MENTION EXTRACTION AND NORMALIZATION USING ONTOLOGIES IN THE BIOMEDICAL DOMAIN

by

Mert Tiftikci

B.S., Computer Engineering, Boğaziçi University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2019

ACKNOWLEDGEMENTS

Firstly, I would like to thank my thesis advisor Assoc. Prof. Arzucan Özgür for her support and guidance throughout my master's education. I am very thankful that she believed in me and accepted me as her student. It has been a great honor for me to work with such a wonderful spirit while learning a great deal from her. I also would like to thank Assist. Prof. Junguk Hur and Assoc. Yongqun "Oliver" He for their support and contribution to my research. My sincere gratitude to Prof. Olcay Taner Yıldız and Assist. Prof. Emre Uğur for kindly accepting to participate in my thesis jury.

I have been fortunate to have an exceptionally supportive family and my dearest friends. It would be pretty hard to keep my sanity if it was not for my parents' endless and unconditional emotional support. I am sincerely grateful to Hakime Öztürk, Arda Çelebi, Göksu Öztürk, and all of the TABI Lab members for the academic discussions and small-talks.

I would like to express my gratitude to Mert İmre, Serkan Buğur, Nadin Kökciyan, Berkant Tarık Kepez, and Can Kurtan as well as all other friends and colleagues from CoLoRs and Artificial Intelligence laboratories.

ABSTRACT

MENTION EXTRACTION AND NORMALIZATION USING ONTOLOGIES IN THE BIOMEDICAL DOMAIN

This thesis proposes a machine learning- and rule-based system for the identification of adverse drug reaction (ADR) entity mentions in the text of drug labels and their normalization through the MedDRA dictionary. The machine learning approach is based on a recently proposed deep learning model that works on the sentence level. The model makes use of the combination of the pre-trained word embeddings and Convolutional Neural Network (CNN) embeddings generated from the characters of a given token. These tokens are initially passed through bi-directional Long Short-Term Memory (Bi-LSTM) layers for feature extraction. Finally, a Conditional Random Fields (CRF) classifier is trained on those extracted features for the prediction of the target mentions. The rule-based approach, used for normalizing the identified ADR mentions to MedDRA terms, is based on an extension of the text-mining system called SciMiner. The proposed system is evaluated with the TAC-ADR 2017 challenge dataset. Since this dataset contains mentions that are disjoint and overlapping, the model also uses a recently proposed chunking scheme designed to handle those types. The model obtained 76.97 f-score performance on the TAC dataset. Some of the challenges for the worse performance compared to performance of the models trained on the generic newspaper text are the small size of the training dataset and the uneven distribution of the class instances.

ÖZET

BİYOMEDİKAL ALANDA VARLIK İSMİ TANIMA VE ONTOLOJİLERİ KULLANARAK NORMALİZE ETME

Bu tezde ilaç prospektüslerinde bulunan ilaç yan etkilerini gösteren varlık isimlerinin tanınarak MedDRA sözlüğü içerisindeki kavramlara normalize etmeyi sağlayan kural ve makine öğrenmesi tabanlı bir sistem önerilmektedir. Makine öğrenmesi yaklaşımı, yakın zamanda önerilen ve cümle seviyesinde çalışan bir derin öğrenme modelini temel almaktadır. Model önceden öğrenilmiş kelime temsilleri ve kelime karakterlerinden üretilmiş evrişimli sinir ağları temsillerinin birleşiminden oluşan temsillerden faydalanır. Üretilen temsiller özniteliklerinin çıkarılması için ilk olarak uzun kısa-süreli bellek katmanından geçirilir. Son olarak, çıkarılan öznitelikleri kullanarak hedeflenen varlık isimlerini tahmin etmek üzere Şartlı Rastgele Alanlar eğitilir. Tanımlanmış ilaç yan etkilerini MedDRA sözlüğü kavramlarına normalize eden kural tabanlı yaklaşım, SciMiner isimli bir metin madenciliği sisteminin uzantısından temellenmiştir. Önerilen sistem, TAC-ADR 2017 yarışmasının veri kümesi ile değerlendirilmiştir. Bu veri kümesi ayrık ve üst üste binen varlık isimlerine sahip olduğu için, model yakın zamanda önerilen ve bu tip varlık isimlerini tanıyabilmek için tasarlanmış öbek şemasından da faydalanmaktadır. Model TAC veri kümesi üzerinde 76,97 f-skor elde etmiştir. Modelin genel gazete yazıları üzerinde eğitilmiş modeller kadar başarılı olmamasına sebepleri arasında veri kümesinin küçük olması ve sınıf örneklerinin eşit dağılmaması yer alır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

W	Weight matrix of a network layer
$\odot$	Hadamart Product
η	Learning rate
$\sigma()$	Sigmoid function
$\alpha_t()$	Forward algorithm, of Forward-Backward algorithm, result at time t
$\beta_t()$	Backward algorithm, of Forward-Backward algorithm, result at time t
$\delta_t()$	Forward algorithm, of Viterbi algorithm, result at time t
Θ	Parameters to be learned for a given model
$\psi_t()$	Factor function for time t
$\ell()$	Conditional log-likelihood

LIST OF ACRONYMS/ABBREVIATIONS

IE	Information Extraction
NLP	Natural Language Processing
NER	Named Entity Recognition
ADR	Adverse Drug Reaction
MUC	Message Understanding Conference
CoNLL	Conference on Computational Natural Language Learning
ML	Machine Learning
ANN	Artificial Neural Network
LSTM	Long Short-Term Memory
Bi-LSTM	Bi-directional LSTM
CNN	Convolutional Neural Network
CRF	Conditional Random Fields
NAdam	Nesterov Adaptive Moment Estimation
SGD	Stochastic Gradient Descent
NAG	Nesterov's accelerated gradient
BPTT	Back Propagation Through Time
P	Precision
R	Recall
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
OOV	Out of Vocabulary
BIO	Beginning-Inside-Outside, tagging scheme
BIOHD	An extension of BIO tagging scheme
BOW	Bag-of-Words
CBOW	Continuous BOW
FDA	Food and Drug Administration

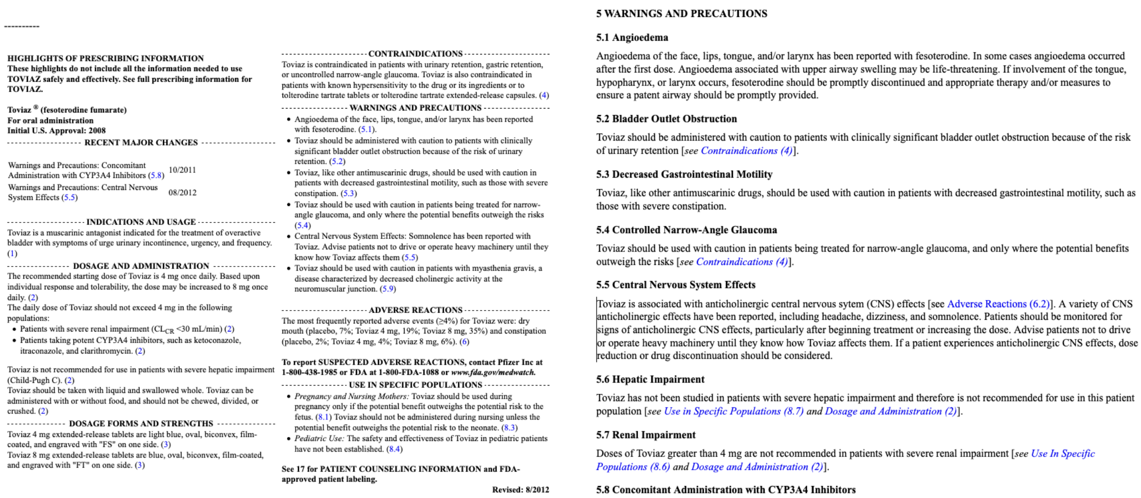
CDER	Center for Drug Evaluation and Research
FAERS	FDA Adverse Event Reporting System
MedWatch	Safety Information and Adverse Event Reporting Program
SPL	Structured Product Labels
MEDLINE	Medical Literature Analysis and Retrieval System Online
PubMed	Public/Publisher MEDLINE
MedDRA	Medical Dictionary for Regulatory Activities
SOC	System Organ Class
HLGT	High Level Group Term
HLT	High Level Term
PT	Preferred Term
LLT	Lowest Level Term
NLTK	Natural Language Toolkit
TAC	Text Analysis Conference
XML	Extensible Markup Language
CPU	Central Processing Unit

1. INTRODUCTION

Pharmacovigilance is defined as "the science and activities relating to the detection, assessment, understanding, and prevention of adverse effects or any other drug problem" [1]. It is impossible to know all possible adverse events of a particular drug since generalizability of the clinical trials is low, sample sizes are small, and the duration of clinical trials is short. The Center for Drug Evaluation and Research (CDER) is a division of the U.S. Food and Drug Administration (FDA) which is responsible for monitoring over-the-counter and prescription drugs, including biological therapeutics and generic drugs. CDER utilizes analysis of the spontaneous adverse event reports submitted to the FDA Adverse Event Reporting System (FAERS), an extensive database supporting pharmacovigilance, to detect adverse events. FAERS includes mandatory reports from pharmaceutical companies and reports that have been submitted to Safety Information and Adverse Event Reporting Program (MedWatch) directly, which is another related system that is dealing with adverse events and sentinel events.

Adverse events can be defined as undesired medical events that can be associated with the use of a drug without caring for its direct relation to the drug itself. Unlike adverse events, sentinel events can cause severe physical or mental disorders, even deaths. Adverse reactions are undesired medical events that can be linked to the use of a specific drug because of unpredictable pharmacological action caused by the drug. The serious adverse reactions are sentinel events since they can result in the death of the patient. Adverse Drug Reactions (ADR) are still in the top 10 leading causes of death and cost approximately $75 billion annually in the United States [2].

Since 2005, it is obligated to submit labels in Structured Product Label (SPL) to the CDER. SPL is a standard for exchanging product and facility information of drug labels developed by Health Level Seven International. These documents are in XML format which includes both label content, all text, tables, and figures, and also additional machine-readable drug listing data elements. A portion of an SPL is given in Figure 1.1. The current approach for FAERS case report review requires manual



HIGHLIGHTS OF PRESCRIBING INFORMATION
These highlights do not include all the information needed to use TOVIAZ safely and effectively. See full prescribing information for TOVIAZ.

Toviaz ® (fesoterodine fumarate)
For oral administration
Initial U.S. Approval: 2008

RECENT MAJOR CHANGES

Warnings and Precautions: Concomitant Administration with CYP3A4 Inhibitors (5.8)	10/2011
Warnings and Precautions: Central Nervous System Effects (5.5)	08/2012

INDICATIONS AND USAGE
Toviaz is a muscarinic antagonist indicated for the treatment of overactive bladder with symptoms of urge urinary incontinence, urgency, and frequency. (1)

DOSAGE AND ADMINISTRATION
The recommended starting dose of Toviaz is 4 mg once daily. Based upon individual response and tolerability, the dose may be increased to 8 mg once daily. (2)
The daily dose of Toviaz should not exceed 4 mg in the following populations:
- Patients with severe renal impairment (CL_{CR} <30 mL/min) (2)
- Patients taking potent CYP3A4 inhibitors, such as ketoconazole, itraconazole, and clarithromycin. (2)

Toviaz is not recommended for use in patients with severe hepatic impairment (Child-Pugh C). (2)
Toviaz should be taken with liquid and swallowed whole. Toviaz can be administered with or without food, and should not be chewed, divided, or crushed. (2)

DOSAGE FORMS AND STRENGTHS
Toviaz 4 mg extended-release tablets are light blue, oval, biconvex, film-coated, and engraved with "FS" on one side. (3)
Toviaz 8 mg extended-release tablets are blue, oval, biconvex, film-coated, and engraved with "FT" on one side. (3)

CONTRAINDICATIONS
Toviaz is contraindicated in patients with urinary retention, gastric retention, or uncontrolled narrow-angle glaucoma. Toviaz is also contraindicated in patients with known hypersensitivity to the drug or its ingredients or to tolterodine tartrate tablets or tolterodine tartrate extended-release capsules. (4)

WARNINGS AND PRECAUTIONS
- Angioedema of the face, lips, tongue, and/or larynx has been reported with fesoterodine. (5.1).
- Toviaz should be administered with caution to patients with clinically significant bladder outlet obstruction because of the risk of urinary retention. (5.2)
- Toviaz, like other antimuscarinic drugs, should be used with caution in patients with decreased gastrointestinal motility, such as those with severe constipation. (5.3)
- Toviaz should be used with caution in patients being treated for narrow-angle glaucoma, and only where the potential benefits outweigh the risks (5.4)
- Central Nervous System Effects: Somnolence has been reported with Toviaz. Advise patients not to drive or operate heavy machinery until they know how Toviaz affects them (5.5)
- Toviaz should be used with caution in patients with myasthenia gravis, a disease characterized by decreased cholinergic activity at the neuromuscular junction. (5.9)

ADVERSE REACTIONS
The most frequently reported adverse events (≥4%) for Toviaz were: dry mouth (placebo, 7%; Toviaz 4 mg, 19%; Toviaz 8 mg, 35%) and constipation (placebo, 2%; Toviaz 4 mg, 4%; Toviaz 8 mg, 6%). (6)

To report SUSPECTED ADVERSE REACTIONS, contact Pfizer Inc at 1-800-438-1985 or FDA at 1-800-FDA-1088 or *www.fda.gov/medwatch.*

USE IN SPECIFIC POPULATIONS
- *Pregnancy and Nursing Mothers:* Toviaz should be used during pregnancy only if the potential benefit outweighs the potential risk to the fetus. (8.1) Toviaz should not be administered during nursing unless the potential benefit outweighs the potential risk to the neonate. (8.3)
- *Pediatric Use:* The safety and effectiveness of Toviaz in pediatric patients have not been established. (8.4)

See 17 for PATIENT COUNSELING INFORMATION and FDA-approved patient labeling.

Revised: 8/2012

(a) Highlights

5 WARNINGS AND PRECAUTIONS

5.1 Angioedema

Angioedema of the face, lips, tongue, and/or larynx has been reported with fesoterodine. In some cases angioedema occurred after the first dose. Angioedema associated with upper airway swelling may be life-threatening. If involvement of the tongue, hypopharynx, or larynx occurs, fesoterodine should be promptly discontinued and appropriate therapy and/or measures to ensure a patent airway should be promptly provided.

5.2 Bladder Outlet Obstruction

Toviaz should be administered with caution to patients with clinically significant bladder outlet obstruction because of the risk of urinary retention [*see Contraindications (4)*].

5.3 Decreased Gastrointestinal Motility

Toviaz, like other antimuscarinic drugs, should be used with caution in patients with decreased gastrointestinal motility, such as those with severe constipation.

5.4 Controlled Narrow-Angle Glaucoma

Toviaz should be used with caution in patients being treated for narrow-angle glaucoma, and only where the potential benefits outweigh the risks [*see Contraindications (4)*].

5.5 Central Nervous System Effects

Toviaz is associated with anticholinergic central nervous sytem (CNS) effects [see Adverse Reactions (6.2)]. A variety of CNS anticholinergic effects have been reported, including headache, dizziness, and somnolence. Patients should be monitored for signs of anticholinergic CNS effects, particularly after beginning treatment or increasing the dose. Advise patients not to drive or operate heavy machinery until they know how Toviaz affects them. If a patient experiences anticholinergic CNS effects, dose reduction or drug discontinuation should be considered.

5.6 Hepatic Impairment

Toviaz has not been studied in patients with severe hepatic impairment and therefore is not recommended for use in this patient population [*see Use in Specific Populations (8.7) and Dosage and Administration (2)*].

5.7 Renal Impairment

Doses of Toviaz greater than 4 mg are not recommended in patients with severe renal impairment [*see Use In Specific Populations (8.6) and Dosage and Administration (2)*].

5.8 Concomitant Administration with CYP3A4 Inhibitors

(b) Warnings and Precautions

Figure 1.1. Excerpt from the Structured Product Label for drug Toviaz

reading of the text of an adverse event report in order to determine whether a candidate ADR has been reported before in the SPLs or not. The automation of the extraction of the ADRs from drug labels would increase the efficiency of this process.

The TAC-ADR 2017 challenge targeted the automatic extraction of ADR mentions from drug labels and normalization of them through the Medical Dictionary for Regulatory Activities (MedDRA). A mention can be defined as a portion of a text that corresponds to a particular concept such as ADR. Normalization is the process of assigning well-defined identities to extracted mentions. For example, given the sentence "Exclusive of an uncommon, mild injection site reaction, no adverse reactions to 11C-choline have been reported." obtained from the drug label of choline, "injection site reaction" is an ADR mention that is mapped to the MedDRA preferred term with id "10022095". Detailed information on mentions and their types are given in the Section 4.1.1.

In this thesis, the integration of machine learning and dictionary/rule-based methods in identifying ADR terms from drug labels and normalizing them to MedDRA Preferred Terms (PT) is investigated. The best results were achieved by an integrated system that is based on a deep learning model for mention extraction and a dictionary/rule-based SciMiner method for the normalization of the extracted ADRs to MedDRA terms.

2. RELATED WORK

Recently, a series of workshops organized by the Text Analysis Conference (TAC) to encourage research in Information Extraction (IE) and Natural Language Processing (NLP) and related applications, included an exclusive track focused on ADR extraction from drug labels [3]. The focus IE is automatically obtaining structured information from the documents that are unstructured and semi-structured. One of the hardest targets for IE is human language texts which require methodologies from NLP to solve. Named Entity Recognition (NER) is one of the subtasks of IE, which aims to find and classify named entities from unstructured texts [4].

The term named entity comes from the Message Understanding Conference (MUC) [5], which is the first major event and the spark for the research on the subtask. Initially, named entities were defined as proper names such as organizations, persons, and locations, where the definition became broader over time. Elisions are one of the exceptions of the sixth MUC dataset (MUC6). For the example text "North and South America", "North America" and "South America" are the two entities that have to be detected as named entity. This example is close to the irregular entities because of the overlapping part, which will be explained in Section 4.1.1.1. Nested expressions are another type of exception in the MUC6 dataset. The example text "1:30 p.m. Chicago time" contains two entities where the first one is the whole text as a time entity and the second one is "Chicago" as a location entity.

The MUC6 dataset targeted only language English language. Building a language-independent NER system is pretty hard since every language has a different structure on top of unique vocabulary. The Conference on Computational Natural Language Learning (CoNLL) focuses on the NER task for different languages on 2002 [6] and 2003 conferences [7]. These datasets are still widely used for evaluating NER systems. The target entities have changed over time as the interest in NER increased in other disciplines with different tasks. An example from bioinformatics is the GENIA corpus [8] and tagger, which is designed to detect entities such as proteins, DNAs, RNAs, cell lines, and cell types.

After the recognition of named entities generally the extracted text phrases are classified. This task is called as entity linking or named entity normalization. The classes are directly related to the given task. Person, Location, and Organizations are the target classes for the MUC6 dataset, and the Miscellaneous class is added for the CoNLL dataset. The class count can be a lot greater like in the cross-linking to Wikipedia pages where the extracted entities have to be linked to the related Wikipedia pages. Given the example sentence "I am visiting friends in Chicago", the named entity "Chicago" should be normalized into the corresponding Wikipedia page [9].

Different sources of information has been used for detecting ADRs. In addition to medical reports [10], it has been proposed to use data from social media [11] for detecting ADRs, since users tend to discuss their sicknesses, treatments and prescribed drugs and their effects on social media platforms. These discussions are not only confined to social networks specifically dedicated to health-related issues, but they also exist in generic platforms which could all be used for multi-corpus training to increase the accuracy of text mining systems for ADR recognition [12]. The process of detecting ADRs can be automized in order to increase efficiency with numerous approaches as well. Preparing a lexicon [11] for detection of ADRs requires much manual work and also limits a system's effectiveness to the extent of the lexicon. Syntactic and semantic patterns have been used to remedy the shortcomings of lexicon-based approaches in [13]. Detailed information on ADR extraction with different techniques on various data sources is available in [14] and [15].

Using a comprehensive and well-structured dictionary is critical in literature mining-based applications. For adverse drug reactions, MedDRA terms [16], SNOMED CT [17], Ontology of Adverse Events [18], and Ontology of Drug Neuropathy Adverse Events [19] can be used. The most widely-used dictionary for supporting ADR reporting is MedDRA, which is a clinically validated standardized medical terminology dictionary (and thesaurus), consisting of five levels of hierarchy [16]. More information on MedDRA and its hierarchy will be given in the Section 3.4.

3. BACKGROUND

This chapter explains the components of the proposed system. There are multiple sections describing each one of them in detail.

3.1. Artificial Neural Networks

Artificial neural networks (ANN) are a group of algorithms designed to learn to perform tasks motivated by the vision of creating algorithms generally without programming any task-specific rules. Incorporating such algorithms into machines would enable us to teach them the solution of a given problem only by showing examples to them. Creating a machine that could solve a problem would be similar to teaching dog a new trick with the existence of such algorithms. Inspired by the power of learning of the animals, these algorithms try to imitate the biological neural networks and artificially create algorithms with the same capacity [20].

ANNs are composed of nodes, which are called artificial neurons, that are connected in such a way to create a specific computational graph which loosely models the actual brain which has many neurons that are connected by the synapses. Most recent and successful networks have multiple layers of nodes connected sequentially. If these connections among nodes do not form a cycle, the constructed neural network

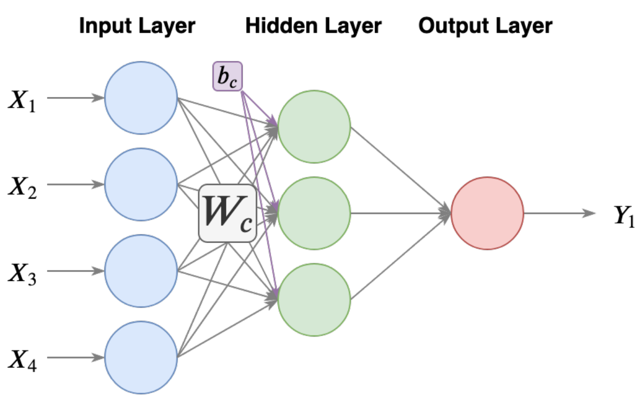


Figure 3.1. Example fully connected feedforward neural network

is called *feedforward*. A *fully connected layer* has all of its nodes connected to all of the nodes of the following layer. A single layer itself is a linear classifier which is only capable of solving linearly separable problems. A non-linear function can be applied after the layer to overcome this difficulty. These are called *activation functions* which are inspired by the frequency of action potentials of the biological neurons.

The process of finding the weights within the layers which would make the model generations best is called training, which is an optimization problem. When the calculation of the error is the difference between the prediction of a given model and expected result, this process is called *supervised learning*. The term difference here means the result of a selected *loss function*. One of the most famous methods for finding the best weights for a model is *backpropagation* [21]. It uses the stochastic gradient descent (SGD) method for parameters in each layer by exploiting the chain rule. This iterative algorithm updates the weights towards a minimum one step at a time. The model performs best at the global minimum, but getting there is not an easy task since gradients may point to a local minimum. *Learning rate* (η) decides the size of an update step (3.1), but convergence may take too long or never happen depending on the updates.

Deciding the update step in neural networks is crucial for efficiency and performance. An optimization problem is given in Figure 3.2 with the loss function surface is shown and the target state is marked with a star. In this example, SGD could not converge, whereas the other algorithms such as Adagrad [22] or Adadelta [23] could after the same amount of update steps. These algorithms manipulate the learning rate per parameter, while momentum and Nesterov's accelerated gradient (NAG) [24] do it globally.

$$\Delta w_i = -\eta \frac{\partial Error}{\partial w_i} \tag{3.1}$$

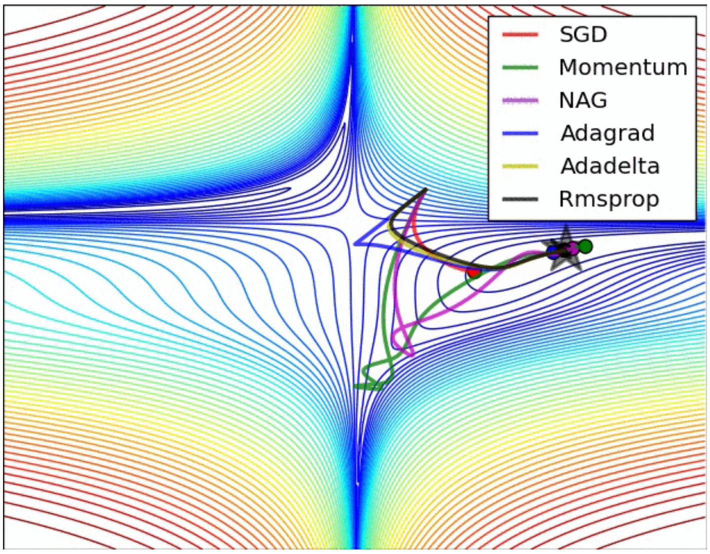
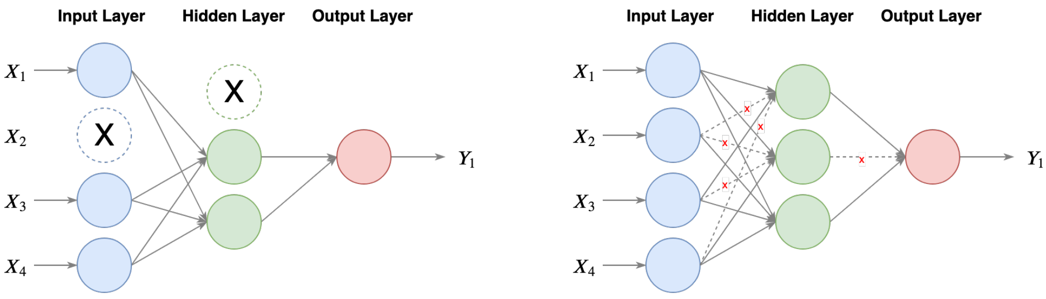


Figure 3.2. Effects of the optimization techniques [27]

In the real world, neural networks are designed to approximate any given function. Naturally, it is not possible to have all the labeled data, so training can only be applied on a small dataset. One of the major problems with training is overfitting. Overfitting happens when the model memorizes the dataset, instead of learning to solve a task. When overfitting happens, the model will give unreasonably high evaluation scores on the training set but not on the unseen data. There are numerous regularization techniques such as dropout [25] or dropconnect [26], to prevent overfitting.

Even though it is hard to construct a network, when done right, it is shown that ANNs are capable of solving many high-level tasks such as image recognition, translation, summarization.



(a) Dropout (b) DropConnect

Figure 3.3. Dropout and DropConnect regularization applied on Figure-3.1

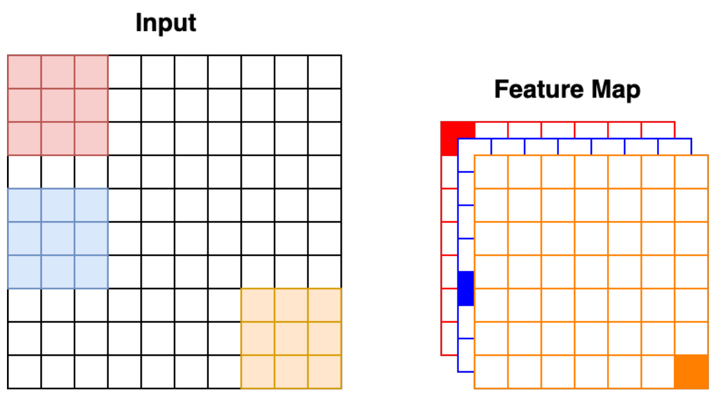


Figure 3.4. Convolutional neural network layer, feature maps and corresponding kernels are coded with color

3.1.1. Convolutional Neural Networks

Convolutional neural networks (CNN) are regularized version of multilayer perceptrons where layer are not fully connected to the input. The general structure and the design of the network has a resemblance with the organization of the visual cortex of an animal brain [28, 29]. A single layer of a CNN consists of multiple weight matrices called filters or kernels. Each kernel slides over the input as it calculates the dot product of the current position. This operation is usually followed with activation and then pooling function. Generated tensors are stacked together to form a feature map. Size of the feature map is determined by the kernel count, size of a kernel, size of the spatial movement and also size of the padding. An example layer with three kernels is given with color code in Figure 3.4.

The sliding operation allows CNN to use the same weights over the entire input. Comparing to fully connected layers, this reduces the required parameters which increase the computational efficiency of the model dramatically. Also, this enables CNN to learn smaller patterns over the entire input. The network becomes position invariant as a result of extracting this pattern regardless of its location.

3.1.2. Recurrent Neural Networks

One of the major weaknesses for the initial neural networks was the persistence in the evaluation of a sequence of data. For a classification problem, those networks would do the reasoning for each time step. The recurrent neural network (RNN) iterates over the data and combines the previous output with the current input in prediction. This feature allows it to incorporate contextual information along the direction of iteration. Capturing contextual information is especially beneficial for natural language processing (NLP) tasks such as named entity recognition (NER).

3.1.3. Long Short-Term Memory

Even though RNNs are theoretically capable of learning long-distance dependencies in the sequential data, it is hard to train them with gradient descent due to the problems of gradient vanishing and gradient explosion [30]. RNN weight updates are calculated by backpropagation through time (BPTT), which is the same with backpropagation but applies to the layer for every iteration. Derivation for calculating the gradients requires multiplication of the weights over and over again for the basic RNNs. This operation can diminish the value of the gradients so much that the gradient vanishes and update do not happen when gradients are smaller than 1. Also, gradient values can grow too large, gradient explosion, to make a proper update when gradients are larger than 1. The gradient explosion problem can be solved with gradient clipping [31] or gradient normalization [32]. The former technique works element-wise and involves clipping the gradient if it exceeds a designated threshold value while the latter technique rescales all the gradients if their norm exceeds the threshold.

Long short-term memory (LSTM) [33] solves the vanishing gradient problem by adding three more layers and cell state along with it. Layers of LSTM are usually called gates since they regulate information flow. Graphical representation of the formulation is given in Figure 3.5. Cell state, shown with c in Figure 3.5, passes through the entire chain and is only altered with linear operations. This changes BPTT formula in such a way that gradient values do not vanish.

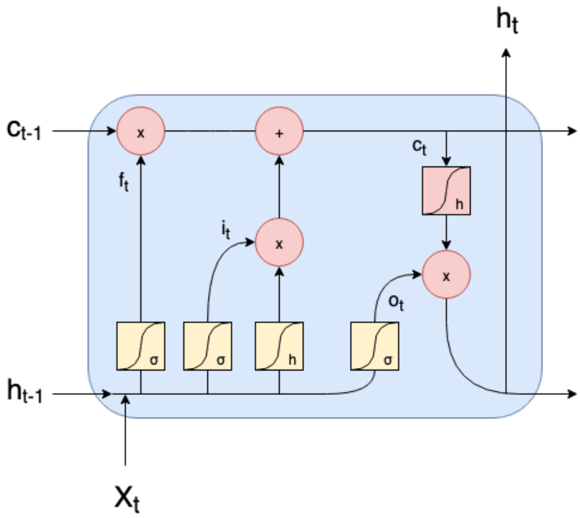


Figure 3.5. Visualization of the LSTM for single timestamp

All the gate values are calculated by the concatenation of the current input and the previous output of the cell, passed through a fully connected layer and followed by a sigmoid operation. The output vector contains values between 0 representing removing and 1 representing keeping.

$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \tag{3.2}$$

The forget gate determines how much of the accumulated information will be removed (3.2).

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \tag{3.3}$$

$$\tilde{c}_t = \tanh(W_c[h_{t-1}, X_t] + b_c) \tag{3.4}$$

Information extracted from the input calculated in (3.4) and the input gate determines how much of it should be incorporated into the cell state (3.3).

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{3.5}$$

New cell state is calculated using the input and the forget gate for the current timestamp (3.5). It is possible for the network to erase all the accumulated information or completely ignore the existing data. Finally, the new cell state is passed from the activation function and then the output gate (3.7).

$$o_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \tag{3.6}$$

$$h_t = f_t \odot \tanh(c_t) \tag{3.7}$$

3.2. Linear-Chain Conditional Random Fields

Conditional random fields (CRF) is a discriminative undirected probabilistic graphical model [34]. Probabilistic graphical models are probabilistic models that try to represent a complex distribution over many variables with a product of factors over subsets of variables. This approach makes it easier to decide factors since it is easier to understand the conditional independence relationships. The term "graphical model" comes from the fact that models can be represented by a graph pretty neatly. Edges in a probabilistic graphical model are drawn directed when factors are conditional.

Linear-chain conditional random fields, shown in Figure 3.6, are restricted in the choice of feature functions so that they work better on sequences. They are considered as discriminative since they model $p(y|x)$ conditional distribution (3.8) directly

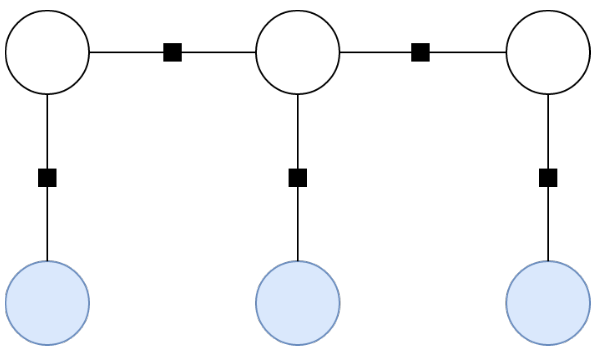


Figure 3.6. Graphical model of linear-chain conditional random fields

instead of modeling joint distributions as generative models do. For the feature vector $x = (x_1, x_1, \dots, x_k)$, the formulation of CRF is (3.8) where $Z(x)$ (3.10) is the input-dependent normalization function, and Ψ_t (3.9) is the local factor.

$$p(x|y) = \frac{1}{Z(x)} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, x_t) \tag{3.8}$$

$$\Psi_t(y_t, y_{t-1}, x_t) = exp\left\{\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right\} \tag{3.9}$$

$$Z(x) = \sum_{y} \prod_{t=1}^{T} exp\left\{\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\right\} \tag{3.10}$$

3.2.1. Forward-Backward and Viterbi Algorithms

In training, it is required to calculate the marginal distributions which can be calculated using the forward-backward algorithm. The marginal distributions (3.13) are computed after calculating forward (3.11) and backward (3.12) algorithms for the entire sequence.

$$\alpha_t(j) = \sum_{i \in S} \Psi_t(j, i, x_t)\alpha_{t-1}(i) \tag{3.11}$$

$$\beta_t(i) = \sum_{j \in S} \Psi_{t+1}(j, i, x_{t+1})\beta_{t+1}(j) \tag{3.12}$$

$$p(y_{t-1}, y_t|x) = \frac{1}{Z(x)}\alpha_{t-1}(y_{t-1})\Psi_t(y_t, y_{t-1}, x_t)\beta_t(y_t) \tag{3.13}$$

Maximizing the conditional log-likelihood (3.14) or minimizing the negative conditional log-likelihood will give us the optimum parameters for the model.

$$\ell(\theta) = \sum_{i=1}^{N} \log\, p(y^{(i)}|x^{(i)}; \theta) \tag{3.14}$$

The Viterbi algorithm (3.17) is used to find the most probable assignments. This algorithm has the same formulation with the forward-backward algorithm except summations are replaced with maximization. Also, all the forward recursion (3.15) should be applied before the predictions which are assigned at backward (3.16) recursion. Inferences of the algorithms and detailed information on conditional random fields can be found in [35].

$$\delta_t(j) = \max_{y \in S} \Psi_t(j, i, x_t)\delta_{t-1}(i) \tag{3.15}$$

$$y_T^* = \underset{i \in S}{\operatorname{argmax}}\ \delta_T(i) \tag{3.16}$$

$$y_T^* = \underset{i \in S}{\operatorname{argmax}}\ \Psi_{t+1}(y_{t+1}^*, i, x_{t+1})\delta_t(i) \quad \text{for } t < T \tag{3.17}$$

3.3. Natural Language Processing

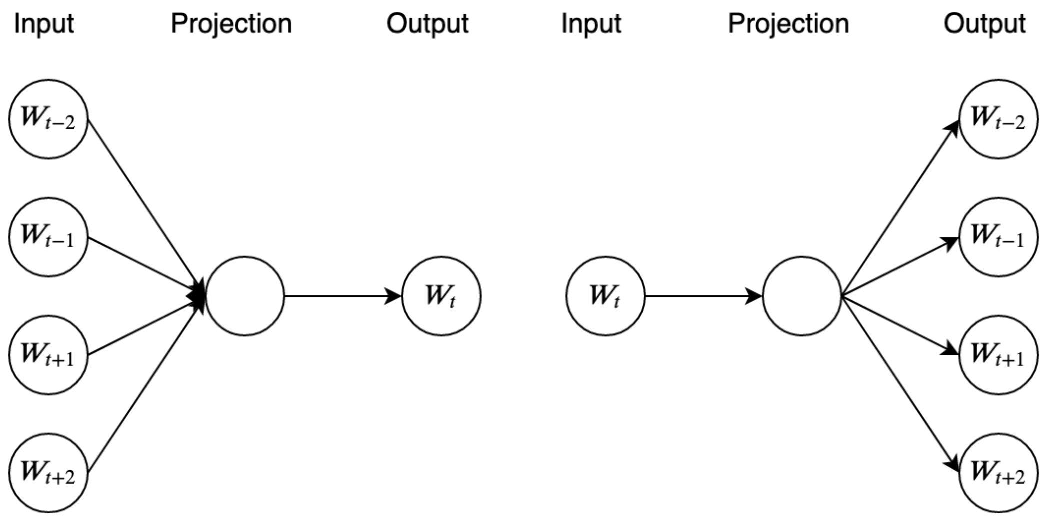
In this section, concepts about natural language processing (NLP), that are related to the model, will be explained.

3.3.1. Word Embeddings

Unlike humans, statistical models work on vectors instead of high dimensional raw data. Traditionally, unique words build up to dictionary called vocabulary which NLP systems work with. One of the techniques that is widely used for encoding data discretely is one-hot encoding. Distance between all one-hot encoded vector pairs is equal in terms of cosine similarity (3.18), which is broadly used as a measure of similarity among vectors.

$$similarity(A, B) = cos(\theta) = \frac{A \cdot B}{||A||||B||} \tag{3.18}$$

Being extremely sparse and having high dimensional space makes it undesirable to use this encoding in a real dataset. Also, words in natural language do have dependencies which cannot be encoded in one-hot representations. There is a substantial amount of effort to project these encodings into a low dimension space with real-valued vectors which have proper dependencies among them that reflect actual similarities between words.



(a) CBOW (b) Skip-gram

Figure 3.7. Word2Vec architectures

Word2Vec [36] is one of many methods for generating real-valued, distributed representations. It is a rather shallow network with two fully connected layers with two different model architectures. Continuous bag-of-words (CBOW), shown in Figure 3.7(a), tries to predict a word by looking at the words surrounding it. Bag-of-words (BOW) model treats text as a collection of words disregarding its grammatical structure. Continuous BOW works on the part of a given text with a fixed length, called window, instead of working on the entire text itself. Skip-gram model, shown in Figure 3.7(b), does the opposite; it takes a single word as an input to predict words around it for a given window.

As illustrated in Figure 3.8, resulting vectors are semantically meaningful. Countries and capitals are aligned along what could be called a context vector. Also, countries and capitals are distributed along these vectors in such a way that an approximate "capital of a country" relation vector can be seen.

3.3.2. Chunking

Chunking is a process of extracting phrases from unstructured text. Tasks like POS tagging and named entity recognition (NER) try to extract meaningful phrases

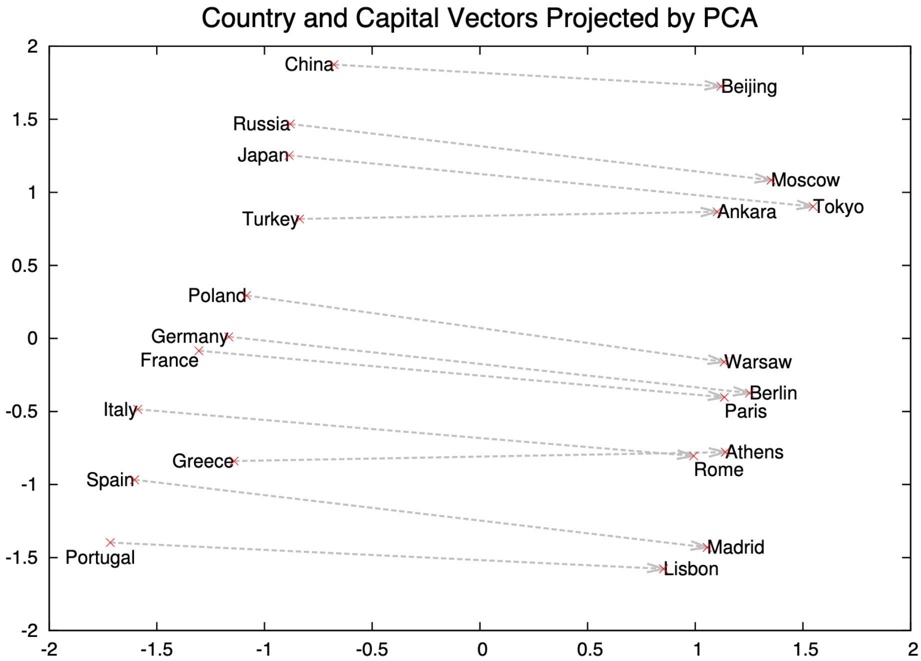


Figure 3.8. PCA projection of a selected country/capital Word2Vec embedding pairs [36].

instead of a collection of tokens separately. Representing data with the chunks with target labels is significant in terms of performance. Different types of chunking schemes change the distribution probabilities of the data [37]. The performance of a proposed model can change according to the selection of the chunking scheme.

One of the chunking schemes widely used in NLP tasks is BIO2 representation [38]. It uses three states with letters "B", "I", and "O" for a single type of target tag. Letter "O" used for the tokens that are not part of the target tag. Phrases that are part of the target tags are tagged with "B" in the begging and "I" with the rest of them. An example sentence tagged with this scheme is shown in Figure 3.9.

Figure 3.9. Chunking sentence example from TAC dataset *AdreView* drug label

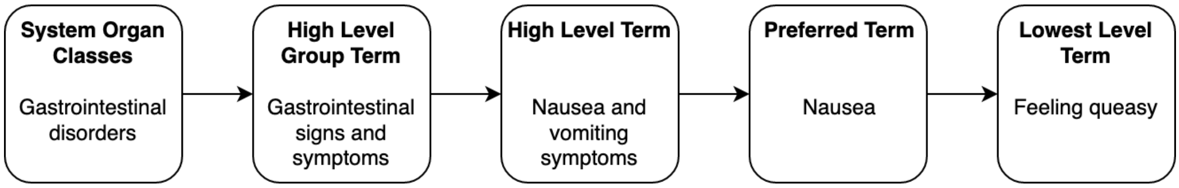


Figure 3.10. MedDRA Hierarchy

3.3.3. Part-of-Speech Tags

Part of speech (POS) is a class of words from a pre-defined set of tags which have similar grammatical properties. Tagging a word with a POS tag is not a trivial task since natural languages are known to be ambiguous. An example for ambiguity could be “The Duchess was *entertaining* last night.” [35] where *entertaining* could be both adjective and verb. There are many corpora such as Brown corpus and Penn Treebank [39] with unique tag sets for English. Also, there are many different methods for building a tagger such as hidden Markov models, support vector machines, and perceptron.

3.4. MedDRA Dictionary

The MedDRA dictionary organizes various ADRs using a five-level hierarchy. The bottom layer is Lowest Level Terms (LLT) at the bottom, followed by Preferred Terms (PT), High Level Terms (HLT), High Level Group Terms (HLGT), and System Organ Class (SOC). While individual ADR cases are usually coded for data entry at the most specific LLT level, the outputs of counts or cases are usually provided at the PT level. Hierarchy of the MedDRA dictionary is shown in Figure 3.10

4. MENTION EXTRACTION AND NORMALIZATION

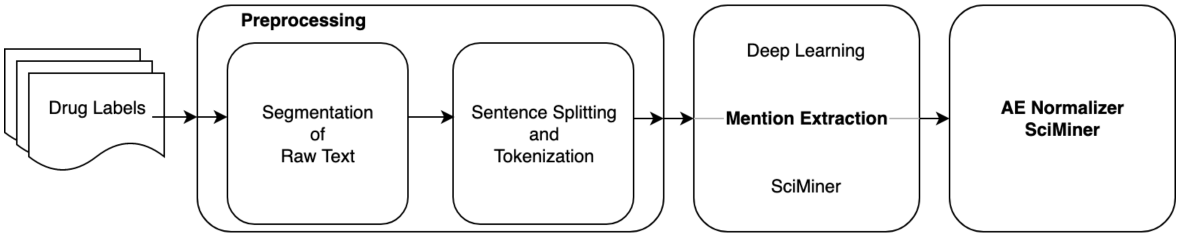


Figure 4.1. Mention extraction and normalization system pipeline

In this chapter, the system architecture and description of the dataset will be covered. A high-level description of the integrated deep learning and dictionary/rule-based approach for mention extraction and normalization is illustrated in Figure 4.1.

4.1. Dataset

As shown in Table 4.1, the TAC dataset included 200 manually curated labels (101 in the Training and 99 in the Test set). The details about the dataset have been published in [40]. The gold standard files have XML format with tags for sections, mentions, relations, and normalizations. The section tags contain the raw text of drug label sections "Adverse Reactions", "Warnings and Precautions", and "Boxed Warning". Mention tags contain the type and the location of the named entity to extract. One of the tasks of the challenge was extracting "negated", "hypothetical", and "effect" relations among named entities. Since this task is not within the scope of this study and relations are required for detecting *positive* ADRs, all the ADRs are considered for normalization instead. The definition for positive ADR is given in Section 4.1.2.

Briefly, four annotators, including two medical doctors, one medical librarian and one biomedical informatics researcher, participated in the manual annotation process of these 200 drug labels. These annotators were all trained for biomedical annotation and the drug labels were annotated independently by these annotators. Any disagreements were reconciled in pairs or collectively resolved by all four annotators.

Table 4.1. Label, section, and sentence counts in TAC dataset

	Training	Testing	Total
# Labels	101	99	200
# Sections	239	237	476
# Sentence	15.046	14.887	29.933

4.1.1. Mentions

There are six mention types in the TAC dataset. *The adverse reaction type* entities are used for annotation of the adverse drug reactions in the text. Adverse drug reactions are unwanted medical events that could be associated with the use of a drug in humans. In the process of curation of the TAC dataset, other types of mentions are only annotated when they are related to an ADR since ADR entities are the main focus of the challenge. This design choice is causing dataset class labels to be unbalanced as well as making it hard to learn the other dependent classes without modeling relations. First six rows of the table show the mention types.

Table 4.2. Annotation distribution in TAC dataset

Annotation	Training	Testing	Total
# AdverseReaction	13,795	12,693	26,488
# Animal	44	86	130
# DrugClass	249	164	413
# Factor	602	562	1,164
# Negation	98	173	271
# Severity	934	947	1,881
# Reactions	7,038	6,343	13,381
# MedDRA Mappings	5,882	5,185	13,501

Figure 4.2. A sentence from the label of the drug Toviaz with irregular entities

The drug class entity type is used in order to annotate the class of the drugs in the label. *The animal entity* type is used to annotate non-humans used during drug testing. Negations of the ADRs are annotated with *the negation entity* type. *The severity entity* type is used to annotate the severity of an ADR, both quantitatively and qualitatively; whereas *the factor entity* type annotates other aspects of ADRs. Reaction is used to tag unique ADR mentions for a given label. MedDRA Mappings shows the reactions that could be normalized to the MedDRA dictionary. The distribution of the mentions and normalizations in the TAC dataset are given in the Table 4.2.

4.1.1.1. Irregular Entities. Irregular entities also pose a challenge for entity recognition. Irregular entities can be discontinuous or overlapping. An entity is considered overlapping when part of it shared with another entity. Discontinuous entities span text portions that are disjoint as exemplified in Figure 4.2. The entity mentions are "Angioedema of the face", "Angioedema of the lips", "Angioedema of the tongue", and "Angioedema of the larynx". These are overlapping entities since the text portion "Angioedema of the" is shared in all four entity mentions. The last three entities are discontinuous since there are other words between the text portion "Angioedema of the" and the remaining parts of the entity mentions. For example, in the entity mention "Angioedema of the tongue", the words "face" and "lips" occur between the entity mention texts "Angioedema of the" and "tongue".

4.1.2. Normalizations

A positive ADR mention is defined as an ADR that is not negated, and that does not have a hypothetical relation to other DrugClass and Animal mentions. In the dataset, normalizations are given only for the positive ADRs. Normalization task targets the PTs, but LLTs could be provided as well. There can be more than one normalization for a single ADR. There is an extra attribute called flag, which is used at the following exceptional conditions:

- The flag value becomes “underspecified” when an ADR is too specific. These ADRs will be mapped to the closest PTs.
- The flag value becomes “unmapped” when a proper MedDRA mapping cannot be found.
- The flag value becomes “HLT” or “HLGT” when an ADR maps onto high-level term or high-level group term. IDs of such dictionary entries will be reported with attributes for PT IDs when this happens.

The TAC dataset uses MedDRA version 18.1 for the normalizations. An example normalization for the term “angioedema of the lips” in the sentence in Figure 4.2 is an LLT MedDRA term, which has an LLT ID “10054495” and PT ID of “10016029”.

4.2. Preprocessing

Preprocessing steps are applied to all the files in the dataset which are given as XML files. These files are parsed and prepared for the training before feeding into the model. The steps that are shown in Figure 4.3 are explained in this section. Initially, all XML files are parsed and separated into four categories named “TEXT”, “HEADING”, “TABLE”, and “TABLE-INTERNAL” in the first step. In the second step, NLTK [41] tokenizer is used to split the text into sentences and tokens.

The TAC dataset is a subset of a reduced and simplified version of structured product labels of FDA. This simplification process resulted in labels with flawed raw texts. Text normalization is designed mostly to remedy these flaws hence to reduce the out of vocabulary (OOV) count in the dataset, but it only applies when the vocabulary of word embeddings are provided. The text normalization steps are applied to every token, and when one of them succeeds the rest of the steps are not applied.

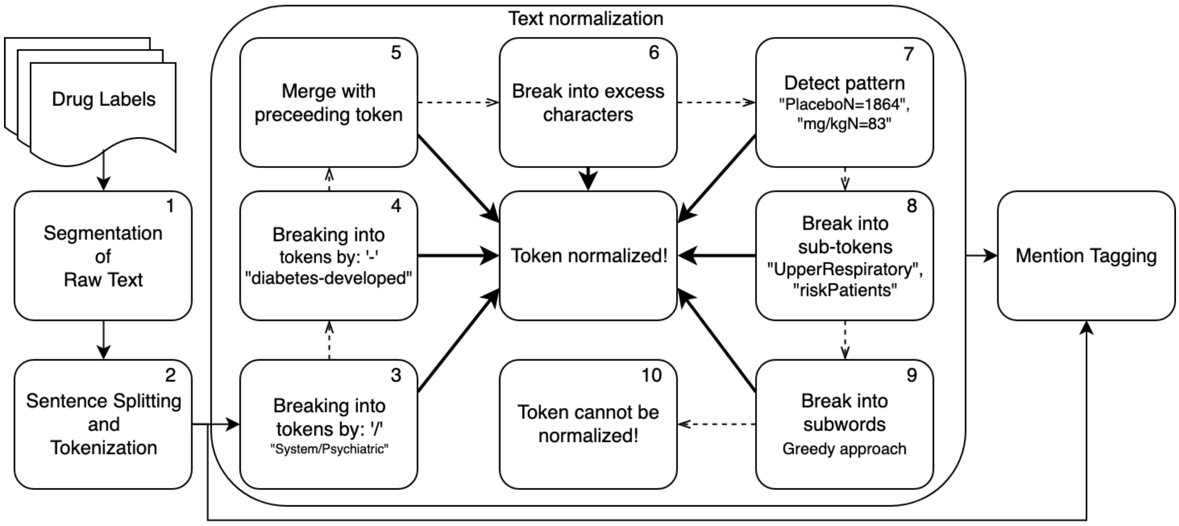


Figure 4.3. Preprocessings pipeline - Dashed arrows in the text normalization are selected when a step fails whereas thick arrows selected when success.

The third step is to separate tokens like “System/Psychiatric” into three tokens. The main reason for this comes from the labeling approach. Some of the mentions in the dataset were a piece of such compound tokens. Since the model used in this study makes predictions on the token level, separating such compound tokens was the only option to support these mentions. The following step does a similar job to this one, but the binding character here is a hyphen. The reason for the forth step to be given separately is this step will not break a given token if it exists in the given vocabulary.

The fifth step is there for catching simplification flaws such as the ones that are in the given sentence: “D rug R eaction with E osinophilia and S ystemic S ymptoms.” The next step detects the tokens containing a preceding or succeeding repeated character c where $c \in \{* \text{ , - , + , . , \^{} , /}\}$ such as “Abnormal+” or “**thyme”. The seventh step captures the table related problems where texts in the columns are merged.

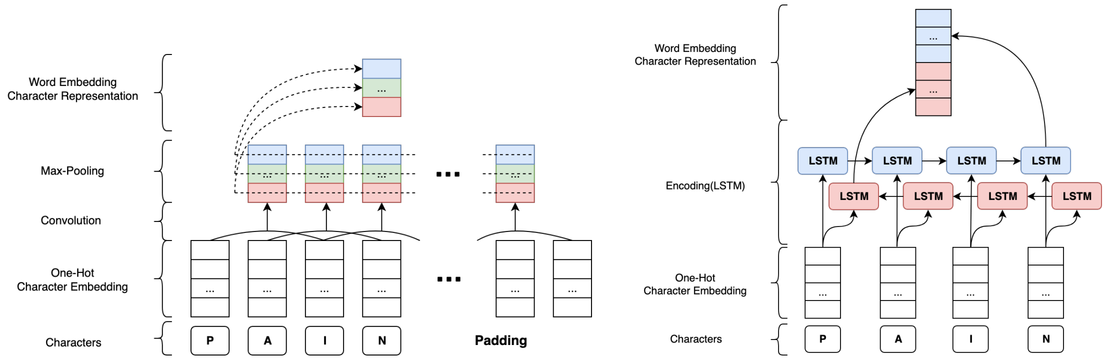
The eighth and ninth steps are mostly for detecting the merged tokens. The eighth step separate merged heading, which is pretty straightforward, such as “Upper-Respiratory” or “ClassPreferred”. The ninth step tries to separate very long sequences

Table 4.3. Comparison of BIO2 and BIOHD chunking representation for the example in Figure 4.2

	Angioedema	of	the	face	,	tongue	,	and	/	or	larynx
BIO2	B-ADR	I-ADR	I-ADR	I-ADR	0	I-ADR	0	0	0	0	I-ADR
BIOHD	HB-ADR	HI-ADR	HI-ADR	DB-ADR	0	DB-ADR	0	0	0	0	DB-ADR

that are the concatenation of mostly medical terms within table columns. These tokens can take up to the length of 140 characters and contain domain-specific words which are hard to distinguish. This step is the most complicated one in the normalization and is tailored specifically for the TAC dataset. It will not work if the given token is all upper case or contains any character other than letters in the alphabet. If the conditions are met, it will iteratively break down the token by trying to find the longest sequence within the token that exists in the vocabulary. This is a greedy approach except for a check that has to be satisfied in every iteration. If the token with one character shorter than the candidate token is also in the vocabulary as well as the rest of the sequence which should have more than three characters, the shorter token will be selected instead of the candidate. In order to clarify this rule, let us assume we have the sequence “livesale” in a text with the context of commerce. The iterative algorithm will generate the tuple “lives” and “ale”, but the actual tokens are “live” and “sale”. It does not accept a solution which has more than two tokens with length lower than four. This rule is to eliminate solutions that could be generated by meaninglessly small tokens because of one or more of the actual tokens were not in the vocabulary.

The final step of preprocessing is tagging the tokens corresponding to the mention types and chunking scheme. Two types of chunking schemes are used in this study. One of which is an adaptation of the BIO2 chunking scheme [38]. Regular entities are tagged as usual. Irregular entities are treated as if they were regular, which means all non-head disjoint segments are tagged with “I” as shown in the Table 4.3. BIOHD is the second available chunking scheme [42]. This scheme can be seen as an extension to the previous one since it contains all the identifiers of that scheme. There will be



(a) CNN Extractor (b) LSTM Extractor

Figure 4.4. Character Embeddigs

no difference between datasets tagged with these schemes if they do not contain any irregular entities. One of the additional identifiers is “H”, which is used in conjunction with “I” to tag the shared part of overlapping entity segments, but only for the shared tokens. The start of the shared part is tagged with “H”, and the rest of the shared part is tagged with “HI”. The last additional identifier is “D” used to tag the disjoint entity segment tokens if they are not shared. The beginning of the disjoint entity segment token is tagged with “D” while following tokens are tagged with “DI”.

4.3. Combined Word Embeddings

Every token in a given sentence is transformed into a vector before being fed into the model. The two main components of this vector are character embeddings and word embeddings. Additional features about a given token can also be incorporated into the vector.

4.3.1. Word Embeddings

Word embeddings are real-valued vectors that are generated by algorithms like Word2Vec [36] or GloVe [43]. Generation of these vectors usually requires big corpora, and the generated vectors are expected to have latent semantic meanings for each word. The quality of the embeddings is highly dependent on the corpus and the training strategy. In this study, embeddings trained on the PubMed that are the output of the

Word2Vec model with parameters tuned for NER type tasks are used. Three different embedding sets, trained with window sizes of 5, 2, and 30, have been tested with the proposed system. All three sets have the embedding dimension of 200. Embeddings generated with the model with window size 5 are explained in [44]. The ones generated with the window size 2 and 30 are created with a more robust approach detailed in [45].

4.3.2. Character Embeddings

These representations are expected to reflect the morphological features of the tokens [46]. Writing a word in all-caps or all-lower cased or the only first letter upper case could be the difference between being emphasized or not, or if it is at the beginning of a sentence or not. Different approaches such as CNN or LSTM have been used for extracting character level features [46, 47].

4.3.3. Additional Feature Embeddings

Different information related to the text can be incorporated into the combined word embeddings to enrich the representation. Lower-cased and normalized versions of the tokens are also searched in the embeddings vocabulary to reduce the OOV count. Normalization here is mapping the tokens that are not in the vocabulary in such a way that the normalized token would be semantically more meaningful for the given token. Currently, normalization detects links, phones, fractions, numbers, and equations with regex rules. With *token normalization*, any of the supported patterns can be mapped to a trainable embedding which would increase the consistency and lower the OOV count. This token normalization process has not been explained in Section 4.2 or shown in Figure 4.3, because it has not been designed to make corrections, and it is not part of the text normalization cycle.

One of the additional features is the one-hot encoded case embeddings [48] which represent casing information of the actual token, so that information lost in the token normalization is minimized. Another feature is the POS embeddings of the tokens. These tags are identified using NLTK [41] tokenizer and POS tagger. This tagger is a greedy averaged perceptron model trained on the Wall Street Journal part of the Penn Treebank which is quite different from the biomedical domain. Even though this POS tagger is not a desirable one for the TAC dataset, a proper tagger is not trained in this study. The default POS tag-set is extended with the "TABLE" tag for differentiating table rows from proper sentences.

4.4. Bi-LSTM-CRF Model

LSTMs are used broadly in tasks that have to deal with sequences such as speech recognition, handwriting recognition, as well as NLP tasks. Its popularity comes from the recurrent aspect of the model, which allows it to extract features with minding the information coming from the context as well. For the NER task, different versions and combinations of the model have been applied [49]. Bidirectional LSTM (Bi-LSTM) can be considered as one of the promising models. Single LSTM can use context information from the processed tokens but not from the tokens to come in a sequence. The Bi-LSTM runs double LSTM models on a sequence where one of them starts from the beginning and the other from the end of a sequence. This version performs better since the output of a single token in a sequence contains information from both sides of the sequence. The outputs of the LSTMs are concatenated, as shown in Figure 4.5.

CRF, which is another widely used model for sequence tagging, performs similarly or better than a Bi-LSTM model for most of the time [49]. A probable reason for this is the fact that CRF models decode a given sequence all at once whereas even though Bi-LSTM have the context information, it does it separately for every token. Using both together where Bi-LSTM used as a feature extractor, outperforms previous variations [49]. Among the LSTM-CRF and Bi-LSTM-CRF, the latter performs better [49]. A similar Bi-LSTM-CRF structure is also utilized in there papers as well [46, 47]. Stacking the Bi-LSTMs before the CRF layer is another approach which is shown to be increasing the performance furthermore [48].

It is shown to be challenging to apply dropout on RNN layers. Hence, variational dropout is applied in the Bi-LSTM layers. This method applies the same mask through

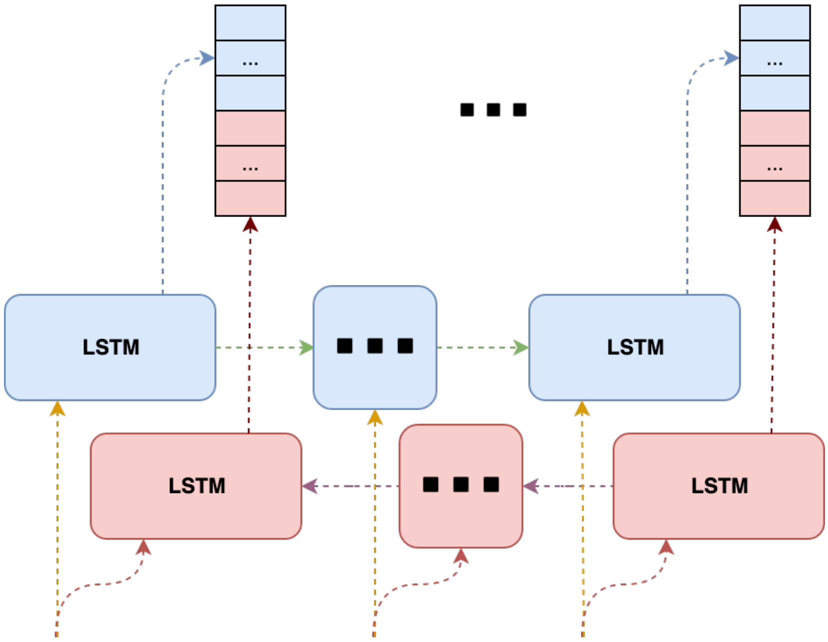


Figure 4.5. Bidirectional LSTM with variational dropout. The same colored arrows use the identical dropout masks.

time as well as to the input and the output [50]. Variational RNN is shown in Figure 4.5 where dropout masks are shown with colored dashed arrows.

4.5. SciMiner

In parallel to the neural network-based approach, dictionary- and rule-based NER approach is employed. SciMiner, written in Perl, was initially developed as a web-based literature mining platform for identifying genes and proteins in biomedical literature [51]. An expansion of SciMiner focusing on ADR, named as ADR-SciMiner, has been recently developed and applied to a study of ontology-based literature mining and drug class effect analysis of ADRs associated with drug-induced neuropathy [52].

Manual review of terms that are unlikely to be ADRs, such as various cancers, was performed. Various rules for term expansion as well as exclusions to increase coverage and accuracy were implemented. For example, Perl library Lingua::EN was used to expand the base ADR dictionary allowing the inclusion of additional plural or

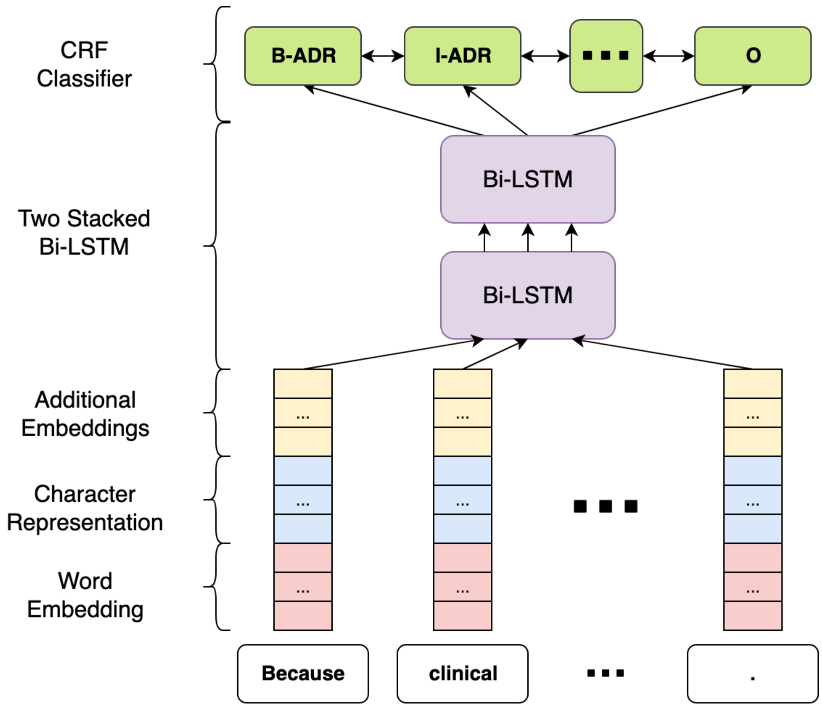


Figure 4.6. The complete sequence tagger neural network architecture

singular forms when only one form was included in the base dictionary. The SciMiner-based approach was also used for normalizing the positive ADR terms, identified by the neural learning-based approach described in the previous section, to their respective MedDRA PTs.

4.6. Model Hyper-parameters

Hyper-parameter is a type of a parameter, which is not learned by the training procedure, but changes the structure or execution of the model. The correct set of hyper-parameters can reduce training time and improve performance considerably. Some of the hyper-parameters for the model described above are given in the Table 4.4. Bi-LSTM-CRF model has more than one configuration, which changes the structure of the model entirely. In [48], Bi-LSTM-CRF variations are compared as well as the hyper-parameter selection and its effect on the performance. They conclude that Bi-LSTM stack size and hidden size, as well as character level embeddings selection have low impact. Optimizers, classifiers, RNN dropout type, and word embeddings

Table 4.4. Hyper-parameters of the network, bold values are selected.

	Values
Word Embeddings	Word2Vec, window size = 2, 5, 30
Word Embeddings Source	PMC, **PubMed**
Character Level Embeddings	**CNN**, Bi-LSTM
CNN Filter Count	20, **30**, 40, 50
Char-LSTM Hidden Size	**50**, 75, 100, 150, 200
Additional Features	**POS tags**, Casing
Bi-LSTM Stack Size	1, **2**, 3
Bi-LSTM Hidden Size	50, 75, **100**, 150, 200
RNN Dropout Type	None, Naive, **Variational**
RNN Dropout	0.1, **0.25**, 0.5
Classifier	Softmax, **CRF**, CRF-tanh
Optimizers	**NADAM**, ADAM, SGD, Adagrad, Adadelta
Epoch Count	20 – **50**
Batch Size	1, 2, 4, 8, 16, **32**, 64, 128, 256, 512
Learning Rate	0.001 – 1

selections have a high impact.

The tagger in this study is built upon the implementation and follows insights of the [48]. Therefore, even though the parameter space is very vast for grid search to handle, the focus of the hyper-parameter tuning is chunking scheme and word embeddings. The rest of the hyper-parameters are the same as the NER implementation of [48]. Maximum sentence length is set to be 100 since there are less than 500 sentences longer than 100 among all the test and training datasets. Maximum token length is set to be 40. Tokens that are longer than this threshold are usually the ones that could not be normalized in the preprocessing. These tokens do not bear any significance, since they are malformed. Therefore, they are discarded.

5. RESULTS

This chapter describes the TAC evaluation measures and the standings in the challenge. Following the official results, the results of the latest configuration of the proposed model are also provided.

5.1. TAC Evaluation and Challenge Results

The mention extraction task is evaluated with micro-averaged F_1 score for exact mention matches which are given in the Table 5.1. The primary metric is an exact match with correct mention types, but there is also a metric for only string matching without types. F_1 scoring is the harmonic mean of precision and recall which is widely used in binary classification problems. For a binary classification problem, correctly labeled instances are called true positives (TP) and incorrectly labeled instances are called false positives (FP). Instances that should have been labeled and not are called false negatives (FN) and ones that are not labeled correctly called false positives (FP). Precision is the portion of the correctly labeled instances among all labeled ones where recall is the portion of correctly labeled instances among all true instances.

$$precision = \frac{TP}{TP + FP} \quad recall = \frac{TP}{TP + FN}$$
$$F_1 = \left(\frac{recall^{-1} + precision^{-1}}{2}\right)^{-1}$$

F_1 scoring can be applied to multi-class labeling by disregarding the class information, which is called micro-averaged F_1. Another method, called macro-average F_1, is averaging the F_1 scores for every class. Calculation of micro-precision, micro-recall, macro-precision, and macro-recall for K classes is:

$$micro\_precision = \frac{\sum_k TP_k}{\sum_k TP_k + FP_k} \quad micro\_recall = \frac{\sum_k TP_k}{\sum_k TP_k + FN_k}$$
$$macro\_precision = \frac{\sum_k precision_k}{K} \quad macro\_recall = \frac{\sum_k recall_k}{K}$$

Table 5.1. TAC 2017 ADR Challenge results for the NER task. [53]

System (Run)	Precison	Recall	F_1
UTH CCB (3)	82.54	82.42	82.48
UTH CCB (2)	80.22	84.4	82.26
UTH CCB (1)	83.78	79.74	81.71
IBM Research	80.9	75.3	78.0
CONDL (1)	76.45	77.49	76.97
GN team (1)	80.19	72.23	76.0
GN team (2)	76.84	74.36	75.58
PRNA SUNY (1)	77.71	63.9	70.13
PRNA SUNY (3)	77.71	63.9	70.13
CONDL (3)	65.19	69.77	67.41
CONDL (2)	65.47	61.4	63.37
PRNA SUNY (2)	64.25	61.58	62.89
MC UC3M (1)	54.79	66.33	60.01
MC UC3M (2)	54.79	66.33	60.01
trddc_iiith	79.14	43.12	55.83
CHOP	57.95	29.64	39.22
BUPT_PRIS	40.47	11.81	18.29

All of the official results for the TAC challenge for normalization are given in the Table 5.2. Three different runs have been submitted for the official evaluation of the TAC Challenge. The first run, which obtains better results compared to the other two, uses the neural network model for mention extraction and SciMiner for normalization. The second run, which results in the lowest performance among the submitted three runs, uses SciMiner for both mention extraction and normalization. The third run uses SciMiner for ADR mention extraction and the neural model for the other types of mentions. It also uses SciMiner for normalization.

Table 5.2. TAC 2017 ADR Challenge results for the normalization task. [53]

System (Run)	Micro-P	Micro-R	Micro-F1	Macro-P	Macro-R	Macro-F1
UTHCCB (3)	84.17	89.84	86.91	83.02	89.06	85.33
UTHCCB (1)	85.00	87.75	86.35	84.04	86.67	84.79
UTHCCB (2)	82.42	90.78	86.40	80.83	89.90	84.53
CONDL (1)	88.81	77.16	82.58	88.20	75.76	80.50
PRNASUNY (1)	86.14	74.89	80.12	85.32	72.76	77.97
PRNASUNY (2)	81.55	78.24	79.86	79.80	76.03	77.25
PRNASUNY (3)	83.60	74.14	78.59	82.22	71.44	75.87
CONDL (2)	74.56	80.96	77.63	73.06	79.92	75.55
CONDL (3)	74.56	80.96	77.63	73.06	79.92	75.55
MCUC3M (1)	73.40	80.25	76.67	72.10	80.38	75.29
MCUC3M (2)	73.40	80.25	76.67	72.10	80.38	75.29
CHOP	71.78	50.14	59.04	70.12	49.84	57.27

Table 5.2 contains the official results for the normalization task. Macro-averaged F_1 score is the primary metric for the normalization task, but micro-averages are also provided.

One epoch of training the model with a character-level embedding layer, either LSTM or CNN, takes approximately 150 seconds while it takes 100 seconds without it. Some of the hyper-parameters of the model are listed in Section 4.6. Epoch count is set to be 50, which takes 2 hours to complete a run with character-level embedding layer and 1 hour and 20 munites otherwise on a computer with Intel 9900K CPU. The training time of the model makes grid search infeasible, which is required to find the optimal hyper-parameters. For this reason, the model has been tested with three different word embeddings, different character-level embeddings, and token normalization configuration that are listed in the Table 5.3.

Table 5.3. Effects of token normalization, word and character embeddings

Normalization	Embedding	Char Emb	+Typed	-Typed	Single
Yes	Win-30	LSTM	77.078	77.113	83.700
		CNN	77.424	77.494	83.900
		–	77.685	77.729	84.100
	Win-2	LSTM	76.985	77.048	84.600
		CNN	76.843	76.939	84.300
		–	77.491	77.533	84.700
	W2V	LSTM	77.059	77.094	84.300
		CNN	76.651	76.708	83.800
		–	77.236	77.507	84.200
No	Win-30	LSTM	76.337	76.414	84.000
		CNN	76.918	76.967	84.000
		–	76.940	76.996	84.700
	Win-2	LSTM	77.293	77.329	84.700
		CNN	76.613	76.663	84.700
		–	76.737	76.799	85.000
	W2V	LSTM	77.458	77.521	84.200
		CNN	76.833	76.861	84.300
		–	76.761	76.803	84.600

- *Normalization* column shows if the token normalization steps are applied or not on the preprocessing.
- *Embedding* column shows different word2vec embeddings. *Win-30*, *Win-2* [45], and *W2V* [44] have different window sizes during training
- *Char Emb* column stands for character-level embedding usage.
- *Typed* columns stand for the F_1 scores for exact matching scores. *+Typed* column holds the scores minding the type of the mention as well whereas *-Typed* column scores does not.
- *Single* column holds token level scores. Official scores are calculated with exact matching in mention level whereas this score contains partial matching as well.

6. DISCUSSIONS

The model components except for word embeddings are not domain-specific. All of the word embeddings that have been utilized are trained on a corpus that has been extracted from PubMed, therefore for other domains the captured latent semantic features would probably not be as helpful as they are for bioinformatics. The token normalization steps are not domain-specific as well. Steps three and four are there to improve models ability for tagging. Without those steps, tagging parts of such tokens would not be possible. It could be argued that this step can change the semantic meaning of the sentence when sub-tokens' embeddings are used instead of the token itself; therefore, it can reduce the overall performance of the model. Forementioned steps and step seven are closest to being domain-specific but not entirely since merging words with hyphens and dashes, and also equations are widely used outside of bioinformatics domain.

The differences between the scores listed in Table 5.3 are very small for the NER task, where the lowest score of the exact typed matching is 76.337, and the highest score is 77.491. Also, the overall performance is lower than the previously reported results (over %90 F1 score) on the CoNLL 2003 NER dataset [48]. Distribution of the class instances, annotation strategy, having irregular entities, poor quality of the text, and the small size of the training set could be the reason for the drop in the performance in the ADR domain.

The developed neural model has many hyper-parameters that affect the performance of the model on different levels. Therefore, the model has been tested for the character-level feature extraction, token normalization, and word embeddings. Using LSTM or CNN does not make much of a difference as expected, but surprisingly omitting character-level features seems to have very similar sometimes even better results. The embeddings denoted by Win-2, which are trained with window size 2, are expected to capture semantic meanings better than the other embedding types for the NER task [45]. However, our results showed that using different word embeddings does not have a significant effect on the performance.

Token normalization has the most consistent yet insignificant performance upgrade. Token normalization mostly fixes the problems occurring on the tables within the drug label texts which do not form a proper sentence, even though every row of a table is considered as a sentence. Table rows may also contain mention annotations. However, the developed model is designed to work on sentence level. Therefore, it may not be successful in identifying the mentions in the table rows, which don't correspond to well-formed sentences.

There were many teams who participated in the TAC 2017 shared task of adverse reaction extraction. The neural model in this work closely resembles the best performing models for the mention extraction task from [54] and [55], since they also used the Bi-LSTM-CRF as the core of the sequence tagger. The best-performing team, with the F1 score of 82.48, used a cascading Bi-LSTM-CRF model for extraction ADRs [54]. They trained two Bi-LSTM-CRFs, while the first one only tags ADR mentions, the second one on top of that tags the other mention types that are related to a single chosen ADR mention. The curation process of the TAC dataset mentions renders it infeasible to treat mention types as independent classes. Therefore, this model expected to perform better since it extracts the ADRs first and then extracts the rest of the mentions related to the candidate ADR. Even though they used BIO2 tagging, which is not fit to handle overlapping and disjoint entities, their model performed well, because they combined disjoint entities during preprocessing when possible [54]. They developed rules that learned from the training set for this purpose. These rules were later used to generate disjoint entities that have tagged as the output of the trained model. This approach allowed mentions to be continuous, thus making BIO2 tags to be more consistent. BIO2 chunking scheme adaptation in this study disrupts the transition probabilities of CRF since it disturbs the restriction of "I" tagged token to be after "O" tagged token.

The model of [55] has a single tagger for all entity types. One of the significant differences with the second-best performing model of [55], with the F1 score of 76.97, is the BIOHD tagging scheme. This tagging scheme is adopted and applied after the challenge. They trained a second sub-model, which makes use of attention mechanism [56], only to classify a disjoint entity segment pair to be merged or not.

In the normalization of the extracted ADR mentions onto the MedDRA ontology, the best performing team was again [54] with a micro-F1 score of 86.91 and a macro-F1 score of 85.33. It is hard to compare different approaches to this problem since this task is dependent on the performance of the first one. The performance levels of the normalization task could be considered as closer to our model compared to the NER task. The difference between ADR extraction performance is 6.2, whereas for the ADR extraction and normalization task the difference in the micro-F1 score is 4.33 and in the macro-F1 score is 4.83, as shown in Table 5.2.

7. CONCLUSION AND FUTURE WORK

In this study, two different methods are employed for detecting mentions of type ADR, drug class, animal, severity, factor, and negations from drug labels. The neural network-based approach outperformed the dictionary- and rule-based approach in terms of extracting ADRs. This study suggests that a system composed of a deep learning architecture for entity recognition and a rule-based model for entity normalization is a promising approach for ADR extraction from drug labels.

As future work, we will investigate incorporating ontology and dictionary knowledge into the deep learning model. Also doing an extensive parameter search is likely to increase the performance of the deep learning model. Incorporating label decoding scheme proposed for BIOHD in [57] instead of greedy decoding seems promising for reducing the errors in merging the detected entity segments into proper entities.

Attention mechanism [56] became more and more popular in the NLP community because of its computational efficiency and high performance over RNNs in various tasks such as embeddings generation [58] or NER [59] problems. We plan to investigate incorporating attention mechanism to the developed neural model for ADR extraction.

REFERENCES

1. World Health Organization and others, *The importance of pharmacovigilance*, Geneva: World Health Organization, 2002.

2. Ahmad, S. R., "Adverse drug event monitoring at the Food and Drug Administration: your report can make a difference", *Journal of general internal medicine*, Vol. 18, No. 1, pp. 57–60, 2003.

3. U.S. National Institute of Standards and Technology (NIST), *Text Analysis Conference (TAC) 2017*, `https://tac.nist.gov/2017/`, accessed in May 2019.

4. Nadeau, D. and S. Sekine, "A survey of named entity recognition and classification", *Lingvisticae Investigationes*, Vol. 30, No. 1, pp. 3–26, 2007.

5. Grishman, R. and B. Sundheim, "Message understanding conference-6: A brief history", *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996.

6. Tjong Kim Sang, E. F., "Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition", *COLING-02: The 6th Conference on Natural Language Learning 2002*, 2002, `https://www.aclweb.org/anthology/W02-2024`.

7. Sang, E. F. and F. De Meulder, "Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition", *arXiv preprint cs/0306050*, 2003.

8. Ohta, T., Y. Tateisi and J.-D. Kim, "The GENIA corpus: An annotated research abstract corpus in molecular biology domain", *Proceedings of the second international conference on Human Language Technology Research*, pp. 82–86, Morgan Kaufmann Publishers Inc., 2002.

9. Ratinov, L., D. Roth, D. Downey and M. Anderson, “Local and global algorithms for disambiguation to wikipedia”, *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 1375–1384, Association for Computational Linguistics, 2011.

10. Gurulingappa, H., J. Fluck, M. Hofmann-Apitius and L. Toldo, “Identification of adverse drug event assertive sentences in medical case reports”, pp. 16–27, First international workshop on knowledge discovery and health care management (KD-HCM), European conference on machine learning and principles and practice of knowledge discovery in databases (ECML PKDD), Sep. 2011.

11. Leaman, R., L. Wojtulewicz, R. Sullivan, A. Skariah, J. Yang and G. Gonzalez, “Towards internet-age pharmacovigilance: extracting adverse drug reactions from user posts to health-related social networks”, *Proceedings of the 2010 workshop on biomedical natural language processing*, pp. 117–125, Association for Computational Linguistics, 2010.

12. Sarker, A. and G. Gonzalez, “Portable automatic text classification for adverse drug reaction detection via multi-corpus training”, *Journal of biomedical informatics*, Vol. 53, pp. 196–207, 2015.

13. Nikfarjam, A. and G. H. Gonzalez, “Pattern mining for extraction of mentions of adverse drug reactions from user comments”, *AMIA Annual Symposium Proceedings*, Vol. 2011, p. 1019, American Medical Informatics Association, 2011.

14. Harpaz, R., A. Callahan, S. Tamang, Y. Low, D. Odgers, S. Finlayson, K. Jung, P. LePendu and N. H. Shah, “Text mining for adverse drug events: the promise, challenges, and state of the art”, *Drug safety*, Vol. 37, No. 10, pp. 777–790, 2014.

15. Karimi, S., C. Wang, A. Metke-Jimenez, R. Gaire and C. Paris, “Text and data mining techniques in adverse drug reaction detection”, *ACM Computing Surveys (CSUR)*, Vol. 47, No. 4, p. 56, 2015.

16. Brown, E. G., L. Wood and S. Wood, "The medical dictionary for regulatory activities (MedDRA)", *Drug safety*, Vol. 20, No. 2, pp. 109–117, 1999.

17. Nadkarni, P. M. and J. D. Darer, "Determining correspondences between high-frequency MedDRA concepts and SNOMED: a case study", *BMC medical informatics and decision making*, Vol. 10, No. 1, p. 66, 2010.

18. He, Y., S. Sarntivijai, Y. Lin, Z. Xiang, A. Guo, S. Zhang, D. Jagannathan, L. Toldo, C. Tao and B. Smith, "OAE: the ontology of adverse events", *Journal of biomedical semantics*, Vol. 5, No. 1, p. 29, 2014.

19. Guo, A., R. Racz, J. Hur, Y. Lin, Z. Xiang, L. Zhao, J. Rinder, G. Jiang, Q. Zhu and Y. He, "Ontology-based collection, representation and analysis of drug-associated neuropathy adverse events", *Journal of biomedical semantics*, Vol. 7, No. 1, p. 29, 2016.

20. Zurada, J. M., *Introduction to artificial neural systems*, Vol. 8, West publishing company St. Paul, 1992.

21. LeCun, Y., D. Touresky, G. Hinton and T. Sejnowski, "A theoretical framework for back-propagation", *Proceedings of the 1988 connectionist models summer school*, Vol. 1, pp. 21–28, CMU, Pittsburgh, Pa: Morgan Kaufmann, 1988.

22. Duchi, J., E. Hazan and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of Machine Learning Research*, Vol. 12, No. Jul, pp. 2121–2159, 2011.

23. Zeiler, M. D., "ADADELTA: an adaptive learning rate method", *arXiv preprint arXiv:1212.5701*, 2012.

24. Nesterov, Y., "A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$", *Soviet Math. Dokl*, Vol. 27.

25. Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting”, *The Journal of Machine Learning Research*, Vol. 15, No. 1, pp. 1929–1958, 2014.

26. Wan, L., M. Zeiler, S. Zhang, Y. Le Cun and R. Fergus, “Regularization of neural networks using dropconnect”, *International conference on machine learning*, pp. 1058–1066, 2013.

27. Karpathy, A., *CS231n Convolutional Neural Networks for Visual Recognition*, `http://cs231n.github.io/neural-networks-3/`, accessed in Jun 2019.

28. Hubel, D. H. and T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”, *The Journal of physiology*, Vol. 160, No. 1, pp. 106–154, 1962.

29. Fukushima, K., “Neural network model for a mechanism of pattern recognition unaffected by shift in position-Neocognitron”, *IEICE Technical Report, A*, Vol. 62, No. 10, pp. 658–665, 1979.

30. Bengio, Y., P. Simard, P. Frasconi *et al.*, “Learning long-term dependencies with gradient descent is difficult”, *IEEE transactions on neural networks*, Vol. 5, No. 2, pp. 157–166, 1994.

31. Mikolov, T., *Statistical language models based on neural networks*, Ph.D. Thesis, Brno University of Technology, 2012.

32. Pascanu, R., T. Mikolov and Y. Bengio, “On the difficulty of training recurrent neural networks”, *International conference on machine learning*, pp. 1310–1318, 2013.

33. Hochreiter, S. and J. Schmidhuber, “Long short-term memory”, *Neural computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

34. Lafferty, J., A. McCallum and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data", *Proceedings of the 18th International Conference on Machine Learning*, pp. 282–289, 2001.

35. Santorini, B., *Part-of-speech tagging guidelines for the Penn Treebank Project*, University of Pennsylvania, School of Engineering and Applied Science ..., 1990.

36. Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality", *Advances in neural information processing systems*, pp. 3111–3119, 2013.

37. Sang, E. F. and J. Veenstra, "Representing text chunks", *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pp. 173–179, Association for Computational Linguistics, 1999.

38. Ratnaparkhi, A., *Maximum entropy models for natural language ambiguity resolution*, Ph.D. Thesis, University of Pennsylvania, 1998.

39. Marcus, M., B. Santorini and M. A. Marcinkiewicz, *Building a large annotated corpus of English: The Penn Treebank*, Tech. Rep. MS-CIS-93-87, University of Pennsylvania Department of Computer and Information Science, 1993.

40. Demner-Fushman, D., S. E. Shooshan, L. Rodriguez, A. R. Aronson, F. Lang, W. Rogers, K. Roberts and J. Tonning, "A dataset of 200 structured product labels annotated for adverse drug reactions", *Scientific data*, Vol. 5, p. 180001, 2018.

41. Bird, S., E. Klein and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*, O'Reilly Media, Inc., 2009.

42. Tang, B., Q. Chen, X. Wang, Y. Wu, Y. Zhang, M. Jiang, J. Wang and H. Xu, "Recognizing disjoint clinical concepts in clinical text using machine learning-based methods", *AMIA Annual Symposium Proceedings*, Vol. 2015, p. 1184, American

Medical Informatics Association, 2015.

43. Pennington, J., R. Socher and C. Manning, "Glove: Global vectors for word representation", *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

44. Pyysalo, S., F. Ginter, H. Moen, T. Salakoski and S. Ananiadou, "Distributional semantics resources for biomedical text processing", *Proceedings of LBM*, pp. 39–44, 2013.

45. Chiu, B., G. Crichton, A. Korhonen and S. Pyysalo, "How to train good word embeddings for biomedical NLP", *Proceedings of the 15th workshop on biomedical natural language processing*, pp. 166–174, 2016.

46. Ma, X. and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf", *arXiv preprint arXiv:1603.01354*, 2016.

47. Lample, G., M. Ballesteros, S. Subramanian, K. Kawakami and C. Dyer, "Neural architectures for named entity recognition", *arXiv preprint arXiv:1603.01360*, 2016.

48. Reimers, N. and I. Gurevych, "Reporting score distributions makes a difference: Performance study of lstm-networks for sequence tagging", *arXiv preprint arXiv:1707.09861*, 2017.

49. Huang, Z., W. Xu and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging", *arXiv preprint arXiv:1508.01991*, 2015.

50. Gal, Y. and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks", *Advances in neural information processing systems*, pp. 1019–1027, 2016.

51. Hur, J., A. D. Schuyler, D. J. States and E. L. Feldman, "SciMiner: web-based

literature mining tool for target identification and functional enrichment analysis", *Bioinformatics*, Vol. 25, No. 6, pp. 838–840, 2009.

52. Hur, J., A. Özgür and Y. He, "Ontology-based literature mining and class effect analysis of adverse drug reactions associated with neuropathy-inducing drugs", *Journal of biomedical semantics*, Vol. 9, No. 1, p. 17, 2018.

53. Roberts, K., D. Demner-Fushman and J. M. Tonning, "Overview of the TAC 2017 Adverse Reaction Extraction from Drug Labels Track.", *TAC*, 2017.

54. Xu, J., H.-J. Lee, Z. Ji, J. Wang, Q. Wei and H. Xu, "UTH_CCB System for Adverse Drug Reaction Extraction from Drug Labels at TAC-ADR 2017.", *TAC*, 2017.

55. Dandala, B., D. Mahajan and M. V. Devarakonda, "IBM Research System at TAC 2017: Adverse Drug Reactions Extraction from Drug Labels.", *TAC*, 2017.

56. Zhou, P., W. Shi, J. Tian, Z. Qi, B. Li, H. Hao and B. Xu, "Attention-based bidirectional long short-term memory networks for relation classification", *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 207–212, 2016.

57. Li, F., M. Zhang, B. Tian, B. Chen, G. Fu and D. Ji, "Recognizing irregular entities in biomedical text via deep neural networks", *Pattern Recognition Letters*, Vol. 105, pp. 105–113, 2018.

58. Devlin, J., M.-W. Chang, K. Lee and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*, 2018.

59. Bahdanau, D., K. Cho and Y. Bengio, "Neural machine translation by jointly learning to align and translate", *arXiv preprint arXiv:1409.0473*, 2014.