresholding. Starting from a confidence score of 0.5, the precision suffers a 5% loss from baseline, slowly rising as we increase the confidence score. As we reach 0.9 and 0.95, there is a very marginal (<=1%) increase on the precision over baseline. The recall starts below baseline at 0.5 confidence and almost matches at 0.6, then continues to decline up to 0.95 confidence.

5.3.2 Test Type Thresholding

Figure 21 shows the token level precision, recall, and F_1 scores of the test entity type when we utilized a thresholding method and trained a single-class model on both i2b2 annotations and pseudo-annotations from sampled MIMIC-III clinical notes after

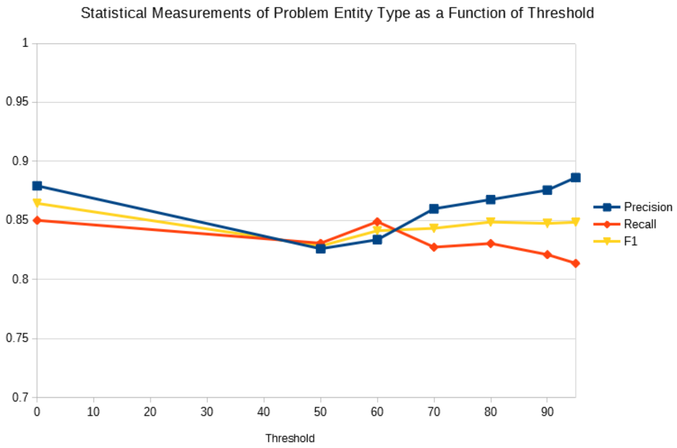


Fig. 20. Statistical Measurements of Problem Entity Type as a Function of Threshold

10 epochs of thresholding. Starting from a confidence score of 0.5, the precision suffers a severe decrease (13%) and slowly climbs back up to baseline as the confidence reaches 0.95. The recall at 0.5 confidence starts off about 2.5% above baseline before quickly returning to baseline as we raise the confidence.

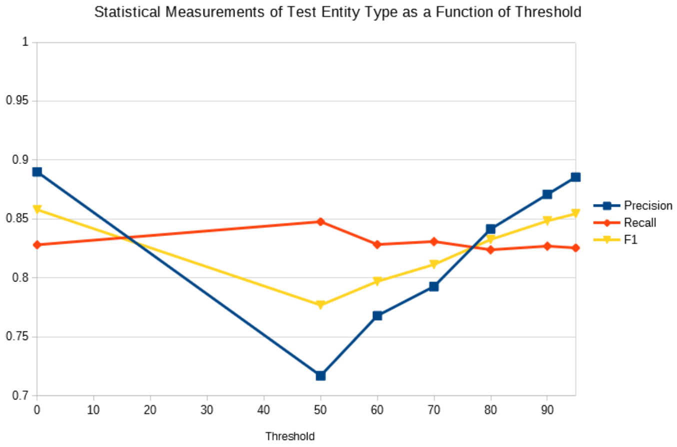


Fig. 21. Statistical Measurements of Test Entity Type as a Function of Threshold

5.3.3 Treatment Type Thresholding

Figure 22 shows the token level precision, recall, and F_1 scores of the treatment entity type when we utilized a thresholding method and trained a single-class model on both i2b2 annotations and pseudo-annotations from sampled MIMIC-III clinical notes after 10 epochs of thresholding. Starting from a confidence score of 0.5, the precision suffers a moderate decrease (8%) and slowly returns to slightly below baseline (3%) as the confidence reaches 0.95. The recall at 0.5 confidence starts off about 5% above baseline and maintains that position, even as the confidence reached 0.95.

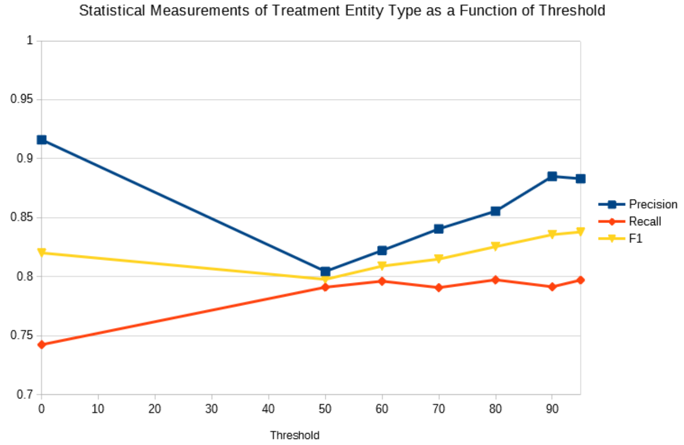


Fig. 22. Statistical Measurements of Treatment Entity Type as a Function of Threshold

5.3.4 Multi-Class Thresholding

Figure 23 shows the token level precision, recall, and F_1 scores of the problem, treatment, and test labels respectively when we utilized a thresholding method and trained a multi-class model on both i2b2 annotations and pseudo-annotations from sampled MIMIC-III clinical notes after 10 epochs of thresholding. The precision of the problem type decreases with the introduction of the pseudo-annotations (5%) though

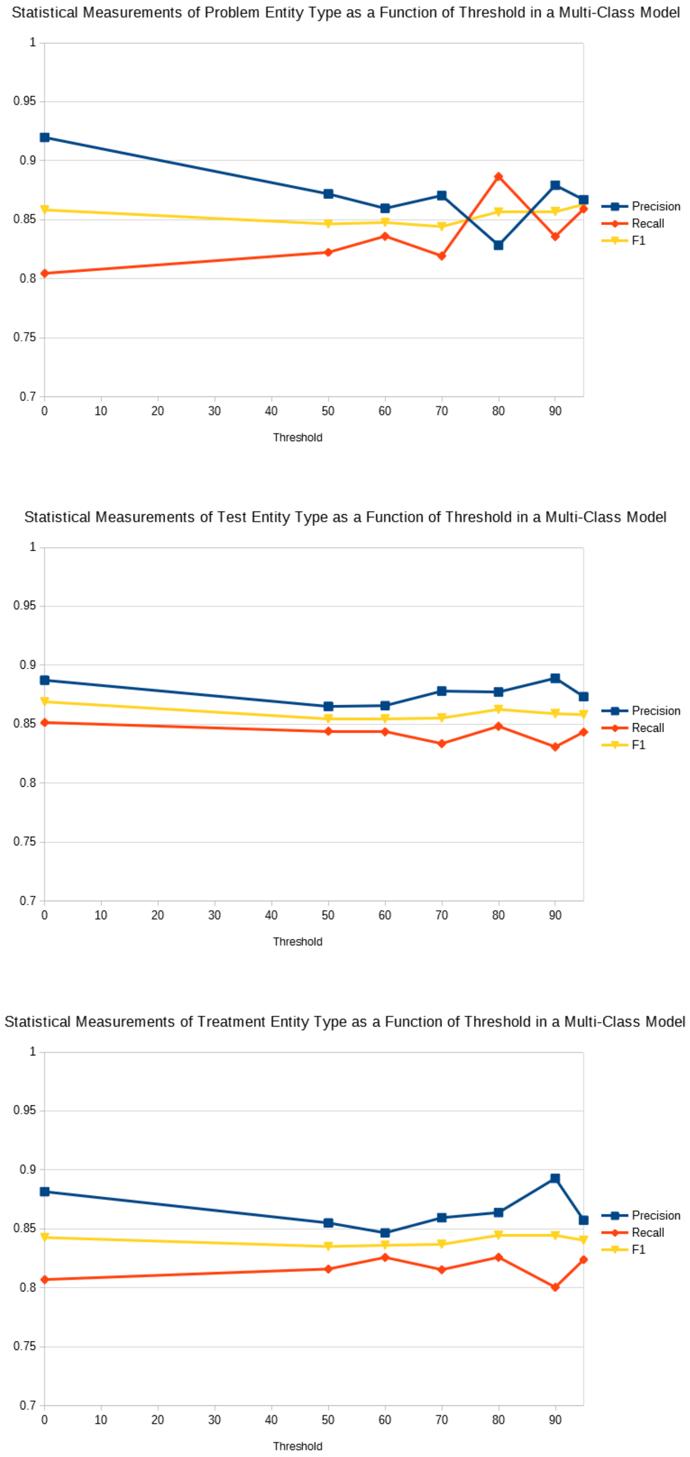


Fig. 23. Statistical Measurements of Problem (Top), Test (Middle) and Treatment (Bottom) Entity Type as a Function of Threshold in a Multi-Class Model

Table 8. Thresholding Multi-Class Phrase Strict (Lenient) F_1 Measurement

Source	**Class**	**Precision**	**Recall**	F_1
Baseline	Problem	**0.696**(**0.919**)	0.631(0.815)	**0.662**(**0.864**)
	Test	**0.793**(**0.902**)	**0.762**(**0.862**)	**0.777**(**0.881**)
	Treatment	0.706(0.882)	0.681(0.834)	0.693(0.857)
90 Confidence	Problem	0.672(0.888)	**0.644**(0.833)	0.658(0.859)
	Test	0.768(0.888)	0.732(0.839)	0.75(0.863)
	Treatment	**0.717**(**0.892**)	**0.684**(0.833)	**0.701**(**0.862**)
95 Confidence	Problem	0.632(0.858)	0.639(**0.839**)	0.635(0.848)
	Test	0.73(0.871)	0.729(0.858)	0.729(0.864)
	Treatment	0.663(0.851)	0.683(**0.85**)	0.673(0.851)

remains roughly at the same level as we increase the confidence. The recall continues to climb upward above baseline as we increase confidence. This is opposite to the observation had in the single-class model. The test precision and recall hover around baseline as confidence increases, never deviating more than 1.5% from baseline for both measurements. This differs greatly from the single-class model where precision declined greatly. The precision of the treatment type hovers around 2-3% lower than baseline while the recall hovers 1-2% above baseline. Like with the test type, treatment differs greatly from how it behaved in the single-class model.

5.3.5 Overall Results

When examining the single class graphs with respect to confidence, we get clear trend lines with some values oscillating around a fixed point at the most confidence points. The effects on precision are drastically higher than recall. What we get in most of the graphs is an initial large drop in precision in return for a boost in recall which reverses up to the 0.95 confidence point. Of the three types, only treatment achieved a noticeable increase in F_1 score over baseline in both token level and phrase level evaluations. The multi-class model exhibited similar results to the single-class model, though with more muted effects. The multi-class model failed to achieve notable results above baseline. Like the gazetteering experiments, the precision-recall trade-off can be observed in all of the thresholding experiments.

CHAPTER 6

CONCLUSIONS & FUTURE WORK

In this work, we explored using gazetteering and thresholding as psuedo-data generation techniques to improve performance in a deep neural network architecture. We showed that using gazetteers, there is a trade-off between precision and recall depending on the entity type. We also showed the same pattern with thresholding. However, the difference between the two was far more imbalanced with thresholding. With the technique used, we do not recommend thresholding at this time.

One potential limitation that we found at the end of our work was the coverage of our Word2Vec model. Only 86% of the terms in the training and test annotations were found in the model. This limitation could provide an explanation for the upper cap that the models demonstrated. Another limitation is the way that gazetteer annotations were generated compared to the original annotations. The training and test annotations contained determiners pronouns at preceding and mid positions in many cases whereas the gazetteer annotations did not. Regenerating the gazetteer annotations and including preceding and mid determiners and pronouns could potentially provide better results.

There are a number of follow-up studies that can be considered as well. To generate sentence structures similar to that of the training and test documentation, term swapping could be employed. This is where terms from gazetteer sources are swapped in place of original terms in training annotations and trained on. Generating new embedding models from different sources or merging multiple embeddings could also be considered for better coverage. Other sources for gazetteers could also be considered that better align with the language used by practitioners writing clinical notes.

The contributions of this thesis are:

1. Examining the effect of gazetteering on precision, recall, and F_1 score for clinical NER. Towards this end, we found:
   - Gazetteers that had a higher number of terms in the datasets had a higher impact. Medical lists used within a hospital system had a large number of unique terms but very little crossover; whereas public lists had a high level of crossover although less unique terms.
   - Gazetteer annotation can result in the inclusion of proper terms with the exclusion of preceding and middle determiners and pronouns that that are annotated within the data set.
2. Examining the effect of thresholding on precision, recall, and F_1 score for clinical NER. Towards this end, we found:
   - Clear trend lines across the thresholds with some values oscillating around a fixed point at the most confidence points.
   - The effects on precision are drastically higher than recall.
3. Examining the trade-off between precision and recall when either of these techniques are used. Towards this end, we found:
   - The measure that increases between precision and recall is based on the entity type.
   - Gazetteering had a more balanced trade-off when precision and recall changed. Thresholding had a more dramatic trade-off that resulted in

lower F_1 scores.

Appendix A

ABBREVIATIONS

BiLSTM	Bi-directional Long Short-Term Memory
CPT	Current Procedural Terminology
CRF	Conditional Random Field
EHR	Electronic Health Record
FDA	Food and Drug Administration
HMM	Hidden Markov Model
ICD	The International Statistical Classification of Diseases and Related Health Problems
IE	Information Extraction
LSTM	Long Short-Term Memory
ML	Machine Learning
NER	Named Entity Recognition
NLP	Natural Language Processing
NN	Neural Network
RNN	Recurrent Neural Network
RVA	Richmond Virginia
SMOTE	Synthetic Minority Oversampling Technique
SVM	Support Vector Machine
VCU	Virginia Commonwealth University

REFERENCES

[1] Antonio Toral et al. "Improving question answering using named entity recognition". In: *International Conference on Application of Natural Language to Information Systems*. Springer. 2005, pp. 181–191.

[2] Massimiliano Ciaramita and Yasemin Altun. "Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger". In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2006, pp. 594–602.

[3] Eiji Aramaki et al. "Text2table: Medical text summarization system based on named entity recognition and modality identification". In: *Proceedings of the BioNLP 2009 Workshop*. 2009, pp. 185–192.

[4] Erik F Sang and Fien De Meulder. "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition". In: *arXiv preprint cs/0306050* (2003).

[5] Ralph Grishman and Beth Sundheim. "Design of the MUC-6 evaluation". In: *Proceedings of the 6th conference on Message understanding*. Association for Computational Linguistics. 1995, pp. 1–11.

[6] Nancy Chinchor et al. "Named Entity Recognition Task Definition (version 1.4)". In: *Online at: http://www. nist. gov/speech/tests/ie-er/er_99/doc/ne99_taskdef_v1_4. pdf accessed* 6 (1999).

[7] Zhiheng Huang, Wei Xu, and Kai Yu. "Bidirectional LSTM-CRF models for sequence tagging". In: *arXiv preprint arXiv:1508.01991* (2015).

[8] Cicero Nogueira dos Santos and Victor Guimaraes. "Boosting named entity recognition with neural character embeddings". In: *arXiv preprint arXiv:1505.05008* (2015).

[9] Guergana K Savova et al. "Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications". In: *Journal of the American Medical Informatics Association* 17.5 (Sept. 2010), pp. 507–513. ISSN: 1067-5027. DOI: `10.1136/jamia.2009.001560`. eprint: `https://academic.oup.com/jamia/article-pdf/17/5/507/5940551/17-5-507.pdf`. URL: `https://doi.org/10.1136/jamia.2009.001560`.

[10] Mónica Marrero et al. "Named entity recognition: fallacies, challenges and opportunities". In: *Computer Standards & Interfaces* 35.5 (2013), pp. 482–489.

[11] Thierry Poibeau and Leila Kosseim. "Proper name extraction from non-journalistic texts". In: *Computational Linguistics in the Netherlands 2000*. Brill Rodopi, 2001, pp. 144–157.

[12] Carol Friedman and Stephen B Johnson. "Natural language and text processing in biomedicine". In: *Biomedical Informatics*. Springer, 2006, pp. 312–343.

[13] Gregory E Simon et al. "Assessing and Minimizing Re-identification Risk in Research Data Derived from Health Care Records". In: *eGEMs* 7.1 (2019).

[14] Clete A Kushida et al. "Strategies for de-identification and anonymization of electronic health record data for use in multicenter research studies". In: *Medical care* 50.Suppl (2012), S82.

[15] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[16] Xiaojin Zhu and Andrew B Goldberg. “Introduction to semi-supervised learning”. In: *Synthesis lectures on artificial intelligence and machine learning* 3.1 (2009), pp. 1–130.

[17] Mohamed Zaıt and Hammou Messatfa. “A comparative study of clustering methods”. In: *Future Generation Computer Systems* 13.2-3 (1997), pp. 149–159.

[18] John Spragins. “Learning without a teacher”. In: *IEEE transactions on information theory* 12.2 (1966), pp. 223–230.

[19] Bernard Widrow. “Bootstrap learning in threshold logic systems”. In: *Proc. Int. Federation Automatic Control* (1966), pp. 96–104.

[20] ER Ide and Cyril J Tunis. “An experimental investigation of a nonsupervised adaptive algorithm”. In: *IEEE Transactions on Electronic Computers* 6 (1967), pp. 860–864.

[21] ER Ide, CE Kiessling, and Cyril J Tunis. “Some conclusions on the use of adaptive linear decision functions”. In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part II.* ACM. 1968, pp. 1117–1124.

[22] WG Wee. “A survey of pattern recognition”. In: *Seventh Symposium on Adaptive Processes.* IEEE. 1968, pp. 25–25.

[23] Bernard Widrow, Narendra K Gupta, and Sidhartha Maitra. “Punish/reward: Learning with a critic in adaptive threshold systems”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 5 (1973), pp. 455–465.

[24] Simon Tong and Daphne Koller. “Support vector machine active learning with applications to text classification”. In: *Journal of machine learning research* 2.Nov (2001), pp. 45–66.

[25] Andreas Vlachos and Caroline Gasperin. “Bootstrapping and Evaluating Named Entity Recognition in the Biomedical Domain”. In: *Proceedings of the HLT-NAACL BioNLP Workshop on Linking Natural Language and Biology.* New York, New York: Association for Computational Linguistics, June 2006, pp. 138–145. URL: `https://www.aclweb.org/anthology/W06-3328`.

[26] Yoshimasa Tsuruoka and Jun’ichi Tsujii. “Boosting precision and recall of dictionary-based protein name recognition”. In: *Proceedings of the ACL 2003 workshop on Natural language processing in biomedicine-Volume 13.* Association for Computational Linguistics. 2003, pp. 41–48.

[27] Kai Xu et al. “Document-level attention-based BiLSTM-CRF incorporating disease dictionary for disease named entity recognition”. In: *Computers in biology and medicine* 108 (2019), pp. 122–132.

[28] David Nadeau and Satoshi Sekine. “A survey of named entity recognition and classification”. In: *Lingvisticae Investigationes* 30.1 (2007), pp. 3–26.

[29] Denys Proux et al. “Detecting gene symbols and names in biological texts”. In: *Genome Informatics* 9 (1998), pp. 72–80.

[30] Philip John Gorinski et al. “Named entity recognition for electronic health records: a comparison of rule-based and machine learning approaches”. In: *arXiv preprint arXiv:1903.03985* (2019).

[31] Leonard E Baum. “An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes”. In: ().

[32] Richard Tzong-Han Tsai et al. “NERBio: using selected word conjunctions, term normalization, and global patterns to improve biomedical named entity recognition”. In: *BMC bioinformatics.* Vol. 7. S5. Springer. 2006, S11.

[33] Xu Wang, Chen Yang, and Renchu Guan. “A comparative study for biomedical named entity recognition”. In: *International Journal of Machine Learning and Cybernetics* 9.3 (2018), pp. 373–382.

[34] Burr Settles. “Biomedical named entity recognition using conditional random fields and rich feature sets”. In: *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)*. 2004, pp. 107–110.

[35] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[36] Kurt Hornik. “Approximation capabilities of multilayer feedforward networks”. In: *Neural networks* 4.2 (1991), pp. 251–257.

[37] Xingyuan Pan and Vivek Srikumar. “Expressiveness of rectifier networks”. In: *International Conference on Machine Learning*. 2016, pp. 2427–2435.

[38] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

[39] Alex Graves. “Supervised sequence labelling”. In: *Supervised sequence labelling with recurrent neural networks*. Springer, 2012, pp. 5–13.

[40] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. “Bidirectional LSTM networks for improved phoneme classification and recognition”. In: *International Conference on Artificial Neural Networks*. Springer. 2005, pp. 799–804.

[41] Raghavendra Chalapathy, Ehsan Zare Borzeshi, and Massimo Piccardi. “Bidirectional LSTM-CRF for clinical concept extraction”. In: *arXiv preprint arXiv:1611.08373* (2016).

[42] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

[43] Gilles Bisson, Claire Nédellec, and Dolores Canamero. "Designing Clustering Methods for Ontology Building-The Mo'K Workbench." In: *ECAI workshop on ontology learning*. Vol. 31. Citeseer. 2000.

[44] Will Lowe. "Semantic representation and priming in a self-organizing lexicon". In: *4th Neural Computation and Psychology Workshop, London, 9–11 April 1997*. Springer. 1998, pp. 227–239.

[45] Holger Schwenk. "Continuous space language models". In: *Computer Speech & Language* 21.3 (2007), pp. 492–518.

[46] Cheng-Yuan Liou et al. "Autoencoder for words". In: *Neurocomputing* 139 (2014), pp. 84–96.

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

[48] Zhilin Yang et al. "Xlnet: Generalized autoregressive pretraining for language understanding". In: *Advances in neural information processing systems*. 2019, pp. 5753–5763.

[49] Matthew E Peters et al. "Deep contextualized word representations". In: *arXiv preprint arXiv:1802.05365* (2018).

[50] *spaCy · Industrial-strength Natural Language Processing in Python*. URL: `https://spacy.io/`.

[51] Martın Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: `https://www.tensorflow.org/`.

[52] Radim Řehůřek and Petr Sojka. “Software Framework for Topic Modelling with Large Corpora”. English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[53] Özlem Uzuner et al. “2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text”. In: *Journal of the American Medical Informatics Association* 18.5 (2011), pp. 552–556.

[54] Alistair EW Johnson et al. “MIMIC-III, a freely accessible critical care database”. In: *Scientific data* 3 (2016), p. 160035.

[55] *WebMD Medical Tests A-Z - Find information on medical tests from A to Z*. URL: https://www.webmd.com/a-to-z-guides/tests.

[56] *List of Surgical Procedures*. July 2014. URL: https://www.southerncross.co.nz/Portals/0/Society/EFulfillment/Product/List_of_Surgical_Procedures.pdf.