2020

Classifying Relations using Recurrent Neural Network with Ontological-Concept Embedding

Mario J. Lorenzo
Nova Southeastern University, mario@mjlorenzo.com

Classifying Relations using Recurrent Neural Network with Ontological-Concept Embedding

by

Mario J. Lorenzo

A dissertation submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy

in

Computer Science

College of Computing and Engineering

Nova Southeastern University

2020

We hereby certify that this dissertation, submitted by Mario Lorenzo conforms to acceptable standards and is fully adequate in scope and quality to fulfill the dissertation requirements for the degree of Doctor of Philosophy.

Sumitra Mukherjee — December 8, 2020

Sumitra Mukherjee, Ph.D.
Chairperson of Dissertation Committee

Date

December 8, 2020

Michael J. Laszlo, Ph.D.
Dissertation Committee Member

Date

Dec 8, 2020

Francisco J. Mitropoulos, Ph.D.
Dissertation Committee Member

Date

Approved:

December 8, 2020

Meline Kevorkian, Ed.D.
Dean, College of Computing and Engineering

Date

College of Computing and Engineering
Nova Southeastern University

2020

An Abstract of a Dissertation Submitted to Nova Southeastern University
in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

Classifying Relations using Recurrent Neural Network

with Ontological-Concept Embedding

by

Mario J. Lorenzo
2020

Relation extraction and classification represents a fundamental and challenging aspect of Natural Language Processing (NLP) research which depends on other tasks such as entity detection and word sense disambiguation. Traditional relation extraction methods based on pattern-matching using regular expressions grammars and lexico-syntactic pattern rules suffer from several drawbacks including the labor involved in handcrafting and maintaining large number of rules that are difficult to reuse. Current research has focused on using Neural Networks to help improve the accuracy of relation extraction tasks using a specific type of Recurrent Neural Network (RNN). A promising approach for relation classification uses an RNN that incorporates an ontology-based *concept embedding* layer in addition to word embeddings. This dissertation presents several improvements to this approach by addressing its main limitations. First, several different types of semantic relationships between concepts are incorporated into the model; prior work has only considered *is-a* hierarchical relationships. Secondly, a significantly larger vocabulary of concepts is used. Thirdly, an improved method for concept matching was devised. The results of adding these improvements to two state-of-the-art baseline models demonstrated an improvement to accuracy when evaluated on benchmark data used in prior studies.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Relation extraction and classification is a common Information Extraction task involving the linking of entities that participate in a semantic or syntactic relationship within a text document. Relation extraction involves the detection of the presence of a relation between a pair of entities, while relation classification attempts to identify the type of relation between a pair of entities. Relation extraction and classification represent a fundamental and challenging aspect of Natural Language Processing (NLP) research and depend on other tasks such as entity detection and word sense disambiguation.

Traditional relation extraction methods were based on pattern-matching using regular expressions grammars and lexico-syntactic pattern rules (Konstantinova, 2014). These approaches suffer from several drawbacks including the labor involved in hand-crafting and maintaining large number of rules that are difficult to reuse and inevitably lead to ambiguity when rules overlap (Chiticariu, L., Li, Y., Raghavan, S., & Reiss, F. R., 2010).

Current research has focused on using Neural Networks to help improve the accuracy of relation extraction tasks using a specific type of Recurrent Neural Network (RNN) (Rumelhart, D. E., Hinton, G. E., & Williams, R. J, 1988) called Long-Short Term Memory (LSTM) proposed by Hochreiter & Schmidhuber (1997) and further refined to add Bidirectionality (BiLSTM) by Schuster & Paliwal (1997). These special RNN were designed to detect long distance patterns that frequently occur within time-series and sequence tagging problems such as entity and relation extraction tasks. Entity extraction (sometimes known as entity chunking) involves the identification of entities within unstructured text by identifying the beginning and ending offsets of each entity within the unstructured text. LSTM uses a gated short-term and long-term memory unit that can hold and forget information over a sequence of time steps (Hochreiter & Schmidhuber, 1997) and is ideally suited to remember patterns that occur between longer distances across the input sequence. This property is especially useful for relation extraction because entities participating in a relation can occur at the beginning and the end of a sentence.

Recent research in NLP has found the use of Word Embedding layers (Mikolov, 2013; Pennington, 2014) within an RNN model as a promising approach that has

demonstrated higher performance and robustness when compared to earlier pattern-based matching or classical feature-based machine learning algorithms. Lamurias & Couto (2019) proposed a novel LSTM-based architecture extending the work of Zhang Y., Zheng, Lin, Wang, Yang Z., & Dumontier (2018) and Xu B., Shi, Zhao, & Zheng (2018) by incorporating a biomedical ontology-based *concept embedding* layer in addition to the frequently used word embeddings. Lamurias & Couto name this model: *BO-LSTM*. Their approach infuses additional domain-specific knowledge into the learning process by encoding words and phrases into concepts that are in turn transformed into a dense feature vector through an ontology-based concept embedding layer. Lamurias & Couto selected the Chemical Entities of Biological Interest (ChEBI) ontology (Hill et al, 2013) as the most topically relevant ontology to support their concept embedding work. This ontology defines drugs and their underlying molecular structures through a collection of concepts and semantic relations between concepts. Their approach demonstrated state-of-the-art performance when evaluated using a well-known relation classification benchmark called the SemEval 2013 Drug-Drug Interaction (DDI) task (Segura-Bedmar et al, 2014; Herrero-Zazo et al, 2013).

The SemEval DDI 2013 dataset includes medical literature documents that contain mentions of drug interactions. This dataset also includes ground truth that labels the correct drug entities and whether there exists a drug interaction relationship between pair of mentioned drug names. These relations can be classified into 5 types: mechanism, effect, advice, interaction not specified (labeled ‘int’), or non-interacting (i.e. negative). *Figure 1* shows an example of the DDI ground truth format used. This example includes a sentence with two identified drug entities with relative offsets and a candidate relation pair with corresponding relation classification label of *effect*. When more than two drug entities are found within the sentence, each distinct pair will be captured and one of the five DDI relation classes will be labeled.

```
<sentence id="DDI-MedLine.d5.s4" text="In ewes given 40 mg of phenobarbital sodium/kg for 5 days
intraperitoneally (IP), the anticholinesterase effect of 4 mg of coumaphos/kg was significantly
reduced and signs of toxicity were not present. ">
    <entity id="DDI-MedLine.d5.s4.e0" charOffset="23-42" type="drug" text="phenobarbital
    sodium"/>
    <entity id="DDI-MedLine.d5.s4.e1" charOffset="123-131"
        type="drug_n" text="coumaphos"/>
    <pair id="DDI-MedLine.d5.s4.p0" e1="DDI-MedLine.d5.s4.e0"
        e2="DDI-MedLine.d5.s4.e1" ddi="true" type="effect"/>
</sentence>
```

Figure 1 - Example from SemEval DDI 2013 dataset

The performance of a model for the DDI relation extraction task is measured in the form of F_1 accuracy score. The F_1 score represents the combined harmonic mean of recall and precision where recall measures the proportion of correct DDI relations found while precision measures that only those correct relations, and no others, are found.

Problem Statement

The proposed approach by Lamurias & Couto (2019) suffers from several drawbacks and restrictions. One of these drawbacks involves the limitation of only considering *is-a* hierarchical relations from ChEBI and discarding many other semantic relations that may provide additional salient information to the model about a pair of drug entities under consideration. An example of some of these semantic relations from the ChEBI ontology include: *has_part*, *has_role*, and *is_conjugate_acid_of* among others. Without considering these other relations, many important properties of these drugs are missed and not visible to the learning algorithms during training.

Another limitation involves the use of a small vocabulary of concepts that only include 2,000 of the more than 114,000 available concepts within ChEBI. This restriction results in a high-occurrence of *Out-of-Vocab* (or OOV) problem encountered by embeddings that limit the words (or concepts) they include in the vocabulary (Young, 2019). This results in a reduction in accuracy performance when evaluated against a blind (holdout) dataset. This limitation also makes it difficult to generalize the model beyond the training data and raises questions about its viability in a real-world dataset.

A third drawback with the Lamurias & Couto *BO-LSTM* model involves the matching of concepts. The approach uses a heuristic based on a similarity score that can sometimes match the wrong concept within the ontology and therefore mislead the rest of

the model. Because the same word can be mapped to multiple concepts within the ontology, this creates an ambiguity that must be resolved to identify the correct concept (Ciaramita et al, 2006).

The misidentification of concepts by the BO-LSTM concept matching method takes two forms. The first involves the matching of a substring from the input text to a concept in the ontology when there exists a longer string that matches a more specific concept. Funk, Baumgartner, Garcia, Roeder, Bada, Cohen, ... & Verspoor (2014) describe this problem as the *under-specification* of a concept and the leading cause of false-positive matches when using biomedical ontologies such as ChEBI.

The following sample sentence taken from the DDI dataset serves as an example: "Concomitant administration of vitamin A and medications with retinoic acid receptor alpha antagonists must be avoided because of the risk of hypervitaminosis A." The concept matching may find many overlapping possible concepts such as "acid", "retinoic acid", "antagonists", and "alpha antagonists" but there exists a more specific concept, with a longer matching string, called "retinoic acid receptor alpha antagonist". The longest spanning string represents the most specific concept and should be the only one selected over that span. This means no other substring such as "acid" or "antagonist" should be matched when the longer match exists.

The second form of misidentification involves the ambiguity that arises when the same word (or words) can match multiple concepts. Without the use of the context surrounding these words, the identification of the most relevant concept may yield the wrong concept. An example of this problem can be described with the word "ice". *Ice* may represent a concept for the solid state of water, while in a different context, "ice" may represent a Federal Law Enforcement Agency or the street name synonym for methamphetamine. Without the context to resolve these concepts correctly, the model can fail to properly identify whether a DDI relation exists between a pair of entities.

Dissertation Goal

The primary goal of this study was to extend the Lamurias & Couto's BO-LSTM model to expand the use of an ontology-embedding layer by providing a new concept embedding method that can capture the semantic relationships of the ancestors for the

concepts. This study also presents a method that improve upon the concept matching techniques used in BO-LSTM by matching words and phrases to the most specific and relevant concepts. The key methods explored in this study are summarized as follows:

1. Devised a new concept embedding that utilizes hierarchical and semantic relationships for concepts within the ChEBI ontology. The embedding projects a concept representing a drug entity into one or more low-dimensional feature space. The number of concept vectors will depend on the number of relation types within ChEBI that the concept participates in. The embedded concepts are presented to an LSTM layer of the model as input so that it can learn higher semantic patterns from the ontology.
2. A concept matching method to find the most specific concept within the ontology by using a greedy matching method that looks for the longest possible match. This longest-span match method produces the most specific concept by discarding matches that are contained as a subsequence within a longer matching span. An implementation of Concept Mapper (Tanenblatt, Coden, & Sominsky, 2010), considered by Funk et al (2014) as the best performing concept matching method for biomedical ontologies, is evaluated along with two other methods.

These methods were all evaluated using the benchmark used by the previous studies (the SemEval 2013 DDI challenge) and the same ontology used by Lamurias & Couto (2019): the ChEBI Ontology. The performance of this model aimed to deliver an overall F_1 accuracy improvement over the BO-LSTM (Lamurias & Couto, 2019) and Hierarchical RNN (Zhang Y. et al, 2019) baseline models. This approach also aimed to demonstrate its ability to support larger ontologies by supporting a larger vocabulary (100,000 concepts) and completing the model training activity within a reasonable amount of time (one day) using the SemEval DDI data set.

Relevance and Significance

The study of Information Extraction (IE) or Natural Language Processing (NLP) has focused on developing methods for processing unstructured information and extracting valuable insights that assist human knowledge workers and researchers quickly obtain answers to their questions. The present day COVID-19 pandemic has demonstrated the

need for effective methods of processing massive amounts of medical information to speed up the investigation of vaccines and treatments. The COVID-19 outbreak has generated an unprecedented global response to sharing medical research with the goal of increasing collaboration amongst research organizations to support drug discovery that can help treat sick patients or vaccinate them from the disease. This unprecedented sharing of information includes previous drug research for coronaviradae (i.e. coronavirus family), observational clinical studies from hospital ICU, and early clinical trial efficacy and drug adverse reaction reports. All this information is largely represented as unstructured text in the form of journal articles, technical reports, and clinical studies. Much of this information is available on the web through government research web sites and private research organizations such as the CDC, FDA, the Allen Institute, Elsevier, National Library of Medicine among many others.

Unfortunately, the rate of contribution to medical literature significantly outpaces the ability for a clinician or researcher to read and understand in a timely manner. As a result, these experts have turned to AI & NLP tools to help distill and summarize the information so that it can be indexed and organized in ways that helps convey insights to research questions. During the recent COVID-19 pandemic response, researchers and clinicians turned to medical literature to help find answers to questions about effective medications that could help treat sick patients that frequently suffer from multiple chronic comorbidities and were frequently on multiple medications. Physicians must cautiously navigate around complex prescribing guidelines when dealing with these complex cases. The FDA provides guidance to help physicians avoid dangerous events called Adverse Drug Reactions (ADR), an injury to the patient due to a side-effect or negative interaction of a drug with another drug (Segura-Bedmar et al, 2010).

The field of pharmacovigilance involves the prevention, monitoring, and reporting of ADR after a drug has been approved by the FDA. There are various initiatives in place to help prevent ADRs from occurring including frequent publications that warn physicians about new discovery of drug side-effects or contraindications for prescribing a drug. Pharmacovigilance also includes the aggregation of reporting by hospitals, pharmacies, and patients regarding ADR. But even with these initiatives in place, ADR occurs in over one-third of hospital visits and resulting in over 2 million hospitalizations in the U.S. yearly

(Lazarou, Pomeranz, & Corey, 1998). The yearly cost of ADR's to the U.S. Healthcare system was previously estimated at $30 Billion (Johnson Jeffery and Booman Lyle, 1996).

The study of Drug-Drug Interactions (DDI) is considered a key area of ADR research and involves the interference of a drug by one or more drugs taken concomitantly. The awareness and detection of DDI is considered an integral part of reducing ADR incidents. Once a DDI is discovered, pharmacy and hospital prescription ordering systems can be updated to alert when there is a potential contraindication due to the presence of another incompatible drug. Currently, these systems are maintained by teams of humans who scour the medical literature in search of DDI that should be reported and detected by automated systems. This human process is slow and often misses subtle statements within biomedical research describing the observation of a DDI. As such, the biomedical research community began to explore NLP tools to help with the automation of DDI identification in literature (Haas, Iyer, Orav, Schiff, & Bates, 2010; Segura-Bedmar et al, 2010).

The field of NLP is comprised of a common set of tasks performed on unstructured text helping to extract the syntactic and semantic structure of what meaning the text is conveying. These common tasks include sentence segmentation, tokenization, lemmatization, normalization, part-of-speech (POS) tagging, entity recognition, concept matching, and relation extraction, among others. The need for improved relation extraction methods is a crucial prerequisite to natural language understanding and an integral aspect of knowledge discovery, Q&A, semantic search engines, and decision support systems (Bach and Badaskar, 2007).

Lamurias & Couto (2019) is based on current state-of-the-art methods based on deep learning using LSTM neural networks (Hochreiter, 1997) that have demonstrated F_1 scores of .75 when extracting drug-drug interaction relationships from biomedical text (W. Zheng et al, 2017). A common method used with LSTM models is to include a word embedding layer trained using word2vec (Mikolov, 2013) or GloVe (Pennington, 2014) on a corpus of documents.

Word embeddings have demonstrated significant improvement to performance because they can detect relationships between entities using unsupervised algorithms such as skip-gram and CBOW (continuous bag of words). Word embeddings have also proven to be an effective solution for dealing with a neural network's difficulty in dealing with

sparse data. By mapping a sparse feature representation, such as one-hot binary encoding, to a dense and low-dimensional feature space, a neural network requires fewer weights (less neurons) and therefore less training data in order to converge on an optimal solution. Additionally, word embeddings also provide semantic similarity between word vectors where neighbors within the embedding space implies similar word meaning. The positive impact on accuracy for entity extraction tasks has been well documented in literature (Bojanowski, 2017; Mikolov et al, 2013; Pennington et al, 2014; Pyysalo et al, 2013).

There are several well-known drawbacks with using word embedding including missing the underlying semantic meaning that underlie the words in a sentence (Lucy and Gauthier, 2017), out-of-vocab problem (when words are not in the embedding space), inability to represent phrases (2 or more words), unable to distinguish multi-sense words (polysemy). See (Young et al, 2018) for a recent review of modern word embedding methods and limitations.

Current research into relationship extraction tasks have focused on LSTM architecture using Word Embedding layers as a promising approach that has demonstrated higher performance and robustness when compared to earlier pattern-based matching or classical feature-based machine learning algorithms. Lamurias & Couto (2019) propose a novel LSTM-based architecture that incorporates an ontology-based concept embedding layer in addition to the frequently used word embedding layer. Lamurias & Couto postulate that the ontology serves as a representation of domain-specific knowledge that is not observable through the training data and therefore depriving the learning algorithm from observing additional patterns that exist behind the words. Specifically, the approach focuses on extracting ancestor concepts for each entity within the input text sequence and produce dense vectors representation based on the ancestor concepts within an ontology. This observation is supported by other researchers that have demonstrated successful results with incorporating domain-specific knowledge to supplement their training data (Xu, 2018; Li, 2016; Kong, 2013; and Deasigi, 2017). Lamurias & Couto demonstrated that including this embedding layer improves the accuracy of the relation extraction task. This was demonstrated by extending a baseline model from Zhang et al (2018) and comparing the performance of the model with and without an embedding layer. The results demonstrated an F_1 improvement of .022 when evaluated against the SemEval 2013 DDI

benchmark. This adapted model was considered the second all-time highest performer in this benchmark with a .75 score.

Barriers and Issues

The detection and classification of relations among entities within unstructured text is a complex task with dependencies on other subtasks such as entity detection, concept matching, and syntactic dependency parsing, among others. Although there have been significant improvements to performance on many of these tasks, relation extraction remains difficult. This study sought to identify additional improvements over the current state-of-the-art by incorporating additional embedding layers that enrich the model features with expert knowledge curated in the form of domain specific ontologies, such as ChEBI, that expose hidden salient patterns that can be learned by downstream RNN LSTM layers of a model.

During this investigation, several challenges were identified that required resolution to achieve the stated goals. These challenges span several areas of the methodology.

1. The first challenge involves an efficient representation of a large ontology to support concept matching and traversing ontological relations. The data structure and algorithms must be able to scale to hundreds of thousands of concepts, millions of relations, and provide an efficient concept look-up using the canonical form and synonyms for concepts. This work sought to flatten and prune the ontology to only include the minimal metadata needed for performing concept matching and traversing relevant relationships. This means removing all other metadata such as definitions, spurious labels, provenance, and unused fields.
2. The second challenge involves the ability to accurately identify and match concepts mentioned in the input text with the corresponding concepts within the ontology. This work explored the use of Concept Mapper as a tool for constructing a concept matching dictionary with several sophisticated algorithms for matching concepts. Additional experimentation evaluated alternatives to Concept Mapper. Concept matching is an important subtask required to support the ontology-concept embedding method presented in this

study. During experimentation, using a basic string-matching approach failed to match ~30% of the drug concepts in the training dataset. This is a significant false-negative rate and directly reduces the effectiveness of the Ontological embedding layer since a lack of a concept nullifies the effects of the layer on model performance.

3. Another challenge involved the selection of the best concept among multiple possible concepts. Experiments using the SemEval DDI training dataset found that 78% of drug mentions resulted in more than one possible concept match using the ChEBI ontology and a basic string-matching between the mentioned drug and the canonical or synonyms of the concepts. This can result in a deleterious effect on the learning process if the wrong concept is selected in an inconsistent manner resulting in noise.
4. Another challenge present when using the method proposed by Lamurias & Couto (2019) involved the situation when there is a lack of common ancestor relations between pairs of candidate drug entities. Theoretically this should not occur in an ontology that includes a top-level root concept that all concepts descend from within the *is-a* (subsumption) relation. However, when including other relation types, there is a strong likelihood that some pairs of drugs do not participate in the same relations and therefore have no common ancestry. This methodology explored the use of a single summary vector, called an *Identity Vector*, representing all the embedded relations.
5. Another challenge involved multiple inheritance paths (multiple ancestors) within the ontology. During preliminary experimentation, a significant percentage of drug concepts descended from multiple parent concepts. This issue was not addressed by (Lamurias & Couto, 2019) or in the literature. This study evaluated a neural embedding strategy that traverses all direct and indirect ancestors along taxonomic (is-a) and semantic relations when training the embedding model.
6. Class imbalance has been a consistent issue documented in the literature. There are several known methods that can be used to mitigate the impact of class imbalance including sample normalization and class weights. Data analysis

demonstrated a 5.9-to-1 ratio between negative and positive drug-drug interactions in the available training data and a much larger imbalance between classes such as the *Int* label class accounting for only 0.6% of the total samples (i.e. 189 of 27,756 samples).

7. Throughout the literature, the use of pre-trained word embeddings demonstrated significant improvements in performance. One popular pre-trained word vector is the Google News w2v that includes 2 million words. During experimentation, many out-of-vocab (OOV) errors occurred due to the nature of a medical dataset that features many rare words that are not encountered in the open domain. This study used a pre-trained word vector model that was trained using Medline corpus of 26 million documents and features a vocabulary of approximately 2 million words. The medical word vector embedding model resulted in far fewer OOV problems. When testing with the Medline sourced training data from SemEval DDI 2013, ~40 unique words were not found compared to ~1,700 unique words not found using Google News. Most of these missing words were drug names and domain-specific terminology that is vital to understanding the DDI relation classification task.
8. This study infused additional aspects of the ontology into the model through an embedding layer connected to an RNN/LSTM. This required a feature representation scheme of ontology relation information to train the embedding layer. During literature review, no specific solutions were found for this problem.
9. High Computational requirements to train deeply layered RNN/LSTMs results in long training time and consumed significant computing resources. For example, training a single Epoch took between ~30-60 minutes and consumed ~9GB of memory.

Summary

This chapter defined the relation extraction task for unstructured text and outlined the corresponding challenges and significance of this problem in the context of the broader

natural language understanding field. The expressed goal of this research was to address numerous challenges and limitations such as accessing hidden information that is captured within domain specific ontologies and using that information to enrich the feature patterns exposed to the neural learning layers. This study builds upon the existing body of work and current-state-of-the-art in relation detection and classification by incorporating an ontology embedding layer that includes semantic relations found within biomedical ontologies such as ChEBI. The study used a comparative approach by establishing two baseline models as the control and evaluating the control and the enhanced model using the SemEval DDI 2013 benchmark.

The rest of this dissertation report includes a thorough Literature Review of the problem and solution domain in Chapter 2, a detailed methodology description and approach discussed in Chapter 3, experiment results and findings in Chapter 4, and dissertation conclusion and future work in Chapter 5.

Chapter 2

Literature Review

This chapter presents a thorough review of the relevant body of literature as it pertains to Relation Extraction within unstructured text. The following areas of literature are considered in scope and serve an integral role in the evolution of the problem and solutions for relation extraction tasks. Relation extraction remains a complex task and an active area of research given the critical role it plays in ascertaining the meaning of a text. This review will discuss the evolution of methodology beginning with early methods and ending with the current state-of-the-art. The review will compare the presented methodologies using well-established benchmarks for relation detection and classification within the biomedical domain. The review will conclude by providing observations including gaps and unexplored areas of the field.

The formation of IE and NLP field, dates to the Message Understanding Conferences held in the 1980's (MUC) where many of the NLP common tasks were defined along with common frameworks to evaluate their relative performance metrics. These conferences were utilized by government agencies like DARPA and industry to promote and motivate research into unexplored areas. This formalization of common tasks and corresponding benchmarks served as a catalyst for new research into algorithms that would serve as the foundation for many of today's core NLP tasks such as syntactic parsers and NER. However, it was not until the MUC-7 conference (MUC-7, 1998), that relation extraction was formalized and research interest into the problem began to increase. The rise of the internet in the late 1990's and into the early 2000 led to an expansion of online publications and scholarly journals that fueled a resurgence of interest into extracting knowledge and insights from a rapidly growing bodies of knowledge (Konstantinova, 2014). The state-of-the-art methods at the time rarely yielded F-score performance above .60 on relation extraction tasks over open-domain datasets (Chiticariu, 2018).

Relation extraction tasks can be decomposed into two types of tasks. Relation Detection is a binary classification task where the presence of a given relation is detected. An example of relation detection could be to identify whether a pair of drugs mentioned in a statement are describe as having a DDI when taken together. Relation Classification can be a multilabel classification task that determines where there is a relation and what type of relation is present. An example of relation classification would be to determine the type of DDI relation between a pair of drug candidates mentioned in a statement.

Both Relation Detection and Classification rely on first performing Entity Detection or Entity Recognition to identify the candidates being considered for participation within a relation. For the DDI example, these entities are drugs (i.e. medications) mentioned in text. The entity recognition task involves identifying the offset locations of the entities and the type of entity identified. In the context of ML methods, entity recognition is modeled as a many-to-many sequence tagging task where a variable length input is presented to the model and each token is tagged with an entity type label. Entity recognition is a form of type of classification. The performance of entity recognition models using Deep Learning methods have achieved high performance over the last decade (Young, Hazarika, Poria, Cambria, 2018).

The benchmarks presented in this study are isolated to evaluating the performance relation extraction models and assume that the candidate entities along with the corresponding input text are provided. This helps to rigorously evaluate and compare different relation extraction methodologies without interference from entity detection tasks. Further discussion of entity detection is out of scope for this review, see the recent review by Young et al (2019) for a survey into these other NLP tasks.

Early Relation Extraction

Early relation extraction methods were based on pattern-matching (Riloff and Jones, 1999), lexico-syntactic patterns (Hearst et al, 1992), cascading grammars (Boguraev, 2004), relational (Reiss, 2008; Krishnamurthy, 2009; Chiticariu, 2010), and others (Fukumoto et al, 1998; Garigliano et al 1998; Humphrey et al, 1998). See (Bach and Badaskar, 2007) and (Konstantinova, 2014) for a comprehensive literature review of early relation extraction methods. All these approaches suffer from several drawbacks

including: lack of portability of rules (Konstantinova, 2014), inherent limitation in scaling large number of rules (Reiss, 2008), ambiguity when resolving overlapping rules (Chiticariu, 2010), and the labor involved in hand-crafting and maintaining many rules.

Other approaches for relation extraction are based on supervised learning methods such as: logistic regression (Kambhatla, 2004), kernel-based methods (Zhao and Grishman, 2005; Bunescu and Mooney, 2006), Condition Random Field (Culotta, 2006) and semi-supervised methods (Brin et al, 1998; Agichtein and Gravano, 2000; and Etzioni, 2005). Most of these methods are focused on extracting only binary relations. McDonald et al (2005) presents a method for extracting higher-order relationships.

Standard Relation Extraction Benchmark

The establishment of formal communities with well-defined common tasks and evaluation metrics with datasets and benchmarks have demonstrated a significant advancement in NLP methodology and performance (Segura-Bedmar et al, 2011). As NLP tooling and frameworks have become more mature, their adoption by industry and government has increased to assist with processing of large-scale datasets across a multitude of domains.

However, much of the focus and progress of these methods were aimed at open-domain topics that include news articles, social media posts, and consumer feedback reviews. These topics tend to have more bounded vocabulary and straightforward syntactic structures to appeal to a broad audience. Unfortunately, NLP tools were inadequate and insufficient for meaningful use within a scientific domain such as biomedical literature. The poor performance of these methods within the biomedical domain is attributed to the complex vocabulary of terms and concepts along with the underlying domain knowledge required to parse and interpret the meaning of a phrase or sentence.

As such, a series of initiatives were formed by a consortium of biomedical organizations (BioCreative, BioNLP, ShARe/CLEF eHealth, i2b2, and SemEval) with the intent of bringing attention to this unexplored domain for NLP. These initiatives focused on formalizing entity recognition and relation extraction tasks within biomedical literature. To bolster research interest in the pharmacovigilance domain, a challenge was announced for detecting DDI relations from biomedical literature. The SemEval DDI 2011 (Segura-

Bedmar, Martinez, & Sánchez Cisneros, 2011) and SemEval DDI 2013 (Segura-Bedmar, Martínez, & Herrero-Zazo, 2013) challenges feature an expert curated DDI corpus (Herrero-Zazo, Segura-Bedmar, Martínez, & Declerck, 2013) with a gold standard labeling sourced from Medline, a collection of over 26 million scientific journal articles, and DrugBank (Wishart, Knox, Guo, Cheng, 2008), a database of drug descriptions. The selection of these two separate sources of literature articles was intended to provide representation for different styles of writing (Herrero-Zazo et al, 2013). According to Herrero-Zazo et al (2013), DrugBank text is curated to have short and concise descriptions about drugs, whereas Medline articles tend to have long complex sentences with more scientific terminology.

The DDI corpus was prepared using an XML format that includes a sentence from DrugBank or Medline article along with the labeled entities and labeled drug pairs with the corresponding DDI relation classification. The corpus was human annotated by subject matter experts and used an Inter-annotator agreement (IAA) score to ensure quality and consistency across the human labels (see Figure 2). The combined corpus includes 27,784 training samples and 5,716 test samples

	DDI-DrugBank	DDI-MedLine
K_{EFFECT}	0.7525	0.5548
$K_{MECHANISM}$	0.4214	0.5577
K_{ADVICE}	0.9428	0.5587
K_{INT}	0.9558	0.7252
K	0.8385	0.6213

Figure 2 - IAA scores by relation type for SemEval DDI 2013 dataset (Segura-Bedmar et al, 2013)

A total of 10 teams participated in the SemEval DDI 2011 challenge and 14 teams participated in the SemEval DDI 2013 challenge. The challenge featured two tasks, the first was a drug entity detection and the second a DDI relation extraction task. Teams were provided training data that could be used to train an ML model and submitted 3 separate output runs using a validation test set. Teams could choose to participate in the entity detection and/or the relation extraction task. During the SemEval DDI 2011 challenge, the DDI relation dataset was labeled as a binary classification task where each sentence and

pair of drug candidates were labeled as true, a DDI was identified, or false, no DDI was identified.

During the SemEval DDI 2013 challenge, the dataset was updated to include a relation classification task that included labels for each type of DDI. The DDI class types are defined as follows:

- Mechanism – a pharmacokinetic drug interaction
- Effect – a pharmacodynamic drug interaction
- Advice – a recommendation given about a drug interaction
- Int – an unspecified interaction type

Figure 3 shows several sample sentences taken from Medline articles and DrugBank descriptions. The first sentence (A) shows an example of a *mechanism* DDI between 4-methylpyrazole and 1,3-difluoro-2-propanol. The second sentence (B) and the first line in (C) show an example of an *effect* DDI between estradiol and endotoxin and between the enumerated set of drugs starting with CNS depressant drugs and Inapsine. The second line in (C) shows an example of *advice* DDI between Inapsine and CNS depressant drugs.

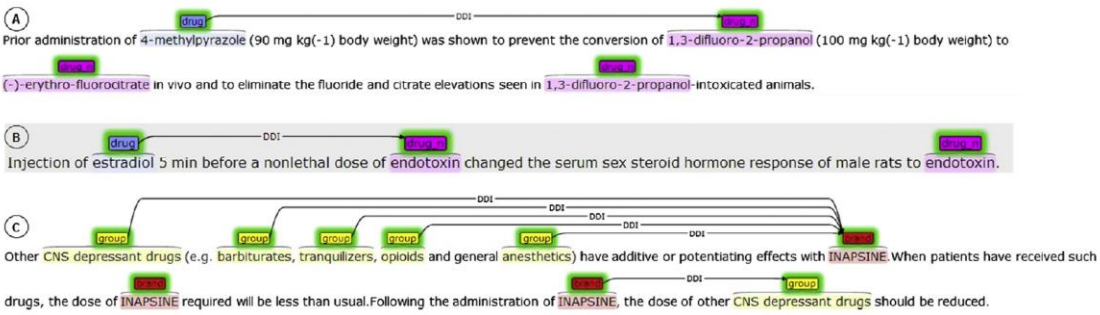

Figure 3 - Example of DDI interaction types; (A) mechanism, (B) effect, (C) first line: effect and second line: advice

Early DDI Relation Extraction Methods

The first known method to detect DDI through binary classification over biomedical literature was presented by Segura-Bedmar, Martinez, and Pablo-Sanchez (2011) using a kernel-based Support Vector Machine (SVM) (Cortes & Vapnik, 1995) classifier. Their SVM classifier utilized a shallow linguistic parse kernel to capture syntactic feature patterns from a sentence featuring one or more drug entities. They used a

pre-processing pipeline that leveraged UMLS (Bodenreider, 2004) metathesaurus for identifying drug types mentioned in the input text. Their method achieved an f1 measure of .6001 on what would later become the SemEval 2011 DDI benchmark. Segura-Bedmar et al (2011)'s contribution was significant, establishing the first baseline for DDI relation extraction and formalizing what would become a standard benchmark for relation classification (Segura-Bedmar et al, 2011; Segura-Bedmar et al, 2014). Additionally, their work also established the significance of DDI and helped attract research interest into biomedical literature.

During the first SemEval DDI 2011 challenge, a total of 10 teams participated. Each team was provided the DDI training set that included sentences with drug pair candidates along with a binary classification label of true or false. The winner of the challenge was (Thomas, Neves, Solt, Tikk, & Leser, 2011) with a DDI detection f1 performance of .6574. All participants utilized an SVM classifier approach that either used feature-based or kernel-based methods. Thomas et al (2011) used multiple lexical and syntactic kernels with a *majority vote* scheme that outperformed the rest of the participants (Chowdhury, Abacha, Lavelli, & Zweigenbaum, 2011; Chowdhury & Lavelli, 2011; Bjorne, Airola, Pahikkala, & Salakoski, 2011).

In (Segura-Bedmar et al, 2011) a meta-analysis is presented of the SemEval DDI 2011 participants methods. This analysis found that kernel-based methods outperformed feature-based methods and that most methods relied on syntactic information, but no method made use of semantic information. Given level of interest and participation of this challenge, Segura-Bedmar et al (2011) announced a second DDI challenge featuring a larger dataset and a multiclass classification task DDI relation extraction.

During the SemEval DDI 2013 challenge, a total of 8 teams participated in the DDI Relation Extraction task. Each team submitted 3 runs that were then evaluated by the challenge judges (Segura-Bedmar et al, 2014). The winning team for this challenge was Chowdhury & Lavelli (2013) with the overall best f-score of .8 for the Relation Detection task and .65 for the Relation Classification task. Their model, called *FBK-irst*, employed a two-stage model that used a binary SVM model for relation detection to separate positive DDI from negative DDI. This stage was trained using linguistic features such as negation

triggers and shallow parse of the sentence. They also employed several filtering heuristics, known as "negative instance filtering" that looked for instances where both drug pairs were the same drug, when drugs were listed consecutively such as in a coordinate conjunction list (such as a list of drugs separated by comma ',', 'and', or 'or'). The first heuristic eliminates the case where a drug does not interact with itself. The second eliminates the case where a brand or generic drug is immediately followed by a synonym of the drug such as: "Advil (Ibuprofen)". In this case, the second mention is simply an alternate variant and therefore can be excluded.

Chowdhury & Lavelli (2013) used a second stage hybrid SVM that combined feature-based kernel with a shallow linguistic kernel and path encoded tree kernel and then a second stage that classified positive DDI into one of the 4 DDI types. This model made use of several tools including the Stanford Parser (De Marneffe, & Manning, 2008) and Charniak-Johnson reranking parser (Charniak & Johnson, 2005) tuned for biomedical text.

All participants utilized SVM for their learning algorithm, half utilized feature-based and the other half used kernel-based models. Overall, the kernel-based models outperformed feature-based model. Most participants utilized Stanford Parser and Charniak-Johnson Reranking parser for syntactic features, but only Chowdhury & Lavelli (2013) utilized negation triggers. *Figure 4* and *Figure 5* shows the ranked order of participants for both the Relation Detection and Relation Classification tasks where Chowdhury & Lavelli's *FBK-irst* model outperformed the other models on both tasks (Segura-Bedmar et al, 2014).

Team	Run	DrugBank				MedLine				Overall			
		Rank	*P*	*R*	*F1*	Rank	*P*	*R*	*F1*	Rank	*P*	*R*	*F1*
FBK-irst	1	1	0.816	0.838	0.827	1	0.558	0.505	0.53	1	0.794	0.806	0.8
	2	2	0.816	0.838	0.827	2	0.558	0.505	0.53	2	0.794	0.806	0.8
	3	3	0.816	0.838	0.827	3	0.558	0.505	0.53	3	0.794	0.806	0.8
WBI-DDI	1	6	0.857	0.686	0.762	8	0.63	0.358	0.456	6	0.841	0.654	0.736
	2	5	0.874	0.696	0.775	12	0.651	0.295	0.406	5	0.861	0.657	0.745
	3	4	0.814	0.755	0.783	4	0.625	0.421	0.503	4	0.801	0.722	0.759
UTurku	1	10	0.846	0.614	0.712	20	0.724	0.221	0.339	11	0.841	0.576	0.684
	2	9	0.861	0.624	0.724	19	0.778	0.221	0.344	8	0.858	0.585	0.696
	3	8	0.843	0.638	0.726	15	0.658	0.263	0.376	9	0.833	0.602	0.699
SCAI	1	11	0.836	0.619	0.711	7	0.688	0.347	0.462	10	0.826	0.592	0.69
	2	12	0.837	0.617	0.71	17	0.686	0.253	0.369	12	0.829	0.581	0.683
	3	7	0.796	0.681	0.734	6	0.431	0.526	0.474	7	0.748	0.666	0.704
UC3M	1	13	0.656	0.758	0.703	13	0.392	0.421	0.406	13	0.632	0.725	0.676
	2	19	0.415	0.814	0.549	10	0.313	0.642	0.421	19	0.404	0.798	0.537
NIL_UCM	1	16	0.615	0.615	0.615	22	0.419	0.137	0.206	16	0.608	0.569	0.588
	2	14	0.673	0.688	0.68	21	0.548	0.242	0.336	14	0.667	0.645	0.656
UWM_TRIADS	1	17	0.525	0.689	0.596	11	0.415	0.424	0.419	17	0.517	0.664	0.581
	2	15	0.573	0.665	0.616	9	0.427	0.446	0.436	15	0.561	0.644	0.599
	3	18	0.465	0.746	0.573	5	0.387	0.63	0.479	18	0.458	0.735	0.564
UColorado_SOM	1	22	0.387	0.739	0.508	16	0.256	0.663	0.37	21	0.37	0.731	0.492
	2	20	0.391	0.765	0.518	14	0.28	0.663	0.394	22	0.378	0.755	0.504
	3	21	0.422	0.646	0.511	18	0.253	0.6	0.356	23	0.398	0.641	0.491

Figure 4 – Table showing ranked order of the SemEval 2013 challenge for Relation Detection task (Segura-Bedmar et al, 2014)

Team	Run	DrugBank				MedLine				Overall			
		Rank	*P*	*R*	*F1*	Rank	*P*	*R*	*F1*	Rank	*P*	*R*	*F1*
FBK-irst	1	3	0.654	0.672	0.663	4	0.407	0.368	0.387	3	0.633	0.642	0.638
	2	1	0.667	0.686	0.676	2	0.419	0.379	0.398	1	0.646	0.656	0.651
	3	2	0.664	0.682	0.673	3	0.419	0.379	0.398	2	0.643	0.653	0.648
WBI-DDI	1	6	0.702	0.561	0.624	7	0.463	0.263	0.336	6	0.685	0.532	0.599
	2	5	0.707	0.563	0.627	12	0.488	0.221	0.304	5	0.695	0.53	0.601
	3	4	0.657	0.609	0.632	5	0.453	0.305	0.365	4	0.642	0.579	0.609
UTurku	1	9	0.723	0.525	0.608	18	0.517	0.158	0.242	9	0.714	0.489	0.581
	2	7	0.738	0.535	0.62	16	0.593	0.168	0.262	7	0.732	0.499	0.594
	3	8	0.706	0.534	0.608	13	0.5	0.2	0.286	8	0.694	0.502	0.582
NIL_UCM	1	12	0.54	0.541	0.54	22	0.387	0.126	0.19	12	0.535	0.501	0.517
	2	10	0.566	0.579	0.573	19	0.357	0.158	0.219	10	0.557	0.538	0.548
UC3M	1	11	0.518	0.598	0.555	15	0.265	0.284	0.274	11	0.495	0.568	0.529
	2	21	0.231	0.454	0.306	21	0.138	0.284	0.186	21	0.222	0.437	0.294
SCAI	1	15	0.546	0.404	0.464	1	0.625	0.316	0.42	14	0.551	0.395	0.46
	2	16	0.545	0.402	0.463	8	0.6	0.221	0.323	16	0.548	0.384	0.452
	3	14	0.513	0.439	0.473	6	0.31	0.379	0.341	15	0.486	0.433	0.458
UWM_TRIADS	1	17	0.407	0.534	0.462	10	0.309	0.315	0.312	17	0.4	0.513	0.449
	2	13	0.452	0.524	0.485	9	0.312	0.326	0.319	13	0.439	0.505	0.47
	3	18	0.361	0.579	0.445	11	0.247	0.402	0.306	18	0.35	0.562	0.432
UColorado_SOM	1	22	0.166	0.317	0.218	20	0.13	0.337	0.188	22	0.161	0.319	0.214
	2	20	0.258	0.503	0.341	14	0.196	0.463	0.275	20	0.25	0.499	0.334
	3	19	0.288	0.441	0.349	17	0.173	0.411	0.244	19	0.272	0.438	0.336

Figure 5 - Table showing the ranking for SemEval 2013 DDI Relation Classification task (Segura-Bedmar et al, 2013)

Because SVM is a binary classifier, all participants used a “one against all” SVM classifier approach where one classifier is trained per class label (He, Yang, Zhao, Lin, & Li, 2013; Chowdhury et al, 2013; Segura-Bedmar et al, 2014). For the DDI classification task, there are 5 possible labels (i.e. Advice, Mechanism, Effect, Int, and No DDI) and therefore 5 classifiers are trained as an ensemble of binary classifiers.

In Segura-Bedmar et al’s (2014), a meta-analysis of the results of the challenge is presented. The report finds that all participants used feature-based or kernel-based SVM.

The kernel-based methods outperformed the feature-based methods overall. Additionally, the report states that all models performed better on the DrugBank source text samples of the dataset compared to the Medline text samples. (Sergura-Bedmar et al, 2014)'s analysis finds that Medline text tends to have long complex sentences that feature scientific language when compared to DrugBank sentences that have shorter and concise sentences. In their error analysis, they determined that the root cause for most of the false-negative misclassifications were due to lack of cataphora and anaphora resolution. They also identify additional causes of misclassification due to lack of detection for coordinate structure (i.e. conjunction lists of drugs) and the variability in expressing DDIs with different lexical and syntactic expressions. Additionally, most methods performed poorly in samples that leveraged domain terminology to implicitly indicate the presence of a DDI relation. They conclude that the primary reason for (Chowdhury et al, 2013) top performance in the challenge was attributed to their use of negation triggers and their negative instance filtering. Again in this challenge, Segura-Bedmar et al (2014), note that no use of semantic information such as ontologies were used by the participants and suggest this as an area worthy of future exploration to address some of the common misclassification causes. The only biomedical domain-specific resource leveraged was an extension to the Charniak-Johnson reranking parser used by the top performing models. This parser takes the output of a syntactic parse and applies additional reranking heuristics that were tuned for a biomedical corpus. Lastly, Segura-Bedmar et al (2014) attempt to construct an ensemble classifier using various combination of top performing models for the DDI challenge but were unable to beat the top performing model.

Following the success of the two SemEval DDI challenges, the DDI relation extraction task has become a standard benchmark for assessing and comparing relation extraction methods. (Lamurias, Ferreira, & Couto, 2014) presents a 2-stage kernel-based SVM with an Ensemble approach building upon the previous model presented in (Chowdhury et al, 2013) and extending it with an entity detection module that uses ChEBI ontology. Their model, however, fell short of matching the state-of-the-art model by Chowdhury et al, 2013 with an f1 score of .6402 (vs .651).

(Kim, Liu, Yeganova, & Wilbur, 2015) demonstrate that a linear kernel-based SVM model using lexical positional features can yield competitive results when compared to the

popular non-linear kernel based methods used by most-all participants in the DDI challenges. Their interest in linear kernels stem from their computational efficiency compared to non-linear. Their results demonstrated the best f1 performance against the DDI benchmark surpassing (Chowdhury et al, 2013) with a score of .67 (vs .651).

(Zheng, Lin, Zhao, Xu, Zhang, Yang & Wang, 2016) proposed a context vector method using a graph-kernel. Their context vector served as a representation of the sentence, also known as a summary vector. This approach of levering a summary vector for down-stream classification would later be incorporated as an essential aspect of modern neural classifiers. Their work demonstrated a new high f1 score of .684 on the DDI relation classification task. They also demonstrated a new best f1 score of .818 for the DDI detection task surpassing (Chowdhury et al, 2013)'s detection model for the first time.

Resurgence of Neural Networks

The era of Deep Neural Networks (DNN) began after several significant developments in image processing and handwriting recognition during the 2010's. In their classic textbook on Deep Learning Bengio, Goodfellow, & Courville (2017) explain a key motivation for shifting away from SVM to NN involved the need for large-scale training data needed for achieving better performance on a range of tasks, such as image processing. The scalability limitations of SVM kernels due were due to the computational complexity caused by the curse of dimensionality problem.

In 2006, Chellapilla, Puri, and Simard demonstrated an implementation of a Convolutional Neural Network (CNN) that could be trained 60 times faster than using a conventional CPU. Ciresan, Meier, Gambardella, & Schmidhuber (2011) demonstrated human-level performance on image recognition tasks such as MNIST and CIFAR10. Krizhevsky, Sutskever, & Hinton (2012) wins the ImageNet challenge followed by a Microsoft team in 2015 featuring a CNN with 100 layers.

These successive achievements have positioned Deep Learning and Neural Networks (NN) at the forefront of research ensuing in the exploration of Deep Learning methods into other fields including NLP tasks. NN are a branch of Machine Learning that utilize the concept of a neuron that has an incoming and outgoing connection to other neurons collectively forming a neural network. The first layer of neurons in a NN is known

as the input layer. The input to the first layer is in the form of a vector that represents the feature space. The network can have one or more layers, known as hidden layers, that can be connected using different network architectures. The most basic neural network is known as a Perceptron (Rosenblatt, 1960) where a single layer is used in the network. These neural networks can grow with multiple layers known as Multi-Layer Perceptron (MLP) or Feed-forward networks. As the number of layers increase, the total number of neurons (i.e. parameters) of the model increases. This increases the depth of computation required to evaluate a given input vector. Recent advances in NN include Convolutional Neural Networks (CNN) (LeCun, 1989) which apply a kernel (filter) to a sweeping window generating a new representation of the feature space for the next layer to process. CNN are considered the state-of-the-art for tackling several types of classification tasks such as image classification, among others. Each of these neurons in the network apply a non-linear activation function, such as Rectified Linear Unit (ReLU) (Jarrett et al., 2009; Nair and Hinton, 2010; Glorot et al., 2011a), Hyperbolic Tangent (tanh), or Sigmoid functions, to an incoming value and update its weight and propagates the value forward to the next layer. Using a method called Back-Propagation (Rumelhart et al., 1986a), a neural network can be trained by updating the weights of each neuron by minimizing a loss function using optimization algorithms such as Stochastic Gradient Descent (SGD) (Yuan & Yu, 2012).

Neural Network Feature Representation

One attractive aspect of DNN involve their ability to perform iterative feature extraction from layer to layer within the network starting with latent features presented to the input layer. For many tasks, such as image classification, the data is already represented as scalar values. These values are typically normalized to conform to the restricted range of activation functions. The two most common intervals are real values between -1 and +1 or 0 and +1. Once the vectorization of the original input is encoded and normalized, it is presented as vectors to the input layer of the NN.

When applying NN to NLP tasks, additional encoding is required to convert words, symbols, and/or sentences into feature vectors. This encoding process is generally referred to as text representation, a form of feature vectorization, and an active area of research. There are exist various methods to encode text into a feature vector including one-hot

encoding, integer encoding, bag-of-words or n-grams. These feature vectors can also represent positional features of the words, such as word-distances or frequency counts, such as Term Frequency (TF) and/or IDF (Inverse Document Frequency) that represent the popularity of the word within a corpus or document.

Representation Learning and Word Embeddings

The study of Representation Learning has contributed to significant performance improvements of NN models trained on NLP tasks. Representation Learning is a semi-supervised learning process that learns the distributional representations of words used within the training corpus. Words sharing similar context are clustered closer together than words with different context. This is known as the Distributional Hypothesis (Young et al, 2019). In their work with Neural Language Models, (Bengio, Ducharme, Vincent, and Jauvin, 2003) proposed the word embeddings as a dimensionality reduction of a high-dimensional and sparse feature space to a low-dimensional and dense embedding space. Salakhutdinov & Hinton (2006) presented a view of NN as a series of representation learning layers (i.e. the hidden layers) followed by a final classification layer. Conceptually these hidden layers represent an embedding that can be used to project the input feature space into the target embedded feature space. Several important improvements in neural word embedding by (Ronan & Jason, 2008) on pre-trained embeddings and by (Mikolov et al, 2013) presented two efficient Neural Embedding algorithms called Skip-gram Negative Sampling (SGNS) and Continuous Bag-of-words (CBOW). The former demonstrated higher accuracy. The use of Mikolov et al's word2vec tool is pervasive in the field of NLP. Other word embedding models have been proposed to address various limitations such as GloVe (Pennington, 2014), FastText (Joulin, Grave, Bojanowski, Douze, Jégou, & Mikolov, 2016) and (Hasimoto et al, 2014).

Word embedding have been essential in NLP models but have several limitations. The biggest limitation involves the out-of-vocab (OOV) problem. This occurs when a word is not found within the embedding space. Since all OOV are mapped to a constant vector representation (such as all zeros) the model is unable to learn from those words. Another issue deals with the problem of word sense disambiguation. Some words are polysemy and require context to derive its meaning. Unfortunately, the embedding will map the word

regardless of context into the same vector representation. This can also confuse the learning of a model in fields where the same word is used with different meanings. Another issue with word embedding involves the lack of normalization for semantical similar but morphologically different words. Recent word embedding models attempt to resolve some of these issues by embedding at the sub-word level to account for morphology (Joulin et al, 2016). Word embeddings and language models is an important and active area of research. Further discussion into word embeddings is considered out of scope for this review.

Early Neural Relation Classification

One of the earliest applications of NN in relation classification was presented by Socher, Huval, Manning, and Ng (2012). Their work featured a recursive neural network trained to classify semantic relations in text (Hendrickx, Kim, Kozareva, … et al, 2019). Socher et al (2012) presented a novel sentence vectorization encoding based on their observation that single word vectors do not capture the meaning of a sentence. Socher et al (2012) refer to (Frege, 1892)'s observation that in natural language, meaning is conveyed through both the individual meaning of words and their arrangement within a linguistic structure. In (Socher et al, 2012) they define a matrix-vector to present the meaning of a sentence. This approach relied on the Stanford Parser to produce a syntactic parse of a sentence and then a recursive traversal of the dependency paths between the two candidate entities of a relation. Their method was evaluated on an open-domain corpus using the SemEval 2010 (Hendrickx et al, 2019) and achieved 2nd highest performance using only a sentence matrix-vector as the model input feature. They achieved the highest performance when they additionally included syntactic features along with their sentence matrix-vector.

Applying CNN on relation extraction tasks began with (Zeng, Liu, Lai, Zhou, & Zhao, 2014) which included positional and syntactic features such as Shortest Dependency Path (SDP) between candidate entities, a method used by early relation classifiers (Bunescu and Mooney, 2005). Their work demonstrated new best f1 performance on SemEval 2010 open-domain semantic relation task. Santos, Xiang, & Zhou (2015) and Wang, Cao, De Melo, & Liu (2016) each achieved new state-of-the-art performance on the SemEval 2010 benchmark by building upon the work of (Zeng et al 2014). (Wang et al, 2016)

demonstrated the performance impact of leveraging an input Attention mechanism (Bahdanau, Chorowski, Serdyuk, Brakel, & Bengio, 2016).

Early Neural Relation Classification on DDI Benchmark

Research into applying CNN-based models on the SemEval 2013 DDI benchmark was pioneered by (Liu, Tan, Chen, & Wang, 2016) and demonstrated a new best f1 of .6975 (compared to the previous high of .684 that used graph-kernel-based method). Liu et al (2016) trained a word embedding using the Medline biomedical corpus featuring a vocabulary of 1.99 million medical words. Liu et al (2016) argued that a competitive performance on relation tasks could be achieved without relying on special NLP tools to extract syntactic features. *Figure 6*, depicts the CNN architecture proposed by Liu et al showing word and position embeddings and the well-known pattern of convolutional layer followed by max pooling for down-sampling and a softmax classifier as the output layer.

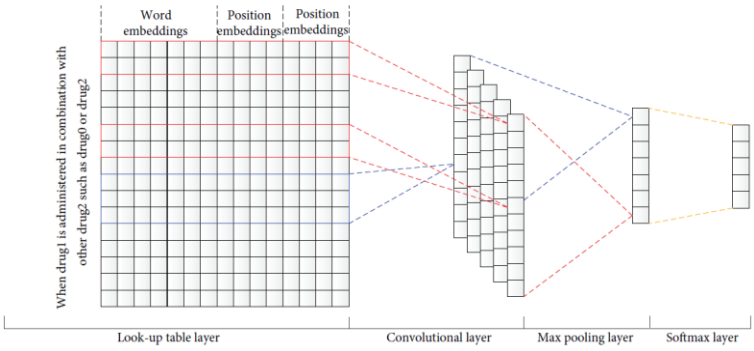

Figure 6 - CNN DDI relation model with word embeddings (Liu et al, 2016)

Although Liu et al (2016) did not directly incorporate any features derived from an NLP tool, their pre-processing did make use of NLTK tool for tokenizing of the sentence.

The convolutional sweep over the word vectors is conceptually equivalent to extracting n-grams. According to the kernel size (i.e. window size) of the convolutional layer, the length of the n-gram will vary. This has the effect of weighing the importance of an n-gram of words with respect to the relation type. This approach is intuitive and resembles techniques used by early statistical learning methods of relation extraction.

Quan, Hua, Sun, & Bai (2016) expand the work of (Liu et al, 2016) by introducing the concept of *multiple embedded channels* for relation extraction models where each channel of input is embedded differently. Their model uses a total of 5 input channels where

each channel embeds the words in the input text using a different word embedding trained on different corpus. Their approach makes use of 4 word vector weight matrix produced by Pyysalo, Ginter, Moen, Salakoski, & Ananiadou (2013) on PubMed, PubMed Central, Medline, and Wikipedia using Mikolov et al's (2013) Skip-gram with Negative Sampling (SGNS) algorithm and word2vec tool for loading word vectors. Quan et al (2016) also trained their own word embedding using (Mikolov et al, 2013) Continuous Bag of Words (CBOW) algorithm over the Medline corpus. Quan et al's model achieved an f1 of .702 on the SemEval 2013 DDI benchmark establishing the new state-of-the-art performance on the benchmark. Their results supported Liu et al (2016)'s observation that high-performance could be achieved using only word embedding feature representations without reliance on additional syntactic information. *Figure 7* depicts the architecture proposed by Quan et al (2016). Their model included two convolutional layers that used different kernel sizes of 3 (red) and 4 (yellow). As previously mentioned, this is equivalent to considering 3-gram and 4-gram phrases. The rest of the model follows Liu et al (2016) method using a Max Pooling and Softmax Classifier output layer.

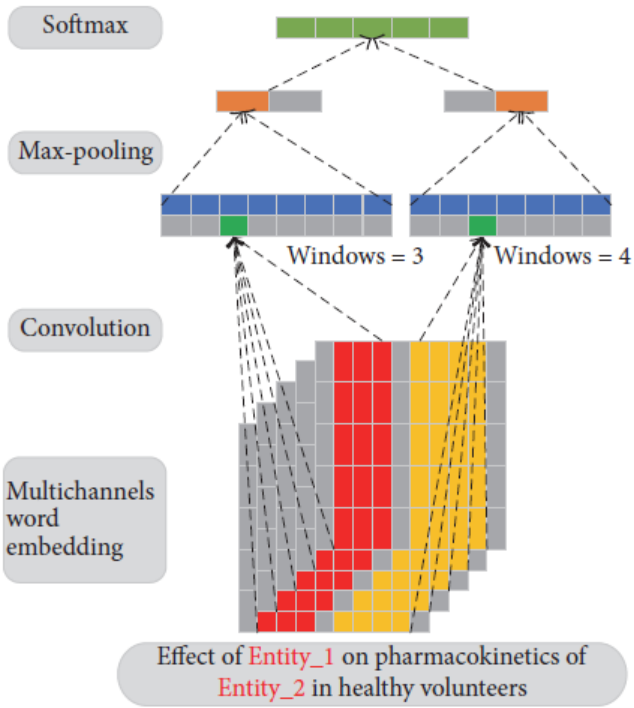

Figure 7 - Quan et al (2016) architecture featuring multiple embedding channels

Zhao, Yang, Lou, Lin, and Wang (2016) propose a CNN model that features a

syntactic word features that are embedded using (Mikolov et al, 2013)'s word2vec tool. This embedding differs from (Quan et al, 2016) and (Liu et al, 2016) because it uses a syntactic parser to produce the part-of-speech tags and dependency path between pairs of drug candidates. These feature vectors are concatenated and presented to the model as a single channel. Their model achieves a high f1 of .686 on the SemEval DDI 2013 benchmark (.0115 less than Liu et al, 2016 and .016 less than Quan et al, 2016). Zhao et al (2016) claims to achieve the highest f1 on the DDI benchmark by comparing their model to the pre-neural model by (Zheng et al, 2016). Zhao et al (2016) was likely unaware of (Liu et al, 2016) and (Quan et al, 2016) work.

Recurrent Neural Networks

The application of CNN models on NLP tasks, such as relation extraction tasks, failed to demonstrate the same kind of performance impact that they had in the image processing and computer vision field. The f1 performance improvement of CNN on the SemEval 2013 DDI benchmark added .018 to the previous highest kernel-based methods. Unlike images, natural language is best modeled as a sequence problem. CNN work well because images can be represented as a fixed vector size where only the spatial ordering of the features are important to understand textures and patterns found in the rendering of the image. In natural language, the ordering of the words has major implication on the meaning being conveyed by a sentence. Because a CNN does not model time dimension, all the words are presented at once like a bag of words. The ordering of the words has no impact to how the model classifiers.

Recurrent Neural Network (RNN) (Rumelhart et al, 1986a) is a family of NN that are designed to model time-series (or sequence) problems. The neural unit (cell) of an RNN can be unfolded to reveal hidden layers that preserves a memory of the weights from a previous time-step. The output weight from the current time-step is a function of the weight of the current activation and the weight from the previous time-step. This design allows RNN to remember and adjust their classification as a sequence is processed. Intuitively, this memory element of an RNN resembles how a human may process information taking into consideration the information that was previously seen.

In practice, RNN suffer from various drawbacks including vanishing and exploding

gradients problem. The vanishing gradient problem occurs when the gradient (i.e. error) becomes increasingly smaller until it vanishes as Back-Propagation Through Time (BPTT) is applied to each of the hidden recurrent layers. This restricts the practical memory range that a RNN can model. To resolve this issue, the Long-Short Term Memory (LSTM) unit was proposed by Hochreiter and Schmidhuber (1997) with the goal of extending the range of memory for an RNN. The LSTM is a gated unit that is self-looped to control input, output, and forget gate. *Figure 8* shows a depiction of the internal wiring of an LSTM unit with its input, output, and forget gates along with the standard input and output connections.

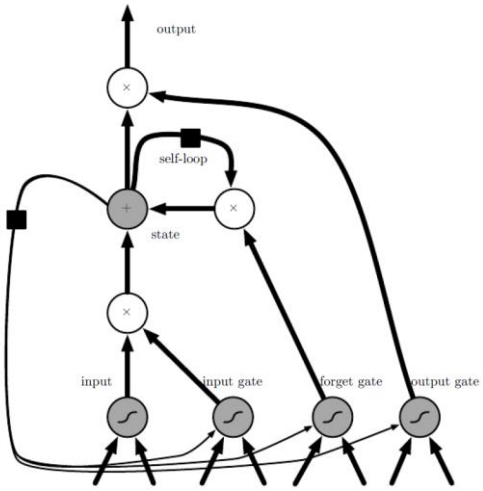

Figure 8 - block diagram of an LSTM unit (Goodfellow et al, 2017)

Recent Neural Relation Classification

An early application of RNN/LSTM to relation extraction tasks was proposed by Xu, Mou, Li, Chen, Peng, and Jin (2015). Their proposed model utilized 4 input channels that included: word embedding, part-of-speech, Shortest Dependency Path (SDP), and WordNet (Lexical classes). They evaluated their model using the SemEval 2010 benchmark, achieving the best f1 and outperforming the previous state-of-the-art model by (Santos et al 2015). Their model incorporated elements that demonstrated performance impact in previous work such as the usage of SDP (Bunescu and Mooney, 2005) along with word embeddings trained on a medical corpus. Their propose a split feature representation of the SDP by splitting the two paths between the pair of drug candidates as

separate feature vectors. *Figure 9* shows an example of the SDP path between a pair of entities. Each path between the entity and the root is represented in a separate feature vector.

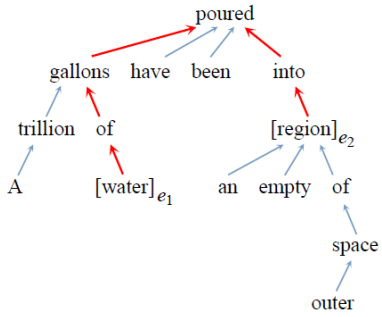

Figure 9 - an example of the SDP between two entities: water and region (Xu et al, 2015)

Xu et al also extract the WordNet lexical classes (hypernyms) using the Super-Sense Tagging (SST) tool from (Ciaramita & Johnson, 2003) from the words represented in the SDP producing an equivalent vector representation embedded using the 45 WordNet classes. *Figure 10* depicts the architecture proposed by (Xu et al, 2015) showing 4 different input embedding channels each with an LSTM layer that is then down-sampled by a pooling layer, a hidden fully-connected (dense) layer, and a Softmax classifier. This model serves as a reference and a baseline in the work of several models that build upon and incorporate many of these elements with some modifications.

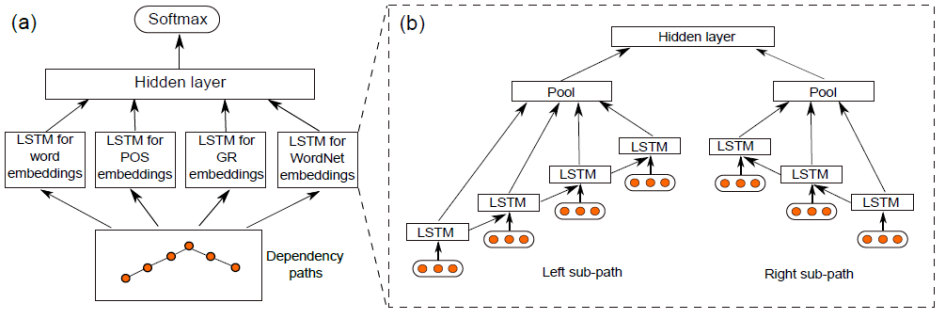

Figure 10 - The architecture for Xu et al (2015) relation classifier

In (Wang, Yang X., Yang C., Guo, Zhang, & Wu, 2017) they propose a Bidirectional LSTM that experiments with different vector sequence ordering based on the dependency parse tree of the sentence. Bidirectional LSTM (or BiLSTM) are equivalent to

applying the LSTM layer to the normally ordered input sequence and the reverse order of the input sequence. In their work, present a model for vector ordering using a depth-first traversal and a breadth-first traversal of the tree nodes where each node represents a word that is embedded using word2vec. They also demonstrate that using an Averaging of the left and right LSTM outperforms using the classical concatenation of the left and right resulting output vector. They also note that the averaging is more computationally efficient since it does not double to dimensionality width of the resulting vector. Their approach demonstrates a new best f1 score for the SemEval 2013 DDI benchmark of .72 (compared to the previous high of .702 by Quan et al, 2016).

In (Zheng, Lin, Luo, Zhao, Li, Zhang, … & Wang, 2017), they propose a BiLSTM model that incorporates an input Attention layer (Bahdanau et al, 2016). Their approach utilizes 3 input channels representing the word, part-of-speech, and position within the input text. Instead of treating each word vector with equal importance, the Attention mechanism applies an importance weight to the features. Bahdanau et al (2016) observed in their Neural Translation research that certain words were more important than others and should therefore be given additional importance (weight). The same could be said about relation classification problems such as the DDI extraction task. Certain words are more indicative of expressing an interaction and a layer (i.e. Attention layer) can be trained to apply weight to those words. Zheng et al (2017) demonstrated a major improvement in f1 on the SemEval 2013 DDI benchmark with a .773 on the classification and a .84 on the detection task. Both these scores place it at the all-time highest f1 metrics reported to-date on this benchmark. Their work built upon various elements of prior research on this task employing BiLSTM with multiple input channels representing word, positional, and syntactic features. Their work also demonstrated the benefits of using an Attention layer to apply importance to predictive words or features. Figure 11 depicts the architecture used.

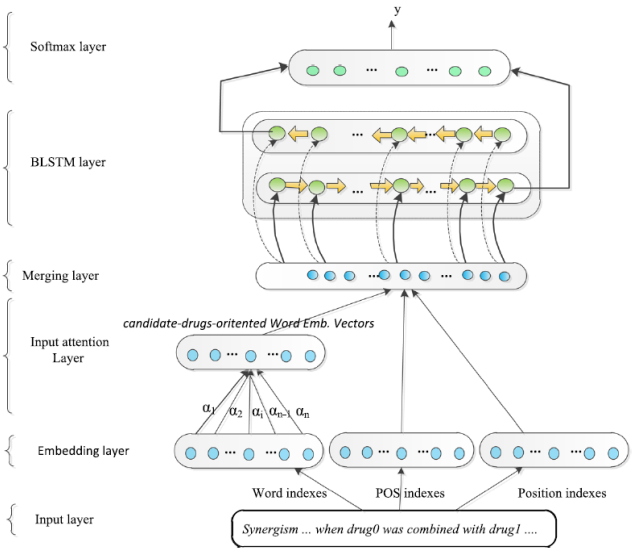

Figure 11 - Zheng et al (2017) BiLSTM architecture with Attention layer

Additional work to improve upon the results of Zheng et al (2017) has continued but thus far failed to exceed the benchmark performance demonstrated by their model. In (Xu, Shi, Zhao, & Zheng, 2018), they propose the use of a concept embedding scheme using the MetaMap tool to perform a concept matching from words in the sample text to UMLS concepts. They demonstrate a competitive f1 score of .7115 but fall well short of the .773 produced by (Zheng et al, 2017).

In (Kavuluru, Rios, & Tran, 2017) they propose the inclusion of a character based BiLSTM in addition to the conventional word-based BiLSTM used by all recent proposed models for the DDI benchmark. Their work did not include a specific quantitative or qualitative analysis of their performance against the benchmark but reported an f1 of .7213 on the classification task using an ensemble of 20 models all trained using the same architecture. They argue that the random parameter initialization could position a specific model instance in favorable global minima that may not be reproducible. They encourage future work to include average model performance over multiple training instances in addition to the best f1 score.

In (Zhang, Zheng, Lin, Wang, Yang, & Dumontier, 2018), they present two stacked (hierarchical) BiLSTM layers with input Attention layer. They present a split model where

the input sequence is embedded and projected through an Attention layer and then divided into 3 subsequences. The subsequence is defined as the sequence of words prior to the first drug candidate, the second sequence are the words between the first and second drug candidate, and the third sequence are the words that follow the second drug candidate. Each of these subsequences are sent to a separate BiLSTM layer. Additionally, a separate SDP sequence of the sentence is embedded and projected through a separate Attention layer and mapped to a fourth BiLSTM layer. Each of those 4 BiLSTM layers serve as inputs to a final hidden BiLSTM layer before the Softmax classification. This approach demonstrated a competitive f1 of .729 on the SemEval DDI 2013 benchmark falling short of the best recorded performance (.773 by Zheng et al, 2017).

In (Sahu & Anand, 2018), they present a Dual BiLSTM architecture with Max pooling and Attentive Pooling layer. Their approach does not rely on any NLP tools or syntactic features, instead their features include word, positional distance from drug1, and positional distance from drug2. Their experiments evaluated a Max Pooling vs Attentive Pooling vs both pooling types. Their results found that having both a Max Pooling layer and an Attentive Pooling layer that converge into a single dense (fully connected) layer yielded the best results. Their findings indicate that applying an Attention alone is not enough to produce the best results. Although their dual BiLSTM model approach with an f1 of .6939 on SemEval DDI 2013 benchmark failed to surpass Zheng et al (2017) they demonstrated that a competitive performance could be reached without sophisticated feature engineering that includes syntactic features such as the part-of-speech information leveraged by (Zheng et al, 2017).

In (Lim, Lee, & Kang, 2018), they revisit the method proposed in (Socher et al, 2005) by leveraging a recursive NN design that features a tree-LSTM architecture based on (Tai KS., Socher R., Manning, 2017). Their model did not make use of any engineered features such as syntactic or semantic information and instead relied on word and positional features. Using the SemEval 2013 benchmark, they achieved an f1 of .735 on the classification task and .838 on detection task. Their results rank their model among the top models evaluated but falls short of surpassing the results by Zheng et al (2017). Their approach departs from the trending BiLSTM with pooling architecture trend in recent work

and instead revisits prior work with recursive NN structures that leverage LSTM units. See Figure 12 for a depiction of the tree-LSTM architecture.

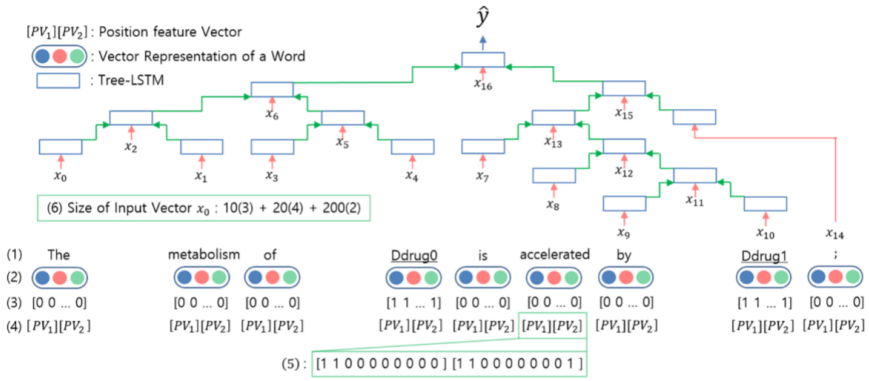

Figure 12 - Recursive tree-LSTM architecture (Lim et al, 2018)

Most recently, (Lamurias, Sousa, Clarke, & Couto, 2019) proposed a model that featured an ontology embedding layer using the ChEBI ontology. Their model is an extension of (Zhang et al, 2018) which uses a BiLSTM with Attention and additionally incorporates elements from prior work such as the use of external biomedical knowledge (Xu et al, 2018), multiple input channels (Quan et al, 2016), two-path SDP feature representation (Xu et al, 2015), and negative instance filtering (Chowdhury & Lavelli, 2013). They propose their own concept-matching algorithm for looking up concepts within the ChEBI using the canonical form and synonyms over the words in the input text. They also propose a simple fuzzy matching method to attempt to improve the recall accuracy of the concept matching scheme. They additionally make use of the Medline word embeddings provided by (Pyysalo et al, 2013) as well as the use of WordNet lexical classes using the SuperSense Tagging utility by (Ciaramita et al, 2006). Their ontology embedding layer focuses on embedding the concatenation of common ancestor concepts between drug candidate pairs along the *is-a* (subsumption) relations define within ontology. They also experiment with including the sequence of concepts from leaf to root of the *is-a* relation hierarchy. Their model demonstrated the second best reported an f1 metric of .751 on the classification task and .6129 on the detection task using the SemEval 2013 benchmark.

This methodology is further explained in Chapter 3 Methodology where it is used as the baseline control for evaluating the impact of this study.

Observations and Gaps

After thoroughly reviewing the Relation Extraction literature with a focus on studies that utilized the DDI benchmarks for evaluation, several noteworthy observations and gaps can be revealed. The evolution of methodology in the field shifted towards Deep Neural Networks around the time of great advancement in the field of Image Processing and Computer Vision which benefited greatly from Deep Learning methods. Although the impact of Deep Neural Networks has had some statistically significant improvement to the accuracy of these models, they have not effectuated the same magnitude of improvements witnessed in other fields of study.

The literature shows a clear convergence of methodology around RNN/LSTM building on the observation that natural language requires understanding of the meaning of a sentence which is largely influenced by the word ordering. The use of Bidirectional LSTM, which provides left-to-right and right-to-left processing of the input sequence has become the popular choice and the primary element responsible for significant improvements to the performance of these models. Other common elements within the recent work involves the use of Pooling layers that follow the LSTM layers. The use of Max Pooling and Attentive Pooling are present in the highest performing models. The pooling layer serves as a dimensionality reduction (down-sampling) by collapsing the time-dimension by selecting the most important features from the sequence. Additionally, the Attention layer helps provide an importance weight to the features that has shown clear improvement in model performance. The combination of using LSTM with Pooling layers yields a summary feature representing the sentence to be classified by the down-stream classifier layer. The hyperparameters tuning across these models vary but share some common elements including the use of a drop out strategy that most commonly is set to .5 (50% drop-out). The number of LSTM units (i.e. dimensionality of LSTM output layer) ranges between 100 and 1024 across the more recent models.

All Neural-based models converged on having a fully connected (dense) layer prior to the classifier output layer. Every neural model leveraged Softmax classifier and used Back-Propagation Through Time (BPTT) for training. All models performed weight initialization using either random or used weight initialization strategies such as Xavier (Glorot & Bengio, 2010).

Other common elements include the use of Embedding layers for feature representation. Most studies found that random weight initialization of the embedding layer underperformed pre-trained embeddings. The use of word2vec (Mikolov et al, 2013) was the most common word embedding algorithm. Numerous studies found that the use of the Medical Word Embedding (Pyysalo et al, 2013) trained on a large medical corpus produced a better overall performance which is attributed to the significantly smaller number of out-of-vocab instances encountered in the DDI dataset. A few studies noted that the dimensionality of the vector within the embedded space could have a slight impact to the performance of the model. They stipulate that too large of a dimension can introduce noise, while too small of a dimension can result in loss of informative features. The most common word vector dimension size used is 200.

A common model architecture pattern observed was the use of multiple input channels. Prior to this approach, most studies relied on a single input channel that simply concatenated together multiple features into a single input vector. This approach demonstrated improvements on model performance. Different input features were used throughout the studies, but the most common included word vectors (as described earlier), positional vectors (measuring the distance of a word to the drug entity candidate), part-of-speech, Shortest Dependency Path (SDP) of words between the first and second drug candidate pair, and WordNet lexical classes (using SST).

Other common model architecture variations were one-stage vs two-stage pipelining of models. The former involves a single end-to-end model for performing relation classification, whereas the latter (two-stage) involves training a model optimized for the detection (binary classification) task and then optimizing another model for the multiclass classification task. Other model architectures involved joint models that use different layers that are then merged before a final hidden layer. These joint models tend

to leverage different NN layers, such as having a word-based and a character-based model side-by-side and then merged. Ensembling multiple models was popular during the early pre-Neural work and typically involved a majority or union voting scheme. For SVM-based models, having more than one model was required since SVM are binary classifiers and therefore require use of a “one-against-all” strategy.

The ordering of word sequences was explored by several studies including using the syntactic tree and performing a recursive depth-first or breadth-first traversal to determine the ordering of the word vectors. Other experiments included splitting the sentence into two or three segments. The former is based on the SDP path between both drug candidate entities, the latter based on the words before, in between, and after the drug candidate entities. None of these elements have been demonstrated to contributing directly to a performance improvement.

Some common pre-processing techniques for preparing the feature input were observed. One notable pre-processing step that consistently improves performance of any model is the known as “negative instance filtering” where certain negative DDI samples are discarded helping to balance the class representation in the training set. All studies noted the class imbalance of the DDI corpus having a negative effect on the model performance. Some classes such as *Int* were consistently the most misclassified across all studies. This is due to its underrepresentation in the training set vs the test set provided by the benchmark. While the negative class label is overrepresented in the training set. This, however, is an intended design of the dataset and used to mitigate overfitted models that do not generalize well in the test dataset. Another consistent finding was the rate of misclassification for the Medline portion of the dataset. The Medline corpus is a large collection of scientific journals that contain very complex terminology and phrasing that have shown to be difficult to accurately classify. The DrugBank training samples feature brief and concise sentences intended for a broader non-expert audience to understand and relatively easier to classify.

Another common pre-processing technique is to mask (or blind) the drug entity candidates from the input sequence to the model. This approach assumes that the actual

drug name is a variance that does not factor into the surrounding phrasing that indicates a DDI relation.

The evaluation analysis of most studies presented their best model performance including recall, precision, and f1 performance. This however does not indicate whether the model is able to consistently reproduce the same performance. Kavuluru et al (2017) encourage future work to provide both the best and average model performance in their evaluation. In their work they noted that any given instance of a model can get lucky and achieve a high metric that cannot be reproduced. In addition to averaging f1 metrics over multiple models runs, another rigorous approach for establishing reproducibility is to use k-fold cross-validation.

In (Segura-Bedmar et al, 2014) and (Segura-Bedmar et al, 2010), that included a review of methodologies for the original SemEval 2011 and 2013 challenge participants, the authors noted a surprising lack of semantic information being incorporated into the proposed models. Much of the focus was placed into syntactically derived features using various NLP tools, but no use of biomedical knowledge sources was incorporated into the models. This idea of incorporating external knowledge is still relatively unexplored. Lamurias et al (2014 & 2019) were the first to explore use of a biomedical ontology (ChEBI) on this DDI relation extraction task. Their model demonstrated the second overall best metrics on the SemEval 2013 DDI benchmark. In their work they discarded much of the ontology and only leveraged the *is-a* hierarchical relations to encode the drug candidate entities within the model. An ontology is a form of explicit knowledge that has been curated by domain experts helping to define the vocabulary and relations between concepts. Intuitively, the inclusion of ontologically driven knowledge together with the training dataset should yield additional information for the model to learn from. This dissertation will build upon the work from Lamurias et al (2019) and Zhang Y et al (2018) and continue to explore additional use of ontologies as an embedding layer within the relation classification model.

Summary

Since the formalization of NLP tasks by the MUC conferences there has been steady advancement in methodologies for performing NLP tasks such as entity and relation extraction. The creation of standard benchmarks has served as a key foundation driving research into complex problems such as relation extraction over biomedical benchmarks. Since the creation of the SemEval 2011 and 2013 DDI benchmark, over two dozen studies have been conducted evaluating various AI-based methodologies on the problem of Relation Detection and Classification within a 9-year span. The steady improvement of performance demonstrates the advancement of methodology on the task, raising the initial f1 metric from .6001 (Segura-Bedmar et al, 2011) to .773 (Zheng et al, 2017). These studies have reflected the latest advancement of AI methodology on NLP tasks. Starting with SVM-based methods in the early 2010's and quickly shifting towards Deep Learning methods in the mid and late 2010's. Although these Deep Learning methods have not had the same kind of impact in the NLP field as they have in image processing and computer vision field, they have demonstrated a significant improvement with an increase of f1 metric of .103 over the SVM-based methods and an increase of .173 since the inception of the benchmark. The literature provided an excellent trajectory to guide future work in this field and this review has identified gaps and unexplored areas that motivated the methods detailed in Chapter 3 Methodology.

Chapter 3

Methodology

For this dissertation, several extensions to the work of Lamurias & Couto (2019) and Zhang Y. et al (2018) were implemented with the goal of achieving a higher overall F_1 accuracy score on the SemEval 2013 Drug-Drug Interaction (DDI) benchmark (Segura-Bedmar et al, 2014; Herrero-Zazo et al, 2013). The enhancements include modifications to the model architecture that features a new ontology neural embedding layer that supports multiple ontology relations. Additionally, an alternate method for matching words and phrases to the most specific and relevant concepts within an ontology was implemented. The longest-span match method helps with the selection of candidate concepts that are the most taxonomically specific within an ontology.

As explained in Chapter 1 and 2, the DDI classification task involves the relation classification of a DDI type given a sentence text and a pair of candidate drug entities within the sentence. There are 5 possible classification labels: mechanism, effect, Int (unspecified DDI), Advice (advice about a DDI), and negative (no DDI present). The models process the input including a pair of drug entities and output one of the five class labels as a prediction. The following sections describe the two baseline models used throughout this study to compare and evaluate the methodology presented in this study.

Baseline: Hierarchical RNN

Zhang Y. et al (2018) proposed a novel DDI classification model that uses five different input channels. The primary contribution over previous approaches is the use of a split input sequence with hierarchical RNN layers. This approach partitions the input sequence into left, middle, and right. Each of these subsequences undergoes a feature extraction to produce word vectors, part-of-speech tags, and positional distance from each word token to the two drug entities mentioned in the text. The partitioning of the subsequences is determined by the position of the drug entities. The words starting from the beginning of the sentence to the first entity (exclusive) represent the *left* sequence. The words between the first drug entity and the second drug entity (exclusive) represent the *middle* sequence. The words starting at the second drug entity to the end of the sentence

represents the *right* sequence. Figure 13 illustrates the partitioning of a sample sentence into left, middle, and right subsequences excluding the pair of drug entities.

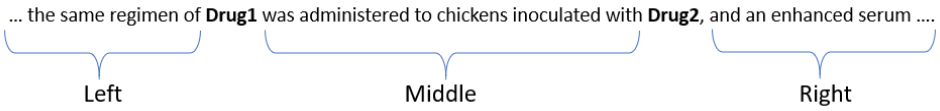

Figure 13 - subsequence example (using sample text from DDI corpus**)**

The words in each subsequence are embedded using a pre-trained medical word embedding matrix using the Medline corpus of abstracts and the word2vec utility by Mikolov et al (2013) to produce embedding weights with 200-dimensional vector representation. Each word in the sequence is first encoded using a one-hot vector representation that is projected into the low-dimensional word vector representation using the pre-trained embedding weights within the embedding layer of the model. The use of word embedding as a method for text representation in natural language tasks is pervasive and its contribution to accuracy performance is significant and well documented in literature (Bengio et al, 2003; Mikolov et al, 2013; Pennington et al, 2014).

The part-of-speech for each word in the sequence is extracted using spaCy POS tagger. The part-of-speech is represented as a one-hot vector presented to an embedding layer that learns the embedding weights together with the rest of the model training. The dimensionality of this embedding layer is 10. Figure 14 shows an example of the part-of-speech tags extracted for each word in a sample text. The use of part-of-speech tags for natural language tasks including relation extraction is pervasive and its positive contribution to accuracy is recognized in literature.

The two positional distance features per word are extracted by measuring the word distance from the first and second drug entity in the sentence. The distance features are represented as one-hot vectors and presented to an embedding layer that learns the embedding weights together with the rest of the model. The dimensionality of this embedding layer is 10 for each of the two distance features represented. Figure 14 shows an example of distance 1 and distance 2 extracted from a sample text. The motivation for using positional distance is based on the positive impact to accuracy reported in literature

by numerous studies (Liu et al, 2016; Zheng W. et al 2017; Zhang Y. et al 2018) (see Chapter 2 Literature Review).

Input Text:	"Cimetidine can inhibit the metabolism of chloroquine ..."
Part-of-Speech:	noun > verb > verb > determiner > noun > preposition > noun
Distance1:	0 > 1 > 2 > 3 > 4 > 5 > 6
Distance2:	6 > 5 > 4 > 3 > 2 > 1 > 0

Figure 14 - Example of part-of-speech and distance features

Figure 15 shows the model architecture for Zhang et al's Hierarchical RNN DDI classifier. Each subsequence is represented as an input channel with a feature group of words, part-of-speech, and two distance features. Each channel merges the embedded features and applies an importance weight learned using an Attention layer. This helps to differentiate the significance of each word in terms of the relation classification task. The use of Attention to weigh importance of features in DDI relation classifier was originally proposed by Wang L. et al (2016) and discussed in *Chapter 2 Literature Review*. Each channel is connected to a Bidirectional RNN layer with 100 units that outputs a 200-dimensional vector representation. The max input sequence of each subsequence is determined by finding the longest subsequence length of the entire training dataset.

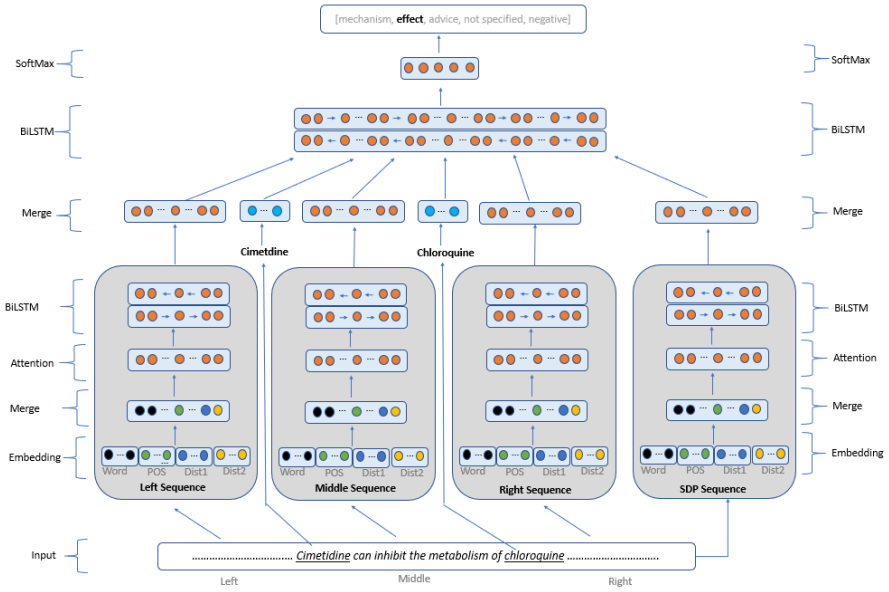

Figure 15 - Zhang et al (2018) Model Architecture

In addition to the *left*, *middle*, and *right* sequences, Zhang et al's model includes a fourth input channel based on the Shortest-dependency path (SDP) between the pair of drug entities in the input sentence. Figure 16 shows an example of the SDP parse of the sentence *"Cimetidine can inhibit the metabolism of chloroquine…"* where cimetidine and chloroquine are the pair of drug entities and "cimetidine > inhibit > metabolism > of > chlorquine" is the SDP between the pair of drugs.

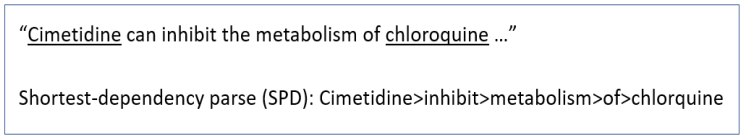

Figure 16 - SDP Example

The features of the SDP channel mirror the previous sequence channels including with words, part-of-speech, and positional distances extracted and embedded using the same methods described earlier. Like the other channels, the SDP channel features are merged and weighed using an Attention layer. The resulting feature representation is connected to a Bidirectional LSTM layer with 100 units that produces a 200-dimensional vector. Table 1 shows the vocabulary size and output dimension for the embedding layers and Table 2 shows the max sequence lengths used in this model.

Embedding Features	**Input Dim (Vocab Size)**	**Output Dim**
Words	685,000	200
Part-of-Speech	32	10
Distance 1	601	10
Distance 2	601	10

Table 1 - Embedding Dimensions for Zhang et al (2018)

Channel	**Input Length (Max subsequence Length)**
Left	98
Middle	107
Right	144
SDP	12

Table 2 - Max subsequence lengths in Zhang et al (2018)

The final hidden layers merge the output from the 4 RNN layers together with the word vector representation of the 2 drug entities in the input text. The merged vectors are presented as a sequence to a final Bidirectional LSTM layer and then to a Softmax classifier as the output layer. The LSTM is configured to 100 units producing a 200-dimensional vector that feeds the Softmax classifier which in turn outputs the probability distribution over a 5-dimensional vector (one for each class label). The model uses the Root Mean Squared (RMS) optimization algorithm with a learning rate configured to 0.001 and SGD with minibatch size of 64. Using this model architecture and hyperparameters, this study evaluated the performance of this model using the original source code and was able to reproduce the .729 F_1 accuracy reported by Zhang Y. et al (2018). The next section discusses the enhancements made to this model to demonstrate the accuracy improvements of incorporating the ontology neural embedding layer.

Enhancing Hierarchical RNN Model

The Hierarchical RNN model architecture by Zhang Y. et al (2018) is used in this study as one of two baseline models. The baseline models are used in a series of controlled experiments to evaluate the accuracy performance impact of a new neural ontology embedding layer proposed and presented in this study. This model is extended to include an embedding layer whose weights are initialized using a pre-trained embedding method discussed later in *Concept Embedding Methods* section of this chapter. Figure 17 shows the enhanced model architecture with the additional embedding layer trained using the ChEBI ontology and applied to the two drug entities of the input. The two drugs are mapped to a drug concept within the ChEBI ontology using the concept matching method discussed in *Concept Matching Methods* section of this chapter. The drug concept is embedded and presented to the final Bidirectional LSTM layer of the model.

Figure 17 - Hierarchical RNN with Neural Ontology Embedding layer

Baseline: Hierarchical RNN with BO-LSTM

Lamurias & Couto proposed two variations of the BO-LSTM model to detect and classify the relationship between pairs of drug entities within medical literature. The first variation extended the model architecture proposed by Xu Y. (2015) and the second extended Zhang Y. et al's (2018) Hierarchical RNN architecture presented in the previous section. The primary contribution over the previous work involves the use of an ontology embedding layer that uses the ChEBI ontology as a method to capture common ancestor concepts between drug entities that can reveal additional semantic information not otherwise available in the training data. The BO-LSTM model extending Zhang et al's Hierarchical RNN demonstrated a significant improvement to accuracy by increasing the F_1 from .729 to .751. As reviewed in *Chapter 2 Literature Review*, this is the second highest reported F_1 accuracy and the highest independently reproducible F_1 accuracy for the SemEval 2013 DDI benchmark.

This variation of the BO-LSTM model architecture, shown in Figure 18, includes the original 4 input channels presented by Zhang et al along with an additional channel labeled *Common Ancestors* proposed by Lamurias & Couto (2019). This channel embeds the common ancestors for both drug entities in the input sentence along with the part-of-speech and positional distance. Lamurias & Couto place a restriction on the ChEBI ontology by discarding all semantic relations except for the *is-a* relation (known as subsumption). The *is-a* relation is a directed and transitive relation that is considered a fundamental aspect of an ontology, providing the taxonomy that defines the classes of concepts within a domain.

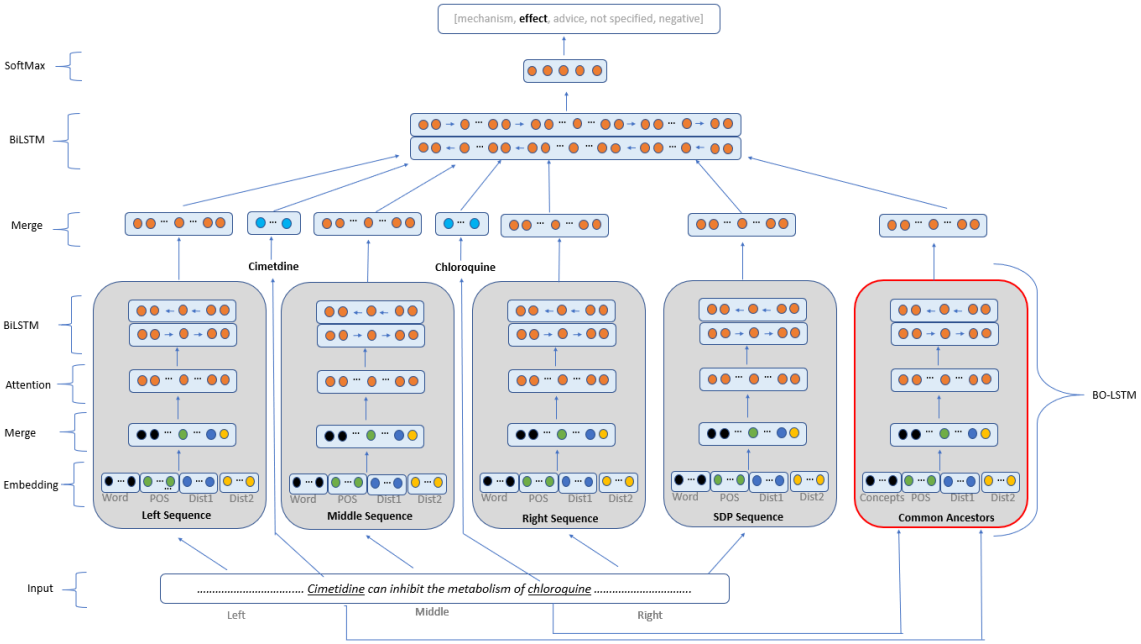

Figure 18 - BO-LSTM extension of (Zhang et al, 2018) Architecture

The input processing for the concept embedding channel begins by performing a concept look-up of each drug entity in the input sequence within the ChEBI ontology. This step attempts to first find a matching concept within the ontology using the exact drug name. This may fail to produce a match and therefore a fuzzy match using Levenshtein distance is used to find the best matching concept within ChEBI. These matching concepts may or may not represent the correct drug concept. Failing to identify the correct and relevant concept has a negative effect on the accuracy of the model.

Lamurias & Couto experimented with several variations of the model but ultimately found the best performance is achieved when including information about a concept's hierarchy. Lamurias & Couto (2019) define a restricted ontology as a directed graph O = (C, R) where C is the set of all concepts (e.g. all the drug concepts within ChEBI) and R is the set of all *is-a* relations (e.g. the concept hierarchy for ChEBI). The concept embedding approach used relies on obtaining the ancestors and then identifying the common ancestors between pairs of drug concepts found within the input sequence.

The concepts are encoded into a sequence of one-hot vectors in order from generic to specific along the *is-a* hierarchy. These one-hot concept vectors have a dimensionality of 2,170 each dimension representing a unique concept. This is far fewer than the 114,000 vocabulary of concepts within ChEBI. Lamurias & Couto restricted the working vocabulary size of concepts to those found within their training and test sets. This limiting assumption is a tactical decision to overcome a series of challenges including computational issues, large memory footprint, and very long training times. Unfortunately, this limitation results in lower accuracy when running in a real-world scenario where the documents and drugs mentions are not known in advanced.

Lamurias & Couto evaluated two layers for concept feature representation: *Common Ancestors* and *Concatenated Ancestor Concepts*. The *Common Ancestor Concepts* layer is presented a sequence of the common ancestor concept vectors. The *Concatenated Ancestor Concepts* layer is presented the concatenation of the ancestors for each respective drug concept vector. This concatenation produces a sequence of vectors that list the ancestors of c_1 followed by the ancestors for c_2. The common ancestor's method may not produce feature vectors due to instances where the pair of drug concepts do not share any common ancestors. In those instances, the channel fails to present information to the model. The *Concatenated Ancestor Concepts* approach, however, always produces information about each respective concept ancestor hierarchy.

The vectors for the drug concepts are processed through an embedding layer that is initialized using random values and trained using back-propagation with gradient decent (SGD) together with the rest of the model. This embedding layer produces dense vectors with a fixed dimensionality of 300. This ontology-concept embedding is given by: $\mathbf{E} = \mathbb{R}^{|C| \times D}$ where **E** is the embedding matrix, D is the dimensionality of the embedding space

(50 in this case), and |C| is the number of concepts defined in the ontology (a vocabulary size of 2,170 concepts). The embedding of a concept vector into the ontology-embedding space can be defined by the function f: $v_c \rightarrow v_e \; in \; \boldsymbol{E}$ and v_e is the embedded vector produced by projecting (i.e. mapping) the concept vector v_c into an embedding space **E**. Table 3 summarizes the hyperparameters used to train the embedding layers for both of these ChEBI concept embedding channels.

Embedding Features	**Input Dim (Vocab Size)**	**Output Dim**
Pre-trained Words	685,000	200
Part-of-Speech	32	10
Distance 1	601	10
Distance 2	601	10
ChEBI Common Ancestors	2,170	300

Table 3 - Table showing the BO-LSTM embedding layers along with their input dimension, output dimension, and input length

All the channels are respectively connected to an Attention and Bidirectional LSTM layer where the channel output is merged using vector concatenation before proceeding to the final Bidirectional LSTM layer. The output layer is a Softmax classifier producing a probability distribution over a 5-dimensional vector using RMS optimizer with a learning rate of .001 and SGD using minibatch size of 64.

Enhancing BO-LSTM Model

This BO-LSTM model that extends Hierarchical RNN is used as the second baseline model in a series of controlled experiments to evaluate the accuracy performance impact of the new neural ontology embedding layer proposed and presented next in this chapter.

The primary enhancement to the *BO-LSTM* model involves changes to the ChEBI embedding layer labeled *Common Ancestors* in Figure 18 above and corresponding input pre-processing. The new embedding layer addresses the limitation where only *is-a* subsumption relations are supported by *BO-LSTM* model and all other semantic relations are discarded resulting in a loss of semantic information. Additionally, the new model changes allow for a larger vocabulary size in order to maximize the benefit from the

ontology embedding and therefore increases the generalization of the model by supporting the entire set of ChEBI concepts, reducing the occurrence of out-of-vocab when applying the model to unseen datasets. Figure 19 shows the enhanced model architecture with the improved layers outlined in red.

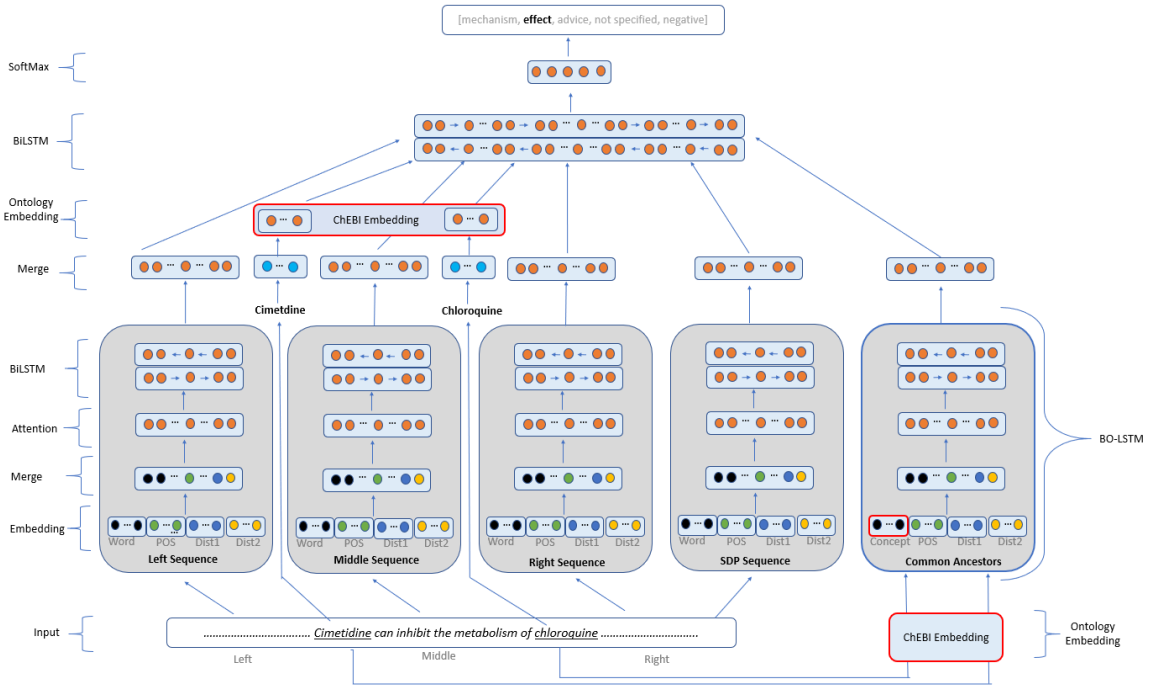

Figure 19 - Enhanced *BO-LSTM* architecture with new Neural Ontology Embedding layer**s**

In addition to the improvements to the *BO-LSTM Common Ancestors* embedding layer, the improvements made to the Hierarchical RNN model are repeated here by including a ChEBI embedding layer for the two drug entities and connected to the final Bidirectional LSTM layer of the model. This is shown in a red outline on the top-left of Figure 19. Furthermore, an improved concept matching method to accompany the new embedding layers is presented later in this chapter. The following sections define and formalize these enhancements and explains the reasons why they improve the performance of the model.

Concept Embedding Method

The concept embedding method presented and evaluated in this study improved upon Lamurias & Couto's embedding approach by introducing a neural ontology embedding method that includes all relevant semantic relations from the ontology in addition to the taxonomic structure (i.e. *is-a* relations). These improvements demonstrated an increase to the model accuracy by including additional semantic information captured from the expanded support of semantic relations defined within the ontology. During this study, several variations of this method were evaluated before identifying the best performing method called: *Ontology-learned Identity Vectors* (OLIV) presented in this section. The description and evaluation for the alternative methods explored are presented in *Appendix B.*

The OLIV embedding method defines an ontology as a directed graph O = (C, R) where C is the set of all concepts and R is a set of heterogenous relations such that $R = R_1 \cup R_2 \ldots \cup R_n$. Minimally an ontology is expected to provide the *is-a* relation that defines the taxonomy of concepts. Additionally, an ontology may define other meaningful relations within the domain called semantic relations. The set R_1 is reserved for the taxonomic (is-a) relations and the sets R_2 through R_n for the remaining relations. Examples of such relations might include *has_role*, *synonym_of*, or *may_treat* to represent the semantics of the specific domain. For the ChEBI ontology there are a total of 9 semantic relations defined.

To leverage these additional relations defined within the ontology, the ontology graph is first partitioned into multiple subgraphs each comprised of a set of homogenous relations as follows:

O = (C, $R_1 \cup R_2 \ldots \cup R_n$) →

{ $O_1 = (C_1, R_1)$, $O_2 = (C_2, R_2)$, …, and $O_n = (C_n, R_n)$ }

where { C_1, C_2, …, C_n} ⊂ C and each R_i is comprised of homogenous relations.

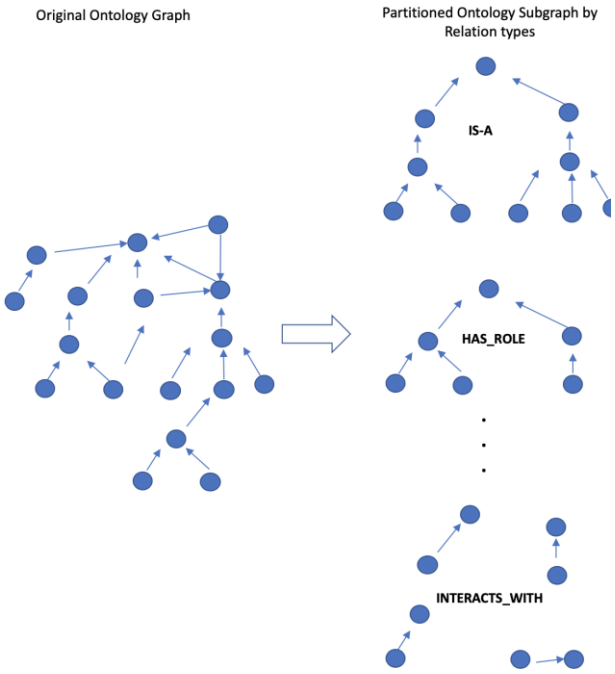

Figure 20 – Decomposition of Ontology into Subgraphs

For each $c_i \in C$, a unique vector is assigned using a one-hot vector encoding scheme. Once all concepts have been assigned a vector, a new embedding matrix **E** is trained such that **E** $= \mathbb{R}^{|C| \times D}$ where $|C|$ represents the number of concepts defined within the ontology graph O and D the dimensionality of the embedding space. The dimensionality D of the target embedding space was refined through experimentation and set to 300 and $|C|$ represents the vocabulary size of 114,000, the number of concepts defined in ChEBI.

The OLIV embedding method is used to learn the weights of the embedding matrix **E** using an unsupervised ontology-guided pre-training approach. The approach starts by constructing an identity set V_i for each concept $c_i \in C$. V_i represents a concept's identity in terms of its inherited and related concepts across the ontology O. The corresponding identity vector v_i for a concept c_i can be expressed as a k-hot vector encoding f: $V_i \rightarrow v_i$ where k is the dimensionality of v_i and equal to the vocabulary size of the embedding. The identity vector, v_i, is used to fit a neural network trained to predict f: $c_i \rightarrow v_i$.

The identity V_i for a concept c_i is defined as follows:

(1) $V_i \leftarrow$ Gather the ancestors of c_i within the taxonomy (is-a) relations in R_1

(2) $V_i \leftarrow$ For c_i **and** each ancestor gathered in (1), gather the related (i.e. connected)

concepts in R_2 through R_n

For this study, the number of ancestors gathered for c_i in (1) was limited to a maximum of 6 traversals (i.e. distance) from c_i. A limit was initially chosen based on intuition and refined through experimentation to balance accuracy with computational efficiency. Because all concepts share a common ancestry within the ontology, these very common ancestors are not informative in differentiating one concept from another. As such, a limit was used to reduce the number of ancestors considered in the identity resolution of a concept. Also note that the identity set definition for a concept does not include the concept itself. This reduces the number of unique concepts represented across all identity sets and therefore reduces the dimensionality required to encode the identity vector v_i. This reduction of ancestors and concepts helps reduce the computational cost to compute the set identity set V.

Once the identity vector v_i for every c_i in the ontology has been resolved, the OLIV model can be trained. Figure 21 shows the OLIV embedding model along with an example depicting the *Amoxicillin* concept as input to the model and the corresponding identity vector as the expected output. Figure 21 also shows the correlation of the identity vector for *Amoxicillin* using two ChEBI ontology subgraphs of *is-a* and *has_role* relations. This model is trained using a hidden dense layer with the Identity activation function and a Softmax output layer. For this study, the model was trained using 100 epochs over the entire set of 114,000 concepts using SGD and minibatch size of 16.

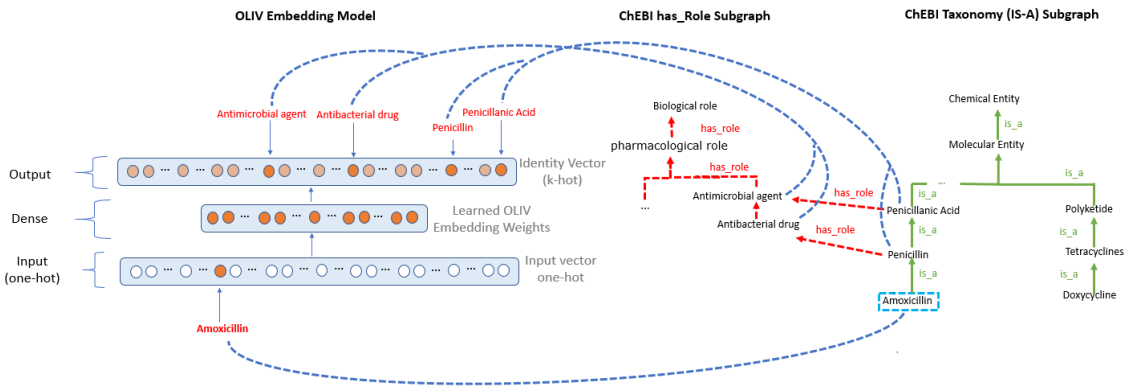

Figure 21 - OLIV Neural Ontology Embedding Model

The OLIV method differs from other approaches because it uses the ontology to guide the training of the embedding. Prior work by (Bengio et al, 2003, Mikolov et al, 2013, Pennington et al, 2014, Hill et al, 2016) among others use an unsupervised corpus-guided approach to train the embedding using word context-prediction tasks. Other embedding techniques, such as Autoencoders (Hinton & Zemel, 1994), are trained to reconstruct the input vector using several hidden encoder layers that perform dimensionality reduction of the input vector. Other common approaches learn the embedding together with the classification model. The embedding weights are randomly initialized and then learned using a task-based supervised training approach. These different approaches were evaluated during this study and their results are presented in *Chapter 4 Results* and their descriptions in *Appendix B.*

Summary of Model Architecture Changes

The primary changes to the Hierarchical RNN architecture occurs to the two drug entity channels while the changes to the *BO-LSTM* model occurs within the Common Ancestors embedding channel where the embedding layer used is replaced with the trained OLIV embeddings. The other channels remain unchanged and follow the same methods used by Zhang et al and Lamurias & Couto as described in the above sections. The ontology embedding channels use a pre-trained concept vector embedding using OLIV (discussed in the previous section).

Concept Matching using Longest-Span Match

Like a thesaurus, ontologies provide synonyms for each concept that may include strings that serve as alternate forms of describing a concept. These synonym strings are not guaranteed to be unique to a concept and therefore may occur among many concepts. Additionally, these synonym strings may occur as a substring of another longer synonym for a different concept. This means that a substring of a given input text may produce one or more overlapping concepts after performing concept matching. The presence of overlapping concepts presents an ambiguity since one of these concepts must be chosen as the best matching concept.

To address this ambiguity, the following resolution is applied: Select the concept

where the matching synonym string S has a greater length than all other substrings S' that match a concept. This matching concept is considered the longest spanning concept. For example, the phrase *"... vitamin A and medications with retinoic acid receptor alpha antagonists must be avoided ..."* has many words that can represent concepts such as '*Acid'* or '*Receptor'* or '*Retinoic Acid'*, but the most specific concept match would be '*Retinoic Acid Receptor Alpha Antagonists'*. This longer spanning match therefore covers the other sub-sequence matches. The longest spanning match is considered a well-known optimization for dictionary matching algorithms and should be leveraged as part of this model to improve the underlying accuracy of identifying the correct concept prior to encoding. Figure 22 shows an example where multiple concepts share the same words (top) and an example sentence with multiple matching concept spans including the longest-span.

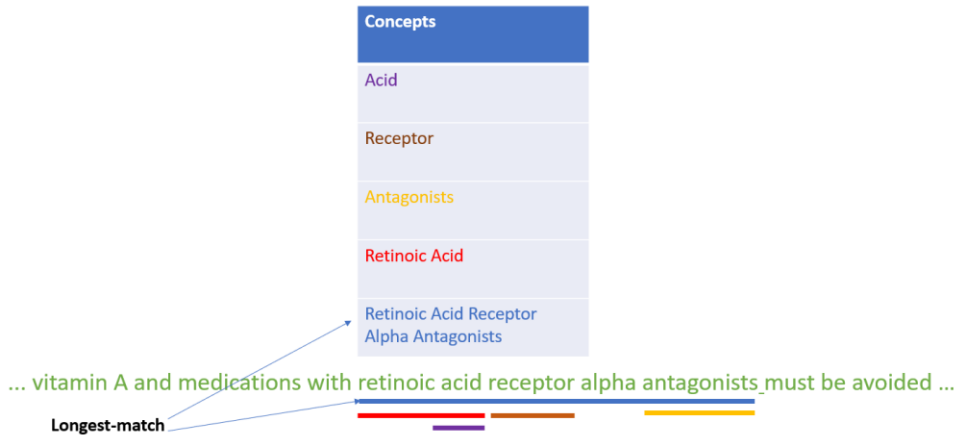

Figure 22 - (top) Example of concepts with shared words (bottom) overlapping concept spans

This study implemented and evaluated several different concept matching methods. The best performing method was based on Concept Mapper (Tanenblatt, Coden, & Sominksy, 2010) which was reimplemented to decouple it from the UIMA framework to evaluate the method in isolation. The other two methods evaluated were a basic string-matching that served as a baseline comparison and a finite state transducer implementation that constructed a finite state machine from the ChEBI ontology and performs a character-level state transition as it scans the input text where each state can hold zero or more concepts.

The Concept Mapper method utilizes an efficient data structure that can represent

a large dictionary of concepts and supports an efficient matching algorithm. The dictionary representation uses a nested Tree Map data structure that is keyed based on the first word of a concept's lemma or synonym entry and then keyed by the word length of the concept entry. An important goal of concept matching is to represent the dictionary entries in a memory efficient way so that it can scale to millions of entries. Additionally, the dictionary representation supports an efficient matching algorithm. Figure 23 shows an example of a dictionary data structure representation (right) and a small example of ChEBI concept entries used to generate the dictionary (left).

ChEBI Example Concept Dictionary
Acid
Antagonists
Receptor
Retinoic Acid
Retinoic Acid Receptor Modulator
Retinoic Acid Receptor Alpha Antagonists
Retinoic Acid Receptor Gamma Antagonists
Vitamin A
Vitamin B
Vitamin D

ChEBI Concept Mapper Dictionary Representation		
Key	Entry Length	Entries
Acid	1	Acid
Antagonists	1	Antagonists
Receptor	1	Receptor
Retinoic	2	Retinoic Acid
	4	Retinoic Acid Receptor Modulator
	5	Retinoic Acid Receptor Alpha Antagonists
		Retinoic Acid Receptor Gamma Antagonists
Vitamin	2	Vitamin A
		Vitamin B
		Vitamin D

Figure 23 - Example showing the dictionary data structure representation (right) given a set of ChEBI concepts (left)

Figure 24 shows a sample text with multiple overlapping concepts denoted as underlines. The Concept Mapper matching algorithm uses a greedy approach to identity only the longest spanning concepts in the text. For this sample, *"Vitamin A"* and *"retinoic acid receptor alpha antagonists"* represent the two longest spanning concepts.

Figure 24 - Sample text showing several overlapping concepts

Figure 25 shows the concept matching algorithm steps as it processes the text using the example dictionary representation shown in Figure 23. The algorithm attempts to match the entire text span and incrementally reduces the text span length until a match is found and the starting position of the search span is advanced forward.

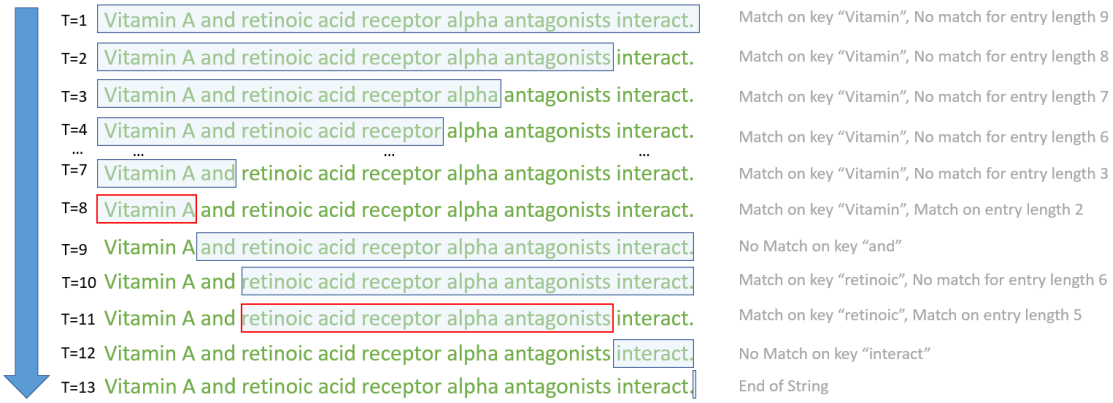

Figure 25 - Example execution of Concept Mapper algorithm over sample text

The results of the concept matching experiments are presented in *Chapter 4 Results – Concept Matching Methods.*

Experimental Design

This study performed several experiments to evaluate the optimal ontology embedding model design, parameters, and experiment settings. The results from these experiments helped answer research questions regarding the barriers and issues previously mentioned in this document. Furthermore, answers to these research questions helped to guide overall methodology presented in this dissertation and advance the state-of-the-art in the study of Relation Classification and establish a new area of exploration in Neural Ontology Embedding.

Ontology Embedding Method Experiments:

This study evaluated 4 different proposed neural ontology embedding methods. The purpose of these experiments was to identify the method that demonstrated most promising results using two comparative test tasks designed to assess the quality of the concept embedding. The first test is labeled the *DDI Detection Test* and uses a subset of the DDI benchmark dataset to measure the binary classification accuracy using a baseline model where each embedding method is measured. The baseline model architecture included the pre-trained ontology embedding as the input layer and a SoftMax with Cross-Entropy loss function as the output layer. The max F_1 metric for each embedding method was collected and compared. The OLIV embedding method demonstrated the best performance for this

test.

The second test, labeled *Ontology distance Test*, measured and compared the relative distance between concepts within the ontology space and the embedding space. The purpose of the test was to identify which embedding method preserved the relative distances between concepts within the embedding space. The intuition underlying this test is based on the idea that the ontology serves as a source of truth for each concept and can be used as a ground truth to measure the mean error between the embedded distances and the ontology distance between a pair of concepts. *Appendix B* defines the evaluation task in detail and *Chapter 4 - Results* presents the results for both tests.

Additional tests were performed to measure the optimal OLIV embedding dimensionality by training the embedding layer using 50, 100, 200, and 300 dimensions and evaluating the relative F_1 accuracy performance using the Zhang et al (2018) model as a baseline.

Concept Matching Method Experiments:

These experiments will evaluate an optimal concept matching method. The proposed methodology relies on the ability to accurately match concepts to represent their ontological relations as features within the model. Preliminary experimentation found that a significant percentage of concepts are missed (~30%) using basic string matching. These experiments will evaluate methods for improving the concept matching using alternative methods that will be compared. Another finding during preliminary experimentation found that a significant number of drug names matched more than one concept using basic string matching over canonical and synonym forms within the ChEBI ontology. This suggests the need for an algorithm to select the best match.

1. Does Concept Mapper provide better recall accuracy when compared to basic string-matching algorithm used during preliminary experimentation?
2. Does a Finite-State Transducer representation of the dictionary and concept matching algorithm improve the runtime performance of the system?
3. How does a Finite-State Transducer recall accuracy compare to Concept Mapper and Basic String-matching?

Ontology Enhanced Models vs Baseline Experiments:

To evaluate the overall accuracy performance impact of the Ontology embedding layer within a DDI Relation Classification model, a series of controlled experiments were conducted using the Zhang (Zhang et al, 2018) and Lamurias & Couto (2019) *BO-LSTM* as baseline control models. This experiment first established the baseline accuracy for each of the two control models using the original source code for both models. The experiment ran 10 trial runs of each model using the best performing hyperparameter configuration reported in their respective studies. The baseline accuracy for each trial run was established by running each model to identify the best F_1 score using a validation test set. The reported F_1 score for each baseline model was reproduced or marginally exceeded in this study. *Chapter 4 Results* presents a summary of these findings.

Once the baseline accuracy was established, each baseline was extended using the OLIV embedding layer. OLIV was identified as the best performing ontology embedding method using the previously described *Ontology Embedding Method Experiments* and the results are presented in *Chapter 4 Results – Ontology Embedding Method Results.* Each baseline model was treated with the OLIV embedding layer and 10 trial runs were conducted each trained for 100 epochs using an 80/20 training and validation split. Using the DDI Test set, the F_1 score was measured using the best performing model on the validation set for each trial run.

The controlled baseline trials and treated baseline trials are evaluated using the min, max, and mean F_1 metrics collected from the 10 trial runs to determine whether the treated baseline trials exhibited an improved F_1 accuracy. This experiment was used to support the primary hypothesis of this study that the inclusion of an ontological-concept embedding layer contributes a significant improvement to the F_1 accuracy.

Additional Experiments:

Additional experiments were conducted to help answer additional research questions raised during this study. During this study, a Java implementation of several different DDI Classifier models were implemented along with supporting function. The experiments in this section were conducted using this Java framework labeled: *DDI workbench* (see *Appendix D* for source availability). The following experiments were

conducted:

- An implementation of the Lamurias & Couto (2018) *BO-LSTM* standalone model based on (Xu Y. et al, 2017)'s SDP features and ChEBI concatenated ancestors
- Applying the intuition that the structure of the DDI sentence text contains informative clues towards the classification task, a feature labeled as *Structure* was evaluated. This feature performs a symbol masking for all words, numbers, and drug entities. The feature significance was evaluated using the DDI benchmark
- Evaluate the impact of performing full-masking vs partial-masking of drug entities within the DDI sentence text. The intuition is that some feature groups that are primarily focused on the contextual clues, such as word phrasing and sentence structure, benefit from masking the drug words and instead using a symbol masking to hide the words from the learning algorithm.
- Evaluate a multi-stage model architecture were each stage is optimized for classifying a subset of the DDI classes. This approach is based on the insight gained from studying the interrelationships between the DDI class types and elaborated further in Chapter 4 Results.
- Evaluate other model architecture strategies such as pre-merging vs post-merging of feature groups, where different input feature channels are concatenated before or after the RNN layer. Measure the impact of different LSTM units used within the RNN layer of the model.
- Evaluate the feature significance of the SDP feature channel. The use of SDP has become a common trend in recent literature based on the intuition that it reduces the complexity of the sentence by focusing on the dependency path between the pair of drug entities and discarding the remaining words in the sentence.

The above questions were evaluated, and their results are presented in *Chapter 4 Results* along with additional insights gained from the experiments.

Evaluation

This proposal seeks to demonstrate an overall accuracy improvement over the baseline *BO-LSTM* model using the SemEval 2013 Drug-Drug Interaction (DDI) benchmark (Segura-Bedmar et al, 2014, Herrero-Zazo, 2013). This is a well-known benchmark used by the NLP community to assess the performance of a model's ability to detect and classify relations from biomedical text and is the benchmark used the baseline models used in this study (Lamurias & Couto, 2019; Zhang Y. et al, 2018).

The focus of this evaluation will be on the F_1 score achieved when evaluating the model on the DDI benchmark. Specifically, the DDI relation classification and detection tasks. The F_1 score is defined as the harmonic mean of recall and precision for classifying the DDI relations in the validation test set of the challenge. The recall is defined as the number of True-Positive observations over the sum of True-Positive and False-Negative observations. Precision is defined as the number of True-Positive observations over the sum of True-Positive and False-Positive observations. The following formulation for F_1, Recall, and Precision will be used:

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

$$\text{Recall} = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$\text{Precision} = \frac{TruePositive}{TruePositive + FalseNegative}$$

The methods presented in this study improved upon the F_1 accuracy documented by Lamurias & Couto (2019) of .751 for DDI classification and Zhang et al (2018) of .729 resulting in a new state-of-the-art performance for this benchmark. Throughout the literature review, virtually all studies (except one) reported their accuracy in terms of the best F_1 score achieved. This study will report the best F_1 score and additionally conduct 10 trial runs for each model evaluated to demonstrate reproducibility of the results.

This study reproduced the documented results from Lamurias & Couto (2019) and Zhang Y. et al (2018) using their original source code including their evaluation function used to compute accuracy metrics. The baseline models were extended with the methods presented in this chapter with minimal changes to the model architecture and hyperparameters. A Java implementation of various DDI models were implemented along with supporting function to evaluate model performance using the DDI benchmark.

Resources

The following section describes the required software packages and computing hardware requirements for supporting the experiments in this study. All the software packages listed in this section are available as open-source and can be used at no cost for research.

Software

During this study, two different Neural Network environments were utilized. The first environment is based on python using the same learning framework and computational back-end used by Zhang Y. et al (2018) and Lamurias & Couto (2019) model source code. This environment was used to perform a controlled comparison to each model using the original environment. The following packages were used in this environment:

- Python 2.7 and Python 3.7
- Keras 2.3.1 and Keras 2.4.1
- Theano 1.0.5 and TensorFlow 2.x
- DiShIn – semantic database
- Obonet – Obo ontology parser
- Networkx – RDF graph parser
- GenSim – word vector utility

The second environment uses the Java programming language and DeepLearning4J framework with ND4J computational backend. The motivation for porting this work to Java is two-fold. First, the availability and maturity of NLP tools and Ontology parsers are well established and easily available as Java packages. This includes packages such as Apache UIMA with Concept Mapper for performing concept matching and Apache Jena and Riot for Ontology OWL and OBO format parsers, and the Stanford Parser referenced ubiquitously throughout the literature of prior work. Secondly, this implementation will be developed to serve as an experimental workbench and made available to facilitate future work by the community of Java developers and researchers (see *Appendix D*). The following is the list of software packages were used to develop the workbench:

- Java OpenJDK
- Apache Jena and Apache Riot – Ontology parsing
- SST Light Utility for Princeton's WordNet Lexical class tagging
- Stanford Parser - Dependency Graph Parser
- Medline Pre-trained Word Vectors (Pyysalo et al, 2013)
- GenSim vector deserializer – used to convert Medline pre-trained word vector format

- Apache Concept Mapper and Apache UIMA framework – to perform concept matching
- DL4J for Deep Learning Framework
- ND4J Tensor Computation Back-end
- Microsoft Code for IDE with RedHat Java extensions
- Apache Maven dependency Management to resolve the above mentioned open-source packages
- Apache Spark for Distributed Computing – for distributed training

Hardware

To develop, train, and evaluate the methods in this study, an adequate computing environment was needed. The use of 3 different systems were used to train and run trial experiments. These systems included Apple Book Pro with 16GB RAM and 8-core CPU; Lenovo W500 with 16GB and 8-core CPU; and Acer Predator with 16GB RAM, 6-core CPU, and Nvidia GeForce RTX 2060 GPU. For GPU-based training, the Nvidia RTX 2060 GPU was used along with the Nvidia CUDA Toolkit version 11.2, Nvidia Driver version 460.20, and Nvidia Docker Container toolkit. See *Appendix D* for additional information on environment and project configuration.

Chapter 4

Results

During this study, a series of controlled experiments were conducted to empirically identify the optimal methods and associated hyperparameters for several key aspects of the model. These experiments, defined in *Chapter 3 Methodology*, were used to comparatively evaluate the performance impact of different ontology-concept embedding approaches and supporting methods, such as concept matching. The results and intuition gained from these experiments guided the final design for the concept-ontology embedding layer and supporting methods.

A final set of controlled experiments were performed to comparatively evaluate the overall effectiveness of the final DDI classification model against two state-of-the-art baseline models. The SemEval 2013 DDI (Segura-Bedmar et al, 2014) classification benchmark was used to evaluate the relative performance of each model. This chapter will summarize the experiments, results, and document the key findings and insights applied towards the final implementation of the model. This chapter is organized into 3 sections:

- Ontology Embedding Layer Experiment Results
- Concept Matching Experiment Results
- Enhanced models vs Baseline Experiment Results

Ontology Embedding Layer Experiments

The purpose of this experiment was to evaluate several different methods for encoding the ontology into a low-dimensional embedding space that could be used within the enhanced models. The ontology-concept embedding layer is considered a key focus of this dissertation. This layer encodes ontological knowledge represented within a domain ontology into the features for the DDI classification model to learn from. For this study, the Chemical Entities of Biological Interest (ChEBI) ontology (Hill et al, 2013) was selected due to its domain relevance to drug and chemical entities. Several embedding methods were evaluated and compared. The evaluation of each method was conducted on two different tasks.

The first task is defined as the *DDI Detection Test*. The goal of this task is to

determine the presence or absence of a DDI relation between a pair of drug entities. Using a subset of the SemEval DDI benchmark, the test measures the F_1 score in detecting a positive or negative DDI for a given sample sourced from a DDI hold-out set. An 80/20 split of the training data was used to conduct *Task 1-DDI Detection* test.

The second task is defined as the *Ontological-Distance Preservation* Test. The goal of this task is to measure the preservation of relative distances between pairs of embedded drug concepts when compared to their corresponding ontological distance. Intuitively, a quality embedding should position semantically similar concepts near each other within the embedding space, and dissimilar concepts should be positioned further apart. For this test, the ontology serves as the ground truth to evaluate how each embedding method positions concept relative to each other.

To compare the positioning of concepts within the embedding space to the positioning within the ontology, this test first constructs a ranked order of the concepts based on their distances from a selected *reference concept*. Using a pool of 100 randomly sampled drug concepts from the SemEval DDI training set, each concept takes a turn as the *reference concept*. For each *reference concept,* two lists are generated, each with 99 entries. Each entry represents one of the 99 concepts. The concepts are sorted based on their distance from the *reference concept*. Each of the two lists are ordered using a different distance measurement discussed later in this section.

These two ordered lists are then compared to a third ordered list generated using the ontological path distance between the *reference concept* and the other 99 concepts in the pool. The ontological path distance is defined as the sum of vertex traversals required to reach a given concept from the *reference concept* within the ontology graph. An error can be calculated by comparing the ordered lists produced from the embedding space to the list generated using the ontology. This error measures the rank displacement between a concept in one of the two lists generated from the embedding space and the third list generated using the ontology which serves as the ground truth. The difference in the rank (i.e. list position) of a concept in the ontology-generated list to the embedding-generated list represents the error.

Before this error can be computed and the ordered lists generated, each of the 100 concepts are projected onto the embedding space yielding a vector coordinate. The distance

between two concept vectors is measured using two commonly used distance measures. The first measure is Cosine distance which measures the angle between two vectors irrespective of the magnitude of the vectors. Equation 1 shows the cosine distance between two vectors *x* and *y*.

$$\frac{x \cdot y}{\sqrt{x \cdot x}\sqrt{y \cdot y}}$$

Equation 1 - Cosine distance between two vectors x and y

The second measure is Euclidean distance which measures the geometric distance between two vectors. Unlike Cosine distance, Euclidean distance considers the magnitude of the vectors. Equation 2 shows the Euclidean distance between two vectors *x* and *y* where *i* represents the vector component and *n* the dimensionality of the vectors.

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Equation 2 - Euclidean distance between two vectors x and y

Both Cosine and Euclidean distances were used to evaluate the distance from the *reference concept* to the other concepts within the embedding space. Since Cosine and Euclidean distance are known to measure different quality characteristics of the embedding space, both are used in this test. As explained earlier in this section, each ordered list is generated by sorting the concepts based on their distances from the *reference concept*. One list sorts the concepts based on the Cosine distance while the other list is sorted using the Euclidean distance. The two ordered lists can then be compared to a third ordered list generated using the ontology distances to compute the error. This process is repeated for each of the 100 sampled concepts in the pool. The following algorithm was used for this test:

OntologyDistancePreservationTest:

1. P ←Select 100 random drug concepts mentioned in the DDI Training dataset
2. For each concept *c1* in P:
    a. For each concept *c2 in P where c1 ≠ c2:*
        i. d_{onto} ←Compute Ontological distance between *c1* and *c2*
        ii. v_1 ← Embed *c1* using the Embedding Method under evaluation
        iii. v_2 ← Embed *c2* using the Embedding Method under evaluation
        iv. d_{euc} ← Compute Euclidean distance between *v1* and *v2*
        v. d_{cos} ← Compute Cosine distance between *v1* and *v2*
        vi. Add d_{onto} to OntoList
        vii. Add d_{euc} to EucList
        viii. Add d_{cos} to CosList
3. Sort OntoList, EucList, and CosList
4. Euc_Err ← Call ***Compute_Error***(OntoList, EucList)
5. Cos_Err ← Call ***Compute_Error***(OntoList, CosList)

OUTPUT: Average(Euc_Err and Cos_Err)

Compute_Error:

INPUT: List1, List2

1. For each item in List1:
    a. Rank1 ← Get concept Rank in List1
    b. Rank2 ← Get concept Rank in List2
    c. AccumulatedError ← AccumulatedError + $(Rank1 – Rank2)^2$
    d. Count ← Count + 1

OUPUT: AccumulatedError / Count

The Mean Squared Error (MSE) is used to compute the error of the rank displacements. MSE is considered a good error metric for this task because it penalizes large rank displacements more heavily than small rank displacements (Twomey & Smith, 1997; Twomey & Smith, 1995). Since the quality goal of the ontology embedding is defined as positioning semantically similar concepts closer together and dissimilar concepts further away, the MSE is a valid error metric for this objective. Equation 1 shows the definition of MSE used to compute the error.

$$MSE = \frac{1}{n}\sum_{i=0}^{n}(X_i - Y_i)^2$$

Equation 3 - Mean Absolute Error where *X* represents the rank position for the *i*th concept from the embedding space and *Y* represents the correct rank position of the *i*th concept using the Ontology; *n* represents the total number of concepts in the ordered lists (i.e. 99)

The accumulated MSE for the Euclidean and Cosine lists are averaged and normalized for the entire pool of concepts. Table 3 shows the computed error under the *Task 2* column for each of the following embedding methods evaluated in this study:

1. Classic Embedding – uses a dense layer with an identity function that learns the distribution of the concept vector embedding space based on the training task
2. Concept Identity Vector Encoding – constructs a k-hot *identity vector* for a concept where each feature position represents a characteristic of the concept within the ontology. (See *Chapter 3 Methodology* for definition of identity vector)
3. Autoencoding of Concept Identity Vector – compresses a concept *identity vector* to a low-dimensional space using an unsupervised input reconstruction model
4. Ontologically Learned Identity Vectors (OLIV) – learns a concept embedding by fitting an unsupervised model that is trained to predict the identity vector for a concept. (See *Chapter 3 Methodology* for detailed description).

See *Appendix B* for a detailed description of each embedding method evaluated in this experiment.

	Method	Task 1: DDI Detection Test F_1	Task 2: Ontology distance Test ME
1.	Classic Embedding	0.093	0.43
2.	Concept Identity Vector (CIV)	0.135	0.33
3.	Autoencoded CIV	0.110	0.50
4.	OLIV	**0.350**	**0.24**

Table 4 - Ontology Embedding Performance Results

Experiment Findings

These experiments were successful in helping to identify the best concept embedding method along with the optimal hyperparameters. Error analysis was conducted to understand the cause for the relatively low scores by all the embedding methods on Task 1. The error analysis concluded that an essential aspect of determining the presence of a DDI is based on linguistic features such as negation which are absent in this test. Since the

models trained for Task 1 only included the embedded drug concepts and no other syntactic or structural features, the models perform poorly when compared to the feature-rich models evaluated later in this chapter.

The Ontology-learned Identity Vector (OLIV) demonstrated better performance to the alternative embedding methods on both the DDI Detection (Task 1) and Ontology-distance test (Task 2). This result aligns with the intuition that guided the approach. This technique, as detailed in *Chapter 3 Methodology - Ontology Embedding Method* section, leveraged the ontology by resolving an identity vector for each concept and learned an embedding through the training of a model that predicts the identity vector for a given concept. This approach differs from other corpus-guided word embedding methods (Bengio et al, 2003, Mikolov et al, 2013, Pennington et al, 2014, Hill et al, 2016) because it uses the ontology to guide the training of the embedding. By leveraging a concept's ancestors and semantically related concepts within the ontology, this approach produces a richer concept embedding that can present additional patterns, otherwise hidden, to the classification layer of the model. Additionally, unlike other approaches (Lamurias & Couto, 2019), this approach demonstrated that it could scale to embed the entire vocabulary of ChEBI concepts making it a better choice for real world application.

To identify the optimal dimensionality of the embedding space, the OLIV embedding algorithm was run using 50, 100, 200, and 300 dimensions. For each embedding dimension, the Zhang et al (2018) baseline model was extended to use the trained embedding layer and the performance impact of each embedding dimension was compared. The 300-dimensional embedding space demonstrated the largest impact (+.06) to F_1 accuracy when compared to the other dimensions. See *Ontology Enhanced Model vs Baseline Experiments* section for a detailed description of the controlled experiment results.

Concept Matching Experiments

The purpose of this experiment is to evaluate the performance of 3 different concept matching methods. Concept matching is an important supporting function of the Ontology Embedding Layer that converts the text representation of drug entities or other words and phrases within the input text into normalized concepts matched within the ontology. This experiment evaluated 3 different implementations of concept matching including: Basic

String-matching, an implementation of the Concept Mapper, and a Finite-State Transducer implementation.

Each concept matching method was evaluated for concept matching accuracy, memory footprint, and runtime performance. Each method was evaluated using the DDI training dataset using a ChEBI dictionary generated from the ChEBI ontology. Each method uses its own data structure to represent the dictionary and its own matching algorithm to identify the matching concepts and offset spans over the text. The methods were configured to produce the longest matching concept over the drug entity text provided by the DDI training dataset. Several metrics were collected for each run of the method. Since the algorithms used for concept matching are deterministic (not stochastic) the accuracy metric was consistent across multiple trial runs. To gather the matching runtime performance and dictionary load time of each method, 3 trial runs were conducted for each method and the median metric was selected. The DDI training set, which includes 27,784 samples featuring 14,748 drug entities, was used to measure the runtime performance of each concept matching method.

Results

Table 5 shows the results for each concept matching method using the SemEval 2013 DDI training dataset. The concept match accuracy was computed based on the accumulated average coverage for each drug entity in the dataset. The coverage metric measures the percentage of the drug entity span that was matched by the concept matching method. Equation 4 shows the equation for computing the accumulated coverage accuracy used in this experiment. The matching algorithm runtime was measured by the elapsed number of milliseconds (ms) to process the entire dataset. Additionally, the memory footprint was measured by checking the change in memory increase before and after the dictionary is loaded.

$$\frac{\sum_1^n \frac{characters\ matched}{total\ characters}}{n}$$

Equation 4 - Match Coverage accuracy equation where $\boldsymbol{n}$ represents the number of drug entities in the SemEval 2013 DDI training dataset

	Method	Match Accuracy	Dictionary Load Time	Match Time	Memory Footprint
1.	Basic String-Matching	0.603	2816ms	**10ms**	**138MB**
2.	Concept Mapper (based on Apache Concept Mapper)	0.832	**1082ms**	61ms	186MB
3.	Concept Matcher using Finite State Transducer	**0.833**	6694ms	75ms	2GB

Table 5 - Concept Matching Quality Test

Experiment Findings

The results demonstrated that Concept Mapper and Concept Matcher methods achieved the best match accuracy, while the Basic String-Matching method achieved the best match time and demonstrated the lowest memory footprint of the 3 methods. The match accuracy, dictionary load time, and match time results were in line with expectations. One unexpected result was the large memory footprint for the Concept Matcher (using Finite State Transducer) method. The Finite State Transducer constructs a data structure that avoids storing identical subsequences of text by representing sequences as character state transitions, while the Basic String-Matching and the Concept Mapper both store every string in the dictionary at least one time. The Concept Mapper additionally utilizes several TreeMaps used to organize the dictionary entries by length and then by the first token of a phrase (Tanenblatt et al, 2010). The Basic String Method was used as a baseline comparison using a HashMap to perform string looks ups in a greedy fashion looking for the longest string match first before proceeding to substrings. The basic string-match demonstrated the worse match coverage accuracy. Overall, these results met the expectations for the experiment and Concept Mapper was chosen over Concept Matcher as the method of choice for concept matching due to its better match time and significantly smaller memory footprint while achieving high accuracy on the concept matching quality tests. The remainder of this study utilized Concept Mapper to support the ontological embedding method results discussed in the next section.

Ontology Enhanced Model vs Baseline Experiments

The purpose of this experiment is to evaluate the performance impact of the ontology embedding layer using two different state-of-the-art baseline models as a control. Using the provided source code for each model, the models were trained and evaluated

against the SemEval 2013 DDI benchmark to establish the baseline performance. The DDI training dataset was partitioned into a training and validation set using an 80/20 split. The same training and validation split were used to fit and evaluate each model configurations for 100 epochs using SGD with minibatch size of 64. The training and validation of a model configuration represents a trial run. The best model from each trial run was selected by evaluating the model performance on the validation set on each epoch and saving the weights for the highest F_1 validation score. The best performing model from each trial run was then evaluated on the DDI test set and the F_1 performance is reported later in this section.

The random seed used for model weight initialization was changed on each trial run to evaluate the reproducibility of model performance and avoid the known issue of "lucky" weight initialization that can position the model weights near the global minima of the loss function. Each baseline model was evaluated using the published hyperparameters and model architecture.

The first baseline model used is the Zhang Y. et al (2018) Hierarchical RNN model. This model achieved the 3rd highest reported performance results on the SemEval 2013 DDI benchmark. The reported results were reproduced during this study, achieving an average max F_1 of .72 when evaluated against the Test dataset. The results will refer to this model as the *Zhang* baseline.

The second baseline model used is the Lamurias & Couto (2019) *Zhang+BO-LSTM* model. This model achieved the 2nd highest reported performance results on the SemEval 2013 DDI benchmark. The reported results were reproduced during this study, achieving an average best F_1 of .75 when evaluated against the Test dataset. The results will refer to this model as the *Zhang+BO-LSTM* baseline.

At the time of this study, Zheng W. et al (2017) reported the highest performance for the SemEval 2013 DDI benchmark of .77 best F_1 score. However, as noted in *Chapter 2 – Literature Review*, the model source code is not available for review and prior attempts by other studies failed to reproduce those results with a similar model.

After establishing and reproducing the baseline model accuracy performance using the DDI benchmark, an extension, referred to as a *treatment*, is made to each model to incorporate the Ontology Embedding layer with the OLIV trained ChEBI concept vectors.

This treatment to each baseline model is performed in a controlled fashion by preserving all other hyperparameters, model architecture, and input features intact with exception to the treated drug entities to be embedded.

Two different treatments were defined for this experiment. The first treatment, labeled *Treatment A*, replaces the word vector embedding method used by both baseline models to embed the drug entities with the ontology embedding method presented in this study. The ontology embedding layer method and related parameters are independently tuned and optimized using two comparative test experiments discussed in the earlier section: *Ontology Embedding Layer Experiments*. The treatment is applied by incorporating the ontology embedding layer to the model and initialized with the pre-trained embedding weights.

The second treatment, labeled *Treatment B*, modifies the *Zhang+BO-LSTM* concept ancestor channel of the model to use the embedded ChEBI concepts produced by this study instead of the classic embedding layer used by the *Zhang+BO-LSTM* model. Treatment B is only applicable to the *Zhang+BO-LSTM* baseline model since it is the only baseline model that includes a concept feature channel representing the *is-a* ancestors for each drug entity candidate.

Figure 26 depicts the locations within the baseline model architecture where the treatment is applied. *Treatment A* is applied to both the *Zhang* baseline model and *Zhang+BO-LSTM*. *Treatment B* is applied only to *Zhang+BO-LSTM*.

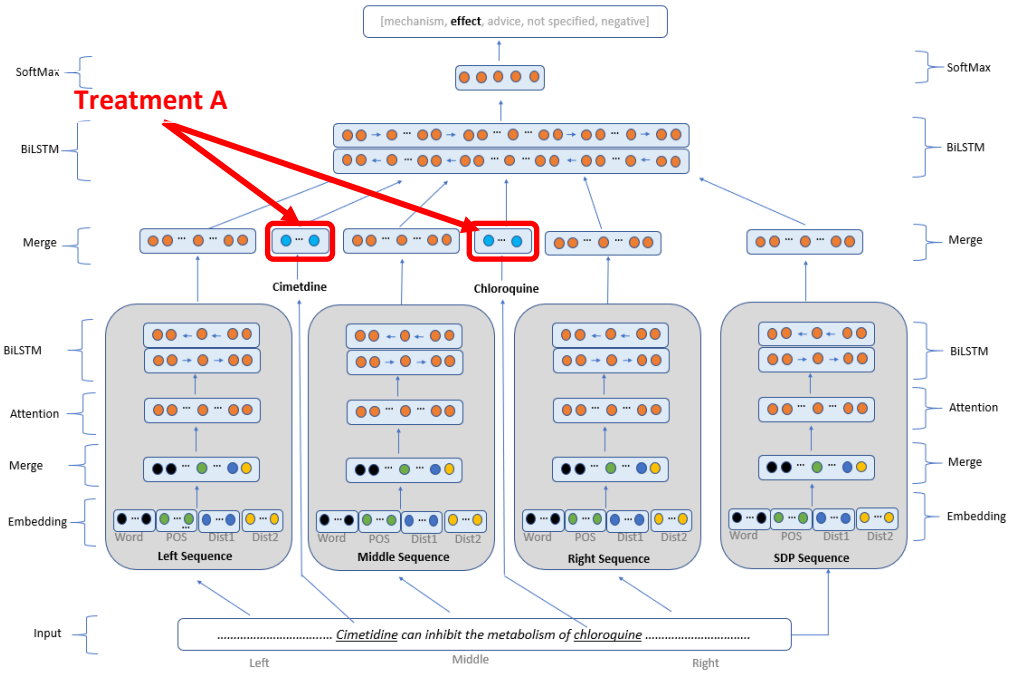

Figure 26 - Location of Treatment A in *Zhang* baseline model

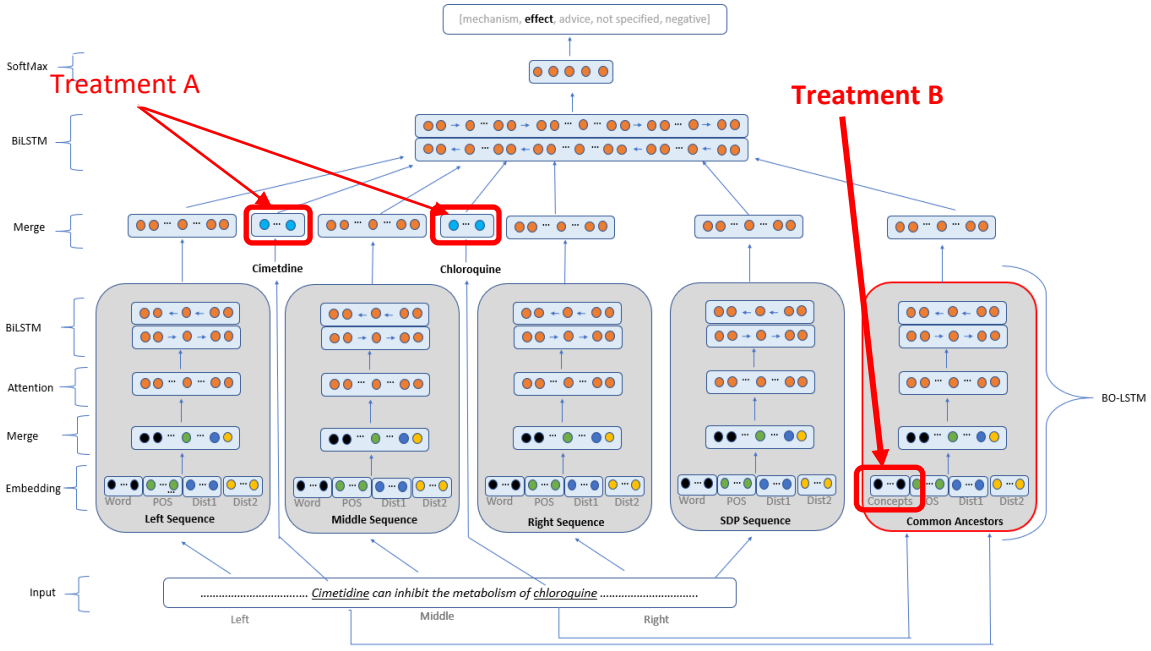

Figure 27 - Location of Treatment A and B in *Zhang+BO-LSTM* model

Figure 28 through Figure 33 shows the Keras model summary for the Zhang+BO-LSTM baseline model which extends the Zhang baseline model with the additional *Common Ancestors* input channel. Figure 28 shows each of the inputs to the model including the words, part-of-speech, and the positional distance from the drug entities for each of the subsequences left, middle, right, and SDP channels.

Layer (type)	Output Shape	Param #	Connected to
input_word_0 (InputLayer)	(None, 98)	0	
input_pos_0 (InputLayer)	(None, 98)	0	
input_dis1_0 (InputLayer)	(None, 98)	0	
input_dis2_0 (InputLayer)	(None, 98)	0	
input_entity_0 (InputLayer)	(None, 1)	0	
input_word_1 (InputLayer)	(None, 107)	0	
input_pos_1 (InputLayer)	(None, 107)	0	
input_dis1_1 (InputLayer)	(None, 107)	0	
input_dis2_1 (InputLayer)	(None, 107)	0	
input_shortest_word (InputLayer	(None, 12)	0	
input_shortest_pos (InputLayer)	(None, 12)	0	
shortest_input_dis1 (InputLayer	(None, 12)	0	
shortest_input_dis2 (InputLayer	(None, 12)	0	
input_entity_1 (InputLayer)	(None, 1)	0	
input_word_2 (InputLayer)	(None, 144)	0	
input_pos_2 (InputLayer)	(None, 144)	0	
input_dis1_2 (InputLayer)	(None, 144)	0	
input_dis2_2 (InputLayer)	(None, 144)	0	
common_input (InputLayer)	(None, 12)	0	

Figure 28 – Model Inputs for the Zhang+BO-LSTM baseline model

Figure 29 shows the embedding layers used for word, part-of-speech, distance, and the BO-LSTM ancestor concepts. The changes made to the model when applying *Treatment A* and *B* are focused in this area of the model. Figure 30 shows where the embedded input features are concatenated together to form separate feature groups.

embedding_1 (Embedding)	multiple	2685600	input_entity_0[0][0] input_entity_1[0][0] input_word_0[0][0] input_word_1[0][0] input_word_2[0][0] input_shortest_word[0][0]
embedding_3 (Embedding)	multiple	320	input_pos_0[0][0] input_pos_1[0][0] input_pos_2[0][0] input_shortest_pos[0][0]
embedding_2 (Embedding)	multiple	6010	input_dis1_0[0][0] input_dis2_0[0][0] input_dis1_1[0][0] input_dis2_1[0][0] input_dis1_2[0][0] input_dis2_2[0][0] shortest_input_dis1[0][0] shortest_input_dis2[0][0]
embedding_4 (Embedding)	(None, 12, 300)	2082600	common_input[0][0]

Figure 29 - Embedding layers for Zhang+BO-LSTM baseline model

concatenate_1 (Concatenate)	(None, 98, 330)	0	embedding_1[2][0] embedding_3[0][0] embedding_2[0][0] embedding_2[1][0]
concatenate_2 (Concatenate)	(None, 107, 330)	0	embedding_1[3][0] embedding_3[1][0] embedding_2[2][0] embedding_2[3][0]
concatenate_4 (Concatenate)	(None, 12, 330)	0	embedding_1[5][0] embedding_3[3][0] embedding_2[6][0] embedding_2[7][0]
concatenate_3 (Concatenate)	(None, 144, 330)	0	embedding_1[4][0] embedding_3[2][0] embedding_2[4][0] embedding_2[5][0]
concatenate_5 (Concatenate)	(None, 12, 330)	0	embedding_4[0][0] embedding_3[3][0] embedding_2[6][0] embedding_2[7][0]
dropout_1 (Dropout)	(None, 98, 330)	0	concatenate_1[0][0]
dropout_2 (Dropout)	(None, 107, 330)	0	concatenate_2[0][0]
dropout_4 (Dropout)	(None, 12, 330)	0	concatenate_4[0][0]
dropout_3 (Dropout)	(None, 144, 330)	0	concatenate_3[0][0]
dropout_5 (Dropout)	(None, 12, 330)	0	concatenate_5[0][0]

Figure 30 - Feature group concatenation and dropout layers for Zhang+BO-LSTM baseline model

lstm_1 (LSTM)	(None, 150)	288600	dropout_1[0][0]
lstm_2 (LSTM)	(None, 150)	288600	dropout_1[0][0]
lstm_3 (LSTM)	(None, 150)	288600	dropout_2[0][0]
lstm_4 (LSTM)	(None, 150)	288600	dropout_2[0][0]
lstm_7 (LSTM)	(None, 150)	288600	dropout_4[0][0]
lstm_8 (LSTM)	(None, 150)	288600	dropout_4[0][0]
lstm_5 (LSTM)	(None, 150)	288600	dropout_3[0][0]
lstm_6 (LSTM)	(None, 150)	288600	dropout_3[0][0]
lstm_9 (LSTM)	(None, 150)	288600	dropout_5[0][0]
lstm_10 (LSTM)	(None, 150)	288600	dropout_5[0][0]
concatenate_6 (Concatenate)	(None, 300)	0	lstm_1[0][0] lstm_2[0][0]
concatenate_7 (Concatenate)	(None, 300)	0	lstm_3[0][0] lstm_4[0][0]
concatenate_9 (Concatenate)	(None, 300)	0	lstm_7[0][0] lstm_8[0][0]
concatenate_8 (Concatenate)	(None, 300)	0	lstm_5[0][0] lstm_6[0][0]
concatenate_10 (Concatenate)	(None, 300)	0	lstm_9[0][0] lstm_10[0][0]

Figure 31 - Forward and Back LSTM and Concatenation of Forward and Back for Zhang+BO-LSTM baseline model

Figure 31 shows the LSTM layers for each of the feature groups. There are two LSTM layers used, one that processes the sequence forward and the other in reverse (i.e. backwards). The concatenation of the forward and backwards LSTM is known as a Bidirectional LSTM. Figure 32 shows the reshaping of each of the resulting outputs from the first layer of LSTM, which are concatenated and passed into a final Bidirectional LSTM layer. The final output layers for the model are shown in Figure 33 along with the total number of trainable model parameters.

reshape_1 (Reshape)	(None, 1, 300)	0	concatenate_6[0][0]
reshape_2 (Reshape)	(None, 1, 300)	0	concatenate_7[0][0]
reshape_4 (Reshape)	(None, 1, 300)	0	concatenate_9[0][0]
reshape_3 (Reshape)	(None, 1, 300)	0	concatenate_8[0][0]
reshape_5 (Reshape)	(None, 1, 300)	0	concatenate_10[0][0]
concatenate_11 (Concatenate)	(None, 1, 2100)	0	reshape_1[0][0] embedding_1[0][0] reshape_2[0][0] reshape_4[0][0] embedding_1[1][0] reshape_3[0][0] reshape_5[0][0]
lstm_11 (LSTM)	(None, 150)	1350600	concatenate_11[0][0]
lstm_12 (LSTM)	(None, 150)	1350600	concatenate_11[0][0]
concatenate_12 (Concatenate)	(None, 300)	0	lstm_11[0][0] lstm_12[0][0]
dropout_6 (Dropout)	(None, 300)	0	concatenate_12[0][0]

Figure 32 - Reshaping and concatenation of all channels and final LSTM Forward and Back Concatenation for Zhang+BO-LSTM baseline model

dense_1 (Dense)	(None, 5)	1505	dropout_6[0][0]
activation_1 (Activation)	(None, 5)	0	dense_1[0][0]

Total params: 10,363,235
Trainable params: 10,363,235
Non-trainable params: 0

Figure 33 - Output layer for Zhang+BO-LSTM baseline model

Figure 34 shows the original BO-LSTM embedding layer taken from the model summary above (top) and the changes made to the Zhang+BO-LSTM model when applying the OLIV trained embedding layer to the two drug entities and the common ancestors input channels (bottom).

embedding_4 (Embedding)	(None, 12, 300)	2082600	common_input[0][0]
OLIV_embedding_4 (Embedding)	multiple	2097900	input_entity_0[0][0] input_entity_1[0][0] common_input[0][0]

Figure 34 - (Top) Compares the original embedding used for common ancestors. (Bottom) shows the OLIV embedding layer used to embed both drug entities labeled *input_entity_0* and *input_entity_1* and the common ancestors labeled *common_input.*

Results

The results of evaluating each model configuration on the SemEval 2013 DDI Test set is shown in Table 6. The best performing model from each trial run was selected based on the best validation F_1 score during training. Table 6 shows the scores for each trial run and provides the min, max, mean, and standard deviation measures for the 10 trials runs.

The results of the experiments demonstrated that applying *Treatment A* to the *Zhang* baseline model produced the overall highest F_1 performance of .781. The second overall highest observed F_1 of .780 was produced by applying *Treatment A* and *B* to the *Zhang+BO-LSTM* baseline model. These F_1 scores represent the highest reported F_1 scores on the Sem Eval 2013 DDI benchmark at the time of this study. Additionally, the *Zhang* with *Treatment A* model outperformed the other models on min, max, and mean F_1 measures.

Both *Treatment A* and *B* when applied to the corresponding baseline model outperformed the baseline across min, max, and mean F_1 measures. Interestingly, Treatment A applied to the Zhang baseline model outperformed all other models including slightly better results than *Zhang+BO-LSTM with Treatment A & B.* This finding may indicate that the additional model complexity of the *BO-LSTM Ancestors* channel does not provide additional informative patterns to the model classifier beyond what the OLIV embedded concepts provide to a simpler model such as the *Zhang* baseline.

Since a higher accuracy is achieved when applying the corresponding treatment to the baseline model, this implies that the OLIV Ontology Embedding approach, presented in this study, contributes an improvement to the model accuracy for this relation classification task.

	Zhang with OLIV Treatment A	**Zhang Baseline**	**Zhang+BO-LSTM with OLIV Treatment A & B**	**Zhang+BO-LSTM Baseline**
Trial #	F_1	F_1	F_1	F_1
1	0.77415	0.74408	0.77958	0.75901
2	0.77136	0.74769	0.76965	0.75325
3	0.77830	0.73725	0.77497	0.76368
4	0.77596	0.74074	0.78065	0.76201
5	0.77010	0.75121	0.77325	0.76558
6	0.77098	0.75434	0.77453	0.75984
7	0.77758	0.74427	0.77320	0.76169
8	0.78167	0.74916	0.77402	0.76014
9	0.77601	0.73666	0.77203	0.76764
10	0.77612	0.73560	0.77521	0.76276
Min	**0.77010**	0.73560	0.76965	0.75325
Max	**0.78167**	0.75434	0.78065	0.76764
Mean	**0.77522**	0.74410	0.774709	0.76156
StdDev	0.00345	0.00616	0.003114	0.00374

Table 6 - Trial results and statistics using SemEval DDI 2013 Test dataset

Error Analysis

To better understand the effects of the new ontology embedding layer within the improved model, an error analysis was conducted using the best performing model run for each of the model variants considered during the experimentation. The analysis was focused on intersecting the correct and wrong prediction on the DDI test set for each model. Using this approach, it is possible to reveal which test samples are uniquely predicted correctly by each model configuration. Additional analysis can be conducted on test instances that were incorrectly classified by all models to better understand the nature of the remaining problem space.

Figure 35 and Figure 36 show a Venn diagram indicating the set of correct and incorrect classifications. Using this visualization, the number of unique correct and incorrect samples along with the intersection of correct and incorrect samples can be observed. Figure 35 shows a comparison of the treated and untreated *Zhang* baseline model with the Ontology Embedding method presented in this study (i.e. *Treatment A*). The treated model had 194 unique correct classifications that the baseline model misclassified

and 149 unique incorrect answers that the baseline model classified correctly. Both the treated and untreated model correctly classified the same 3,324 samples and incorrectly classified the same 339 samples.

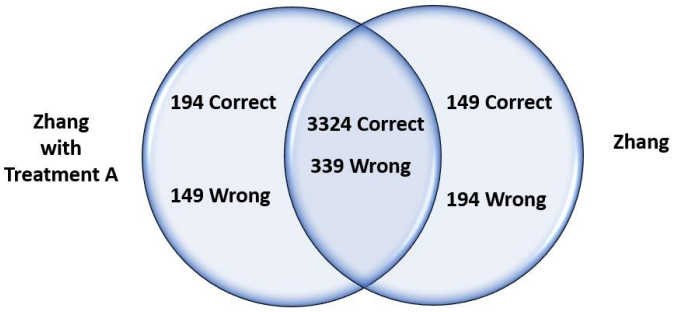

Figure 35 - Error Analysis of Zhang with Treatment A compared to Zhang baseline

Figure 36 shows the comparison of the treated and untreated *Zhang+BO-LSTM* baseline model. The treated model correctly classified 167 samples that were misclassified by the baseline model and misclassified 129 samples that were correctly classified by the baseline model.

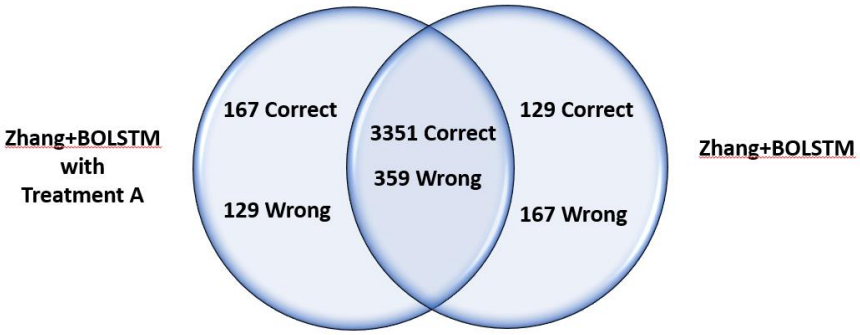

Figure 36 - Error Analysis of *Zhang+BO-LSTM* with *Treatment A* compared to *Zhang* baseline

These results demonstrate additional opportunity for accuracy improvement if the methods were tuned to capture more of the unique correct classifications by the baseline models. Additional analysis can explore common patterns of the unique correct and incorrect classifications for each model configuration to gain additional intuition that can led to future model improvements.

Additional Experiments

During this research study, a Java implementation of several different DDI Classifier models along with a collection of utilities that including Ontology parser, NLP parser, DDI Dataset parser, Concept matchers, feature extractors and vectorizers, and model source code. See *Appendix D* for information on obtaining access to the source code and resources. This section reviews additional experiments for models implemented using Java and DeepLearning4J along with a framework implemented during this study to facilitate the training and evaluation of the SemEval 2013 DDI benchmark.

Several different alternative model configurations were explored including an implementation of Lamurias & Couto *BO-LSTM* model. The performance of this model against the DDI benchmark was .57 F_1, underperforming the *Zhang+BO-LSTM* baseline model.

Applying insights and learnings from the previous experiments, an additional experiment was conducted to explore a model that included additional feature groups and layers. The first addition included the UMLS ontology as an additional ontology embedding layer that can replace the general word vector feature group and pre-trained using the ontology OLIV embedding method that achieved the best performance. The intuition for this addition involves the ability to normalize phrases within the sentence to concepts. This can also serve as a noise filter by excluding words that are not defined within the ontology and therefore assumed to not be important within the domain. Results of including UMLS as a feature embedding did not demonstrate a competitive accuracy improvement. The results for models featuring UMLS as the primary feature along with a combination of Part-of-speech, Word Distance from Drug Entity ranged between .459 and .669 F_1 accuracy.

Another feature idea explored whether the structural elements of a sentence was important and predictive within the DDI task. A feature called *Structure* was implemented by masking all words, numbers, and drug entity names with symbols. The feature included 14 different possible symbols including: *WORD*, *COMMA*, *PAREN*, *WORD-INPAREN*, *DRUG1*, and *DRUG2* among others. The following is an example of a sentence that has been included into the *Structure* feature representation:

[WORD]|[WORD]|[DRUG1]|[AND]|[WORD]|DRUG2]|[PERIOD]|

The intuition behind the Structure feature is that the occurrence of the two drugs within the sentential structure is an important predictor for negative DDI relations. When the pair of drugs are both located within the same coordinated conjunction list, the DDI relation is always negative since the items within the conjunction lists are not being compared to each other, but rather to another drug outside of the coordinate list. Similarly, clues such the pair of drugs occurring immediately after the other within parenthesis is an indication of a synonym of the other and therefore is also always a negative DDI. The Structure feature was evaluated within several different model configurations that included other features such as Word Vectors, Part-of-Speech, Distance to Drug entity, and with the UMLS embedding defined earlier. The results for models that included the Structure feature ranged from .458 to .460. This feature also underperformed expectations.

Various other model configurations were evaluated including: pre and post merging feature groups, full and partial masking of drug entities mentioned in text, different number of LSTM units used within the hidden layers, and different learning rates. From these various configurations, the masking of the drug entities demonstrated the most impact to the F_1 metric of a model. This result aligned with expectations given the intuition that for many DDI samples, the drug names not under consideration in a given sample do not contribute to, but rather detract from the performance of the model given the additional sequence length and noise introduced by the additional drug names. The optimal masking configuration included using one mask symbol replacement for the two drugs under consideration and a different mask symbol for the other drugs mentioned within the sample text. Note that masking was only performed for feature groups that were not undergoing an ontology embedding but rather were using a general word vector embedding. The unmasked drug name is required by the ontology embedding layer presented in this study.

Another model configuration idea explored was a multi-staged model where each stage was optimized towards classifying a subset of the DDI classes. The intuition behind this method is based on the insight gained by carefully studying the DDI task and the meaning behind each class label. The presence of interclass relationships may be confusing the learning algorithm since similar patterns are encountered for different DDI classes. For example, DDI relations labeled as *Advice* may include the mention of an *Effect* or *Mechanism* with the addition of some advice given to mitigate the DDI. This means that

the model will see very similar patterns that sometimes are considered Advice and other times they are considered a specific DDI type. Another example of these interclass relations include *Effect* and *Mechanism* are both a type of DDI relation while *Int* is also a DDI relation but was not specifically qualified as *Effect* or *Mechanism*. This implies a hierarchical classification scheme. These heterogenous class labels can be partitioned in a way that allows a more optimal model to perform class partitioning and leverages these interclass relations to its advantage. This model can decompose into a stage that classifies Positive vs Negative DDI, a stage that classifies between Int, Effect, and Mechanism, and a stage that classifies the presence of Advice within the text. This decomposition maximizes the amount of training data that can be used to learn the boundary between these classes for each respective stage. The model is trained as 3 separate classifiers that are run in sequential order. This model configuration demonstrated an F_1 score of .764. This result underperformed expectations but additional error analysis and fine-tuning may yield better results.

Another investigation involved the significance of the SDP feature channel within the models. Both the Zhang and Lamurias & Couto models leverage an SDP channel that is used to reduce the sentence to the shortest dependency path between the two drug entities under consideration. Intuitively this feature appears to be a very important feature since it discards irrelevant words that are not part of the linguistic parse between the two drug words of interest. This also has the added benefit of reducing the sequence length presented to the RNN/LSTM layer which is known to improve the accuracy and avoids exploding and vanishing gradient problems during back propagation of deep hidden layers. Intuitively, SDP should be a primary feature within the model that contribute a significant performance improvement. However, during experimentation, models featuring SDP word channels without the presence of other input channels that included the full sentence sequence underperformed significantly. Error analysis identified the root cause as the absence of important contextual trigger words, such as negation, from the SDP parse returned by parser such as Stanford Parser and SpaCy. These trigger words are critically important to determining the separation between Positive and Negative DDI classes.

After experimenting with different model architectures, Zhang's hierarchical RNN architecture consistently demonstrated the best performance. Additional experiments with

Zhang's hierarchical RNN such as removing the SDP feature channel (-.006 to F_1) and removing the Attention Pooling layer (-.009 to F_1) resulted in minimal impact to the model performance. This demonstrates that the hierarchical RNN architecture is robust to the absence of features that have been reported in the literature as contributing significant improvements to accuracy for relation extraction tasks. When comparing the performance of the hierarchical RNN architecture to a baseline RNN model that includes the same features (Word, Part-of-Speech, and Distance to Drug entity), the hierarchical RNN performs +.166 F_1 over the baseline. These results demonstrate that modeling a solution architecture based on the structure of the problem is effective. The hierarchical RNN is modeled as 3 separate input channels that each represents a subsequence of the sample text. The left channel represents the sequence of words from the beginning of the sentence to the first drug entity, the middle channel represents the sequence between the first and second drug entity, and the right channel represents the sequence of words from the second drug entity to the end of the sample text. Each channel is processed by an independent Bidirectional LSTM. The output of the LSTM layer is merged and processed by a final Bidirectional LSTM layer before a final Softmax Classification layer.

Chapter 5

Conclusion

Relation extraction remains a complex natural language task. Modern neural relation classifiers have demonstrated significant improvements to the accuracy performance of relation classification tasks. Most recently, the use of external knowledge resources, such as ontologies, together with Recurrent Neural Network architectures have demonstrated state-of-the-art performance on these tasks. This study examined the impact of expanding the use of ontologies within neural relation classifiers to reveal additional hidden patterns not available within the training data. This study presented an ontology concept embedding layer that can be trained on the taxonomic and semantic axioms defined within a domain ontology. The results gathered during this study support the primary hypothesis that incorporating ontological knowledge in addition to the training data, results in a repeatable and statistically significant accuracy improvement.

Using the SemEval 2013 DDI dataset as an established benchmark for measuring relation extraction performance, the Ontologically Learned Identity Vector (OLIV) embedding method demonstrated higher F_1 accuracy when applied to two different state-of-the-art baseline models. This result demonstrates a new state-of-the-art performance for the DDI benchmark. Additionally, the pre-trained OLIV embedding method required significantly less trainable model parameters when compared to the corresponding baseline model. This results in a smaller model that can be trained faster and less likely to overfit the training data.

Implications

Complex fields such as medicine and pharmacology require deep technical knowledge of the domain's concepts and the semantic relations between the concepts. Human subject matter experts are trained for many years to learn and master these domains. Training machine learning models to match or exceed human performance on tasks within these complex domains has proven to be very difficult. The use of neural networks has demonstrated some improvements in the performance of these tasks. High-Performance computing (HPC) and the use of GPU have allowed neural networks to reach billions of trainable parameters. Unfortunately, the amount of training data required to train models

of such capacity is scarce and difficult to acquire when operating within regulated industries such as healthcare and life sciences research.

This presents a new challenge that cannot be overcome by improved hardware, but instead requires improved algorithms and methods. An additional complication arises from the rare class problem that is frequently present within these complex domain tasks. Rare events, such as uncommon diseases, present a challenge to the model design due to significant class imbalances present in the available training data. Since obtaining additional computing capacity and massive data sets are not viable options, the need for new methods are required to deliver improved accuracy on these complex tasks.

This study contributes to several new areas of research that helps to address these constraints by leveraging ontologies as an external knowledge source that is used to augment training data. Ontologies already serve as a rich source of expertly curated knowledge about a domain. By learning an embedding based on all the concepts and relations within the ontology, this study demonstrated an improvement to the accuracy of a drug-drug interaction classification model with fewer trainable parameters using the same amount of training data.

Future Work

Natural Language Processing remains an active area of research. Relation extraction is a fundamental NLP task that enables higher-order Language Understanding tasks. The results of this study open new areas of future research in neural relation extraction. The following is a list of additional research areas:

- Several different ontology-based embedding methods were explored during the course of this study including using different neural embedding model architectures trained to either reconstruct an identity vector of ontological features or to predict similar concepts within the ontology graph across different relationship types. Future work can explore alternative methods to training the embedding layer to capture additional salient information within the ontology such as semantic types and other domain specific attributes.

- During this study, the ChEBI served as the primary ontology used for training the embedding layer given its relevance to the DDI task. Additional experiments were performed using UMLS ontology as an additional ontology layer. Unlike the ChEBI ontology, it was trained using an Embedding layer together with the DDI classifier. Additional work can explore using the OLIV embedding method on the UMLS ontology.
- Ontology-concept embedding depends on effective Concept Matching to resolve the best concept match. This study evaluated 3 different concept matching methods including an implementation of Concept Mapper which demonstrated the best performance during concept matching experiments. During error analysis, a significant percentage of False-Negative were attributed to non-contiguous drug entity spans. Future work can explore the impact of a non-contiguous span matching algorithm to the immediate concept matching performance as well as the overall DDI task performance.
- Several alternative model architectures were explored during this study that included different types of layers including Bidirectional LSTM, Dense, Max, Average, and Attentive Pooling. The Hierarchical RNN model architecture presented by Zhang et al (2018) demonstrates effectiveness towards the relation classification tasks studied in this dissertation. Future exploration of new model architectures may lead to greater improvements in accuracy on similar relation extraction tasks.
- Shortest-dependency path (SDP) has been utilized by many relation extraction models reviewed in this study. The intuition for using SDP relies on the assumption that a sentence can be simplified using linguistic rules to discard irrelevant words while preserving the words that connect a pair of entities to each other. During feature significance analysis, the SDP feature performed poorly when compared to other features. Error analysis revealed the likely cause for this unexpected result as the absence of critical modifiers, such as negation terms, that are omitted from the shortest dependency path. Additional exploration of a dependency path method that preserves the salient terms within the sentence can help to simplify models such as (Zhang Y. et al, 2018) and (Xu B. et al, 2018) which include full sentence sequences

in addition to SDP. Models such as (Lamurias & Couto, 2019) that solely rely on SDP-based features miss out on important clues in classifying DDI relations and therefore significantly underperform comparable models. Work by (Liu, Y., Wei, F., Li, S., Ji, H., Zhou, M., & Wang, H, 2015) proposes a novel neural dependency parser that can be trained together with the relation classifier and learn the salient features that support the classification task. (Kiperwasser, & Goldberg, 2016) propose a standalone neural dependency parser that can be used in place of parsing utilities. The *learned dependency* approach is conceptually similar to Attention-based methods (Bahdanau et al, 2016) that seek to learn an importance weight for words in neural translation tasks. Incorporating these approaches can lead to a simplified model that does not depend on external utilities for parsing and is more resilient to learning the salient features for a specific relation extraction task.

- Due to the heterogeneity and implicit interclass relationships between the DDI class types, new approaches may benefit from leveraging multi-staged model architectures, hierarchical classification layers, or multi-task learning. These approaches may help improve the model's ability to learn from overlapping patterns between related class types. This approach may also help mitigate the class imbalance problem present in the DDI dataset. For example, the *Int* class, which denotes unspecified DDI interaction type, represents only 0.6% (189 of 27,756) of the training dataset. However, the Int class can be thought of as the generic type for an *Effect* or *Mechanism* class. Another approach may be to frame the classification task as a multi-task problem where one or more class labels are assigned to each sample. Using a pre-processor to augment the provided training dataset, the *Int* label can be included to each *Effect* and *Mech* label. Similarly, the *Advice* label can be added to each Int labeled sample since a DDI interaction is implied on all samples classified as *Advice*. Applying these insights to the model design may improve the model performance through transfer learning on a larger sample set.

References

Henry, S., Buchan, K., Filannino, M., Stubbs, A., & Uzuner, O. (2020). 2018 n2c2 shared task on adverse drug events and medication extraction in electronic health records. Journal of the American Medical Informatics Association, 27(1), 3-12.

Sousa, D., & Couto, F. M. (2020, April). BiOnt: deep learning using multiple biomedical ontologies for relation extraction. In European Conference on Information Retrieval (pp. 367-374). Springer, Cham.

Couto, F., & Lamurias, A. (2019). Semantic similarity definition. Encyclopedia of bioinformatics and computational biology, 1.

Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Séaghdha, D. O., Padó, S., ... & Szpakowicz, S. (2019). Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. arXiv preprint arXiv:1911.10422.

Lamurias, A., Sousa, D., Clarke, L. A., & Couto, F. M. (2019). BO-LSTM: classifying relations via long short-term memory networks along biomedical ontologies. BMC bioinformatics, 20(1), 1-12.

Lim, S., Lee, K., & Kang, J. (2018). Drug interaction extraction from the literature using a recursive neural network. PloS one, 13(1), e0190926.

Rodríguez, P., Bautista, M. A., Gonzalez, J., & Escalera, S. (2018). Beyond one-hot encoding: Lower dimensional target embedding. Image and Vision Computing, 75, 21-31.

Sahu, S. K., & Anand, A. (2018). Drug-drug interaction extraction from biomedical texts using long short-term memory network. Journal of biomedical informatics, 86, 15-24.

Xu, B., Shi, X., Zhao, Z., & Zheng, W. (2018). Leveraging biomedical resources in bi-lstm for drug-drug interaction extraction. IEEE Access, 6, 33432-33439.

Zhang, Y., Zheng, W., Lin, H., Wang, J., Yang, Z., & Dumontier, M. (2018). Drug–drug interaction extraction via hierarchical RNNs on sequence and shortest dependency paths. Bioinformatics, 34(5), 828-835.

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. IEEE Computational Intelligence magazine, 13(3), 55-75.

Bengio, Y., Goodfellow, I., & Courville, A. (2017). Deep learning (Vol. 1). Massachusetts, USA:: MIT press.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5, 135-146.

Dasigi, P., Ammar, W., Dyer, C., & Hovy, E. (2017). Ontology-aware token embeddings for prepositional phrase attachment. arXiv preprint arXiv:1705.02925.

Dozat, T., Qi, P., & Manning, C. D. (2017, August). Stanford's graph-based neural dependency parser at the conll 2017 shared task. In Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies (pp. 20-30).

Herbelot, A., & Baroni, M. (2017). High-risk learning: acquiring new word vectors from tiny data. arXiv preprint arXiv:1707.06556.

Kavuluru, R., Rios, A., & Tran, T. (2017, August). Extracting drug-drug interactions with word and character-level recurrent neural networks. In 2017 IEEE International Conference on Healthcare Informatics (ICHI) (pp. 5-12). IEEE.

Lucy, L., & Gauthier, J. (2017). Are distributional representations ready for the real world? Evaluating word vectors for grounded perceptual meaning. arXiv preprint arXiv:1705.11168.

Tai KS., Socher R., Manning CD. Improved semantic representations from tree-structured long shortterm memory networks; 2015. Preprint. Available from: arXiv:1503.00075. Cited 20 October 2017

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 5998-6008).

Wang, W., Yang, X., Yang, C., Guo, X., Zhang, X., & Wu, C. (2017). Dependency-based long short term memory network for drug-drug interaction extraction. Bmc Bioinformatics, 18(16), 578.

Zheng, W., Lin, H., Luo, L., Zhao, Z., Li, Z., Zhang, Y., ... & Wang, J. (2017). An attention-based effective neural model for drug-drug interactions extraction. BMC bioinformatics, 18(1), 445.

Zhang, L., Xiang, T., & Gong, S. (2017). Learning a deep embedding model for zero-shot learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2021-2030).

Zhang, Y., Zheng, W., Lin, H., Wang, J., Yang, Z., & Dumontier, M. (2017). Drug–drug interaction extraction via hierarchical RNNs on sequence and shortest dependency paths. Bioinformatics, 34(5), 828-835.

Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., & Bengio, Y. (2016, March). End-to-end attention-based large vocabulary speech recognition. In 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP) (pp. 4945-4949). IEEE.

Goldberg Y. A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research. 2016; 57:345–420

Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651.

Li, Q., Li, T., & Chang, B. (2016). Learning word sense embeddings from word sense definitions. In Natural Language Understanding and Intelligent Applications (pp. 224-235). Springer, Cham.

Liu, S., Tang, B., Chen, Q., & Wang, X. (2016). Drug-drug interaction extraction via convolutional neural networks. Computational and mathematical methods in medicine, 2016.

Kiperwasser, E., & Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. Transactions of the Association for Computational Linguistics, 4, 313-327.

Quan, C., Hua, L., Sun, X., & Bai, W. (2016). Multichannel convolutional neural network for biological relation extraction. BioMed research international, 2016

Santos, C. D., Tan, M., Xiang, B., & Zhou, B. (2016). Attentive pooling networks. arXiv preprint arXiv:1602.03609.

Wang, L., Cao, Z., De Melo, G., & Liu, Z. (2016, August). Relation classification via multi-level attention cnns. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1298-1307).

Zhao, Z., Yang, Z., Luo, L., Lin, H., & Wang, J. (2016). Drug interaction extraction from biomedical literature using syntax convolutional neural network. Bioinformatics, 32(22), 3444-3453.

Zheng, W., Lin, H., Zhao, Z., Xu, B., Zhang, Y., Yang, Z., & Wang, J. (2016). A graph kernel based on context vectors for extracting drug–drug interactions. Journal of biomedical informatics, 61, 34-43.

Abacha, A. B., Chowdhury, M. F. M., Karanasiou, A., Mrabet, Y., Lavelli, A., & Zweigenbaum, P. (2015). Text mining for pharmacovigilance: Using machine learning for drug name recognition and drug–drug interaction extraction and classification. Journal of biomedical informatics, 58, 122-132.

Kim, S., Liu, H., Yeganova, L., & Wilbur, W. J. (2015). Extracting drug–drug interactions from literature using a rich feature-based linear kernel approach. Journal of biomedical informatics, 55, 23-30.

Santos, C. N. D., Xiang, B., & Zhou, B. (2015). Classifying relations by ranking with convolutional neural networks. arXiv preprint arXiv:1504.06580.

Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., & Jin, Z. (2015). Classifying relations via long short term memory networks along shortest dependency paths. In proceedings of the 2015 conference on empirical methods in natural language processing (pp. 1785-1794).

Liu, Y., Wei, F., Li, S., Ji, H., Zhou, M., & Wang, H. (2015). A dependency-based neural network for relation classification. arXiv preprint arXiv:1507.04646.

Frank, C., Himmelstein, D. U., Woolhandler, S., Bor, D. H., Wolfe, S. M., Heymann, O., ... & Lasser, K. E. (2014). Era of faster FDA drug approval has also seen increased black-box warnings and market withdrawals. Health affairs, 33(8), 1453-1459.

Funk, C., Baumgartner, W., Garcia, B., Roeder, C., Bada, M., Cohen, K. B., ... & Verspoor, K. (2014). Large-scale biomedical concept recognition: an evaluation of current automatic annotators and their parameters. BMC bioinformatics, 15(1), 59.

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural Turing machines. arXiv preprint arXiv:1410.5401.

Hashimoto, K., Stenetorp, P., Miwa, M., & Tsuruoka, Y. (2014, October). Jointly learning word representations and composition functions using predicate-argument structures. In

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 1544-1555).

Hill, F., Cho, K., Jean, S., Devin, C., & Bengio, Y. (2014). Embedding word similarity with neural machine translation. arXiv preprint arXiv:1412.6448.

Kelly, L., Goeuriot, L., Suominen, H., Schreck, T., Leroy, G., Mowery, D. L., ... & Palotti, J. (2014, September). Overview of the share/clef ehealth evaluation lab 2014. In International Conference of the Cross-Language Evaluation Forum for European Languages (pp. 172-191). Springer, Cham.

Konstantinova, N. (2014, April). Review of relation extraction methods: What is new out there? In International Conference on Analysis of Images, Social Networks and Texts (pp. 15-28). Springer, Cham.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Lamurias, A., Ferreira, J. D., & Couto, F. M. (2014). Identifying interactions between chemical entities in biomedical text. Journal of integrative bioinformatics, 11(3), 1-16.

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

Segura-Bedmar, I., Martínez, P., & Herrero-Zazo, M. (2014). Lessons learnt from the DDIExtraction-2013 shared task. Journal of biomedical informatics, 51, 152-164.

Zeng, D., Liu, K., Lai, S., Zhou, G., & Zhao, J. (2014, August). Relation classification via convolutional deep neural network. In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers (pp. 2335-2344).

Chowdhury, M. F. M., & Lavelli, A. (2013, June). FBK-irst: A multi-phase kernel based approach for drug-drug interaction detection and classification that exploits linguistic information. In Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013) (pp. 351-355).

He, L., Yang, Z., Zhao, Z., Lin, H., & Li, Y. (2013). Extracting drug-drug interaction from the biomedical literature using a stacked generalization-based approach. PloS one, 8(6), e65814.

Herrero-Zazo, M., Segura-Bedmar, I., Martínez, P., & Declerck, T. (2013). The DDI corpus: An annotated corpus with pharmacological substances and drug–drug interactions. Journal of biomedical informatics, 46(5), 914-920.

Hill, D. P., Adams, N., Bada, M., Batchelor, C., Berardini, T. Z., Dietze, H., ... & Hastings, J. (2013). Dovetailing biology and chemistry: integrating the Gene Ontology with the ChEBI chemical ontology. BMC genomics, 14(1), 513.

Kong, X., Cao, B., & Yu, P. S. (2013, August). Multi-label classification by mining label and instance correlations from heterogeneous information networks. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 614-622). ACM.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

Nédellec, C., Bossy, R., Kim, J. D., Kim, J. J., Ohta, T., Pyysalo, S., & Zweigenbaum, P. (2013, August). Overview of BioNLP shared task 2013. In Proceedings of the BioNLP shared task 2013 workshop (pp. 1-7).

Pyysalo, S., Ginter, F., Moen, H., Salakoski, T., & Ananiadou, S. Distributional Semantics Resources for Biomedical Text Processing, 2013.

Segura Bedmar, I., Martínez, P., & Herrero Zazo, M. (2013). Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics.

Graves, A. (2012). Supervised sequence labelling. In Supervised sequence labelling with recurrent neural networks (pp. 5-13). Springer, Berlin, Heidelberg.

Fellbaum, C. (2012). WordNet. The encyclopedia of applied linguistics.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012, July). Semantic compositionality through recursive matrix-vector spaces. In Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning (pp. 1201-1211).

Björne, J., Airola, A., Pahikkala, T., & Salakoski, T. (2011). Drug-drug interaction extraction from biomedical texts with svm and rls classifiers. Proceedings of DDIExtraction-2011 challenge task, 35-42.

Chowdhury, M. F. M., & Lavelli, A. (2011). Drug-drug interaction extraction using composite kernels. Challenge Task on Drug-Drug Interaction Extraction, 27-33.

Chowdhury, M. F. M., Abacha, A. B., Lavelli, A., & Zweigenbaum, P. (2011). Two different machine learning techniques for drug-drug interaction extraction. Challenge task on drug-drug interaction extraction, 19-26.

Ciresan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2011, September). Convolutional neural network committees for handwritten character classification. In 2011 International Conference on Document Analysis and Recognition (pp. 1135-1139). IEEE.

Segura-Bedmar, I., Martinez, P., & de Pablo-Sánchez, C. (2011). Using a shallow linguistic kernel for drug–drug interaction extraction. Journal of biomedical informatics, 44(5), 789-804.

Thomas, P., Neves, M., Solt, I., Tikk, D., & Leser, U. (2011). Relation extraction for drug-drug interactions using ensemble learning. Training, 4(2,402), 21-425.

Aronson, A. R., & Lang, F. M. (2010). An overview of MetaMap: historical perspective and recent advances. Journal of the American Medical Informatics Association, 17(3), 229-236.

Chiticariu, L., Li, Y., Raghavan, S., & Reiss, F. R. (2010, June). Enterprise information extraction: recent developments and open challenges. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (pp. 1257-1258). ACM

Glorot, X., & Bengio, Y. (2010, March). Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256).

Haas, J. S., Iyer, A., Orav, E. J., Schiff, G. D., & Bates, D. W. (2010). Participation in an ambulatory e-pharmacovigilance system. Pharmacoepidemiology and drug safety, 19(9), 961-969.

Tanenblatt, M. A., Coden, A., & Sominsky, I. L. (2010, May). The ConceptMapper Approach to Named Entity Recognition. In LREC (pp. 546-51).

De Marneffe, M. C., & Manning, C. D. (2008). Stanford typed dependencies manual (pp. 338-345). Technical report, Stanford University. Wishart, D. S., Knox, C., Guo, A. C., Cheng, D., Shrivastava, S., Tzur, D., ... & Hassanali, M. (2008). DrugBank: a knowledgebase for drugs, drug actions and drug targets. Nucleic acids research, 36(suppl_1), D901-D906.

Chellapilla, K., Puri, S., & Simard, P. (2006, October). High performance convolutional neural networks for document processing.

Ciaramita, M., & Altun, Y. (2006, July). 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 (pp. 594-602). Association for Computational Linguistics.

Uzuner, O., Szolovits, P., & Kohane, I. (2006). i2b2 workshop on natural language processing challenges for clinical records. In Proceedings of the Fall Symposium of the American Medical Informatics Association.

Nebeker, J. R., Hoffman, J. M., Weir, C. R., Bennett, C. L., & Hurdle, J. F. (2005). High rates of adverse drug events in a highly computerized hospital. Archives of internal medicine, 165(10), 1111-1116.

Zhou, G., Su, J., Zhang, J., & Zhang, M. (2005, June). Exploring various knowledge in relation extraction. In Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05) (pp. 427-434).

Bodenreider, O. (2004). The unified medical language system (UMLS): integrating biomedical terminology. Nucleic acids research, 32(suppl_1), D267-D270.

Bengio Y, Ducharme R, Vincent P, Jauvin C. A neural probabilistic language model. J Mach Learn Res. 2003;3(Feb):1137–55.

Ciaramita, M., & Johnson, M. (2003). Supersense tagging of unknown nouns in WordNet. In Proceedings of the 2003 conference on Empirical methods in natural language processing (pp. 168-175).

Donaldson, M. S., Corrigan, J. M., & Kohn, L. T. (Eds.). (2000). To err is human: building a safer health system (Vol. 6). National Academies Press.

Scholkopf, B., Mika, S., Burges, C. J., Knirsch, P., Muller, K. R., Ratsch, G., & Smola, A. J. (1999). Input space versus feature space in kernel-based methods. IEEE transactions on neural networks, 10(5), 1000-1017.

Fellbaum, C. (1998). Towards a representation of idioms in WordNet. In Usage of WordNet in Natural Language Processing Systems.

Lazarou, J., Pomeranz, B. H., & Corey, P. N. (1998). Incidence of adverse drug reactions in hospitalized patients: a meta-analysis of prospective studies. Jama, 279(15), 1200-1205.

MUC-7. (1998). Proceedings of the 7th Message Understanding Conference (MUC-7). Morgan Kaufmann, San Mateo, CA.

Bates, D. W., Spell, N., Cullen, D. J., Burdick, E., Laird, N., Petersen, L. A., ... & Leape, L. L. (1997). The costs of adverse drug events in hospitalized patients. Jama, 277(4), 307-311.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

Twomey, J. M., & Smith, A. E. (1997). Validation and verification. Artificial neural networks for civil engineers: Fundamentals and applications, 44-64.

Johnson Jeffery and Booman Lyle. Drug-related morbidity and mortality. Journal of Managed Care Pharmacy, 2(1):39–47, 1996.

Williams, C. K., & Rasmussen, C. E. (1996). Gaussian processes for regression. In Advances in neural information processing systems (pp. 514-520).

Bates, D. W., Cullen, D. J., Laird, N., Petersen, L. A., Small, S. D., Servi, D., ... & Vander Vliet, M. (1995). Incidence of adverse drug events and potential adverse drug events: implications for prevention. Jama, 274(1), 29-34.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

Twomey, J. M., & Smith, A. E. (1995). Performance measures, consistency, and power for artificial neural network models. Mathematical and computer modelling, 21(1-2), 243-258.

Hinton, G. E., & Zemel, R. S. (1994). Autoencoders, minimum description length and Helmholtz free energy. In Advances in neural information processing systems (pp. 3-10).

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshop on Computational learning theory (pp. 144-152).

LeCun, Y. (1989). Generalization and network design strategies. Connectionism in perspective, 19, 143-155.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning representations by back-propagating errors. Cognitive modeling, 5(3), 1.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. nature, 323(6088), 533-536.

Rosenblatt, F. (1960). Perceptron simulation experiments. Proceedings of the IRE, 48(3), 301-309.

Appendix A

List of DDI Relation Extraction Models and SemEval Benchmark Metrics

Authors	**Year**	**Algorithm**	**Detection F_1**	**Classification F_1**
Lamurias, A., Sousa, D., Clarke, L. A., & Couto, F. M	2019	BiLSTM w/ Attention w/ SDP w/ ChEBI. Extends (Zhang Y et al, 2018)	.6129	.751
Lim, S., Lee, K., & Kang, J.	2018	Recursive Tree-LSTM w/ W2V	.838	.735
Sahu, S. K., & Anand, A. (2018).	2018	Dual BiLSTM w/ Attentive Pooling and Max Pool	-	.6939
Zhang, Y., Zheng, W., Lin, H., Wang, J., Yang, Z., & Dumontier, M	2018	Hierarchy BiLSTM w/ Attention w/ SDP	-	.729
Xu, B., Shi, X., Zhao, Z., & Zheng, W	2018	BiLSTM w/ Concept Matching MetaMap	-	.7115
Kavuluru, R., Rios, A., & Tran, T	2017	Char-BiLSTM and Word-BiLSTM		.7213
Zheng, W., Lin, H., Luo, L., Zhao, Z., Li, Z., Zhang, Y., ... & Wang, J.	2017	BiLSTM w/ Input Attention w/ Word Embedding and Position Embedding	.84	.773
Wang, W., Yang, X., Yang, C., Guo, X., Zhang, X., & Wu, C.	2017	Multichannel BiLSTM w/ DFT/BFT dependency parse	-	.72
Quan, C., Hua, L., Sun, X., & Bai, W.	2016	MultiChannel CNN	.79	.702
Zhao, Z., Yang, Z., Luo, L., Lin, H., & Wang, J	2016/7	CNN w/ Word Embedding	.772	.686
Liu, S., Tang, B., Chen, Q., & Wang, X.	2016 (2015 /12)	CNN+Word Embedding and Position Embedding	-	.6975
Zheng, W., Lin, H., Zhao, Z., Xu, B., Zhang, Y., Yang, Z., & Wang, J.	2016 (2015)	Context Vector Graph-Kernel	.818	.684

Kim, S., Liu, H., Yeganova, L., & Wilbur, W. J.	2015	SVM w/ Linear kernel	-	.67
Lamurias, A., Ferreira, J. D., & Couto, F. M	2014	2-stage Kernel-based SVM w/ Ensemble	.7457	.6402
Chowdhury, M. F. M., & Lavelli, A.	2013	Kernel-based SVM w/ 2-stages	**.8**	**.651**
He, L., Yang, Z., Zhao, Z., Lin, H., & Li, Y	2013	Feature-based and Kernel-based SVM	.6924	-
Björne, J., Airola, A., Pahikkala, T., & Salakoski, T.	2011	Feature-based SVM w/ syntactic features	.6299	
Chowdhury, M. F. M., & Lavelli, A.	2011	SVM w/ Shallow Linguistic Parse Kernel	.6370	
Chowdhury, M. F. M., Abacha, A. B., Lavelli, A., & Zweigenbaum, P	2011	Feature-based SVM	.6398	
Thomas, P., Neves, M., Solt, I., Tikk, D., & Leser, U	2011	Kernel-base SVM w/ multiple kernels	.6574	
Segura-Bedmar, I., Martinez, P., & de Pablo-Sánchez, C.	2011	SVM w/ Shallow Linguistic Parse Kernel	**.6001**	

Appendix B

Embedding Methods and Findings

This appendix provides a description of five different ontology embedding methods explored during this study. The embedding methods evaluated fall into three different categories of embeddings: Corpus-guided, Ontology-guided, and Task-guided.

The first category is *corpus-guided* embedding. This form of embedding is similar to other neural word embedding methods such as Mikolov et al (2013) and Pennington et al (2014) that rely on a corpus of data used to guide the unsupervised embedding algorithm. Instead of using a corpus of words, the corpus is processed using a concept matching method to extract the sequence of concepts over the words and phrases in the corpus. Concept matching over the corpus text produces a reduction in the amount of information by normalizing domain relevant concepts based on the ontology used by the concept matching algorithm. This approach has the added effect of denoising the data by eliminating words that do not match a concept and therefore is deemed irrelevant within the domain. This approach, however, does not leverage the semantic relations of the ontology and therefore its use was only evaluated as part of preliminary experiments and with the inclusion of additional Ontologies such as UMLS.

The second category of embeddings explored in this study is an *ontology-guided* embedding. This form of embedding relies solely on the ontology data to learn the concept embedding. This novel technique was explored as a method to encode the identifying attributes of each concept within its taxonomy (*is-a* hierarchy) and semantic relations to other concepts. This approach attempts to cluster similar concepts within a low-dimensional embedding space where each dimension within the embedding space represents semantic properties learned from the ontology.

The third category of embeddings described as *task-guided* does not rely on a corpus or an ontology, but instead trained as part of a classification model on a specific task. This approach can be considered the classical form of neural embeddings first proposed by Bengio et al (2003).

The following section provides a description of each embedding method evaluated in the study:

Classic Embedding (Dense Layer)

This embedding layer is based on the classic neural embedding by (Hill, Cho, Jean, Devin, & Bengio, 2014). This approach was evaluated first and serves as a baseline to compare the relative performance of other more sophisticated embedding methods. The approach utilizes a dense (fully connected) input layer to the model. The layer's input dimension V is equal to the number of concepts (i.e. vocabulary) within the ontology. This dense layer uses the Identity function as its activation function to propagate the incoming connection weights V into the D dimensional outgoing weights. Using this approach, pretraining the embedding layer is possible by saving the dense layers weights and restoring the weights within a target model.

This approach was evaluated on embedding task 1 and 2, defined in *Chapter 4 Results - Ontology Embedding Experiments* section, by first training the embedding layer within a shallow neural network model trained to detect DDI for a pair of candidate drug entities. *Table 4* shows the results of this experiment. This approach performed poorly relative to the other methods evaluated for both the task 1 and task 2. The F_1 metric for task 1 was .093 and .43 average mean error for task 2.

These results demonstrate that a conventional embedding approach is not sufficient and does not take advantage of the knowledge encoded within the ontology since it only relies on the normalized concepts and learns the embedding based on the task definition. This further demonstrates that it is not possible to learn the semantics axioms expressed by the ontology through specific task such as task 1 (DDI detection).

Concept Identity Vector Encoding

This approach uses the ontology to produce a k-hot vector representation for the concept's *identity vector*. The vector features represent the characteristics of the concept within the ontology. These characteristics include the ancestor concepts within the taxonomy as well as other concepts it relates to. This idea of an identity vector is leveraged by other embedding methods that follow. In this method, the identify vector is used as the input to a shallow classification model that is trained on task 1 and evaluated on both task 1 and task 2. The classifier layer used Softmax with Cross-Entropy loss function as the

output layer and the identify vector for each of the two drug entities were concatenated and passed to the classifier layer of the model. Additional variations of the model were explored such as adding a fully connected dense layer before the output layer. Concatenation of the identity vector before and after the dense layer was evaluated. This approach demonstrated slightly better performance when compared to the Classical Embedding method on task 1, with an F_1 metric of .135 and performed significantly better on task 2, with an average mean error of .330.

Figure *37* shows an example of an identity vector for a concept within the ChEBI ontology. Each vector feature (component) represents an ontological attribute for the concept. The identify vector is computed by traversing the ontology for each concept and setting the feature position flag for each of the taxonomic ancestors as well as the related concepts. For this experiment, only the matching concepts from the training dataset were used to compute the identity vector. Later experiments build upon this idea of a concept identity vector.

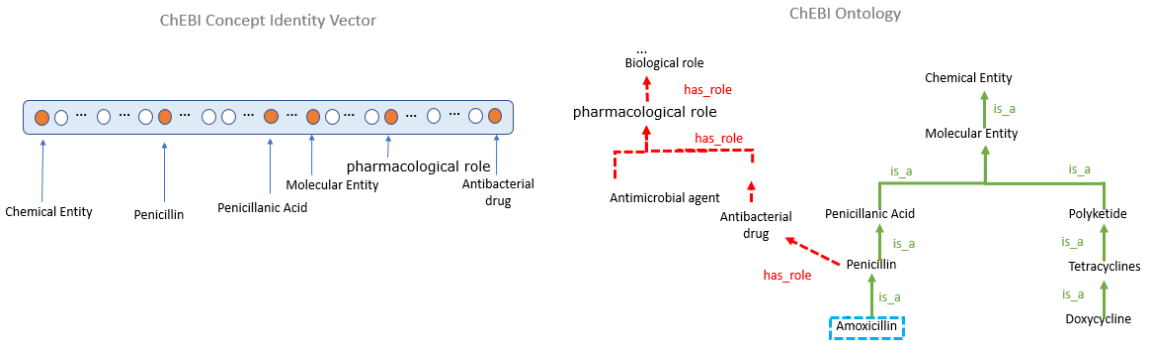

Figure 37 - Concept Identity Vector

Autoencoding of Ontological Identity Vector

This method explores the use of an unsupervised Autoencoder model trained using the concept identity vectors generated from the previous method (*Concept Identity Vector Encoding*). The goal of this task-guided method is to produce a compact representation of the k-hot identity vector by reducing the dimensionality and increasing the density of the vector positions within the encoded embedding space. Figure 38 shows an illustration of the Autoencoder model architecture. An autoencoder (Hinton & Zemel, 1994) employs

several hidden dense layers called the *encoder* that decrease the dimensionality of the input vector until it reaches the target decoder dimensionality. The *decoder* layers then increase the dimensionality until it reaches the original input vector dimensional width. This is at times referred to in literature as a funnel architecture. The funnel serves as a compression of information that can be trained to efficiently represent features within a reduced dimensional feature space. The autoencoder model is trained using an unsupervised input reconstruction task that attempts to produce an output vector that matches the input vector. This model used the Mean Squared Error (MSE) loss function and a Logistic-Sigmoid activation function to bound output range of 0 and 1.

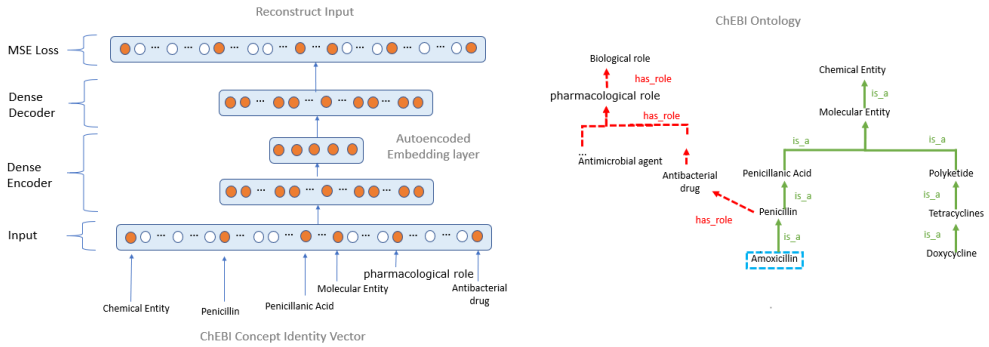

Figure 38 - Autoencoder for Ontological Identity Vector

Several experiments were conducted with different number of hidden dense layers that reduced the original input vector width to different target dimensionality. The best performing model configuration was selected based on its performance in reconstructing the input feature. The selected model was then evaluated using task 1 and task 2 for comparison to other embedding methods. The results demonstrated relatively poor performance in task 1 and average performance for task 2.

```
graph.addLayer("DENSE1", new DenseLayer.Builder().nIn(chebiIdIndexer.getIndexSize()).nOut(2500).activation(Activation.LEA
graph.addLayer("DENSE2", new DenseLayer.Builder().nIn(2500).nOut(1000).activation(Activation.LEAKYRELU).build(), "DENSE1"
graph.addLayer("DENSE3", new DenseLayer.Builder().nIn(1000).nOut(500).activation(Activation.LEAKYRELU).build(), "DENSE2");
graph.addLayer("DENSE4", new DenseLayer.Builder().nIn(500).nOut(250).activation(Activation.LEAKYRELU).build(), "DENSE3");
graph.addLayer("DENSE5", new DenseLayer.Builder().nIn(250).nOut(100).activation(Activation.LEAKYRELU).build(), "DENSE4");
graph.addLayer("DENSE6", new DenseLayer.Builder().nIn(100).nOut(250).activation(Activation.LEAKYRELU).build(), "DENSE5");
graph.addLayer("DENSE7", new DenseLayer.Builder().nIn(250).nOut(500).activation(Activation.LEAKYRELU).build(), "DENSE6");
graph.addLayer("DENSE8", new DenseLayer.Builder().nIn(500).nOut(1000).activation(Activation.LEAKYRELU).build(), "DENSE7");
graph.addLayer("DENSE9", new DenseLayer.Builder().nIn(1000).nOut(2500).activation(Activation.LEAKYRELU).build(), "DENSE8"

graph.addLayer(
        "OUTPUT", new OutputLayer.Builder().activation(Activation.SIGMOID)
                .lossFunction(LossFunctions.LossFunction.MSE).nIn(2500).nOut(chebiIdIndexer.getIndexSize()).build(),
        "DENSE9");// .nIn(10)
```

Ontologically Learned Identity Vectors (OLIV)

This method uses a novel unsupervised ontology-guided training approach to produce a concept embedding. First a pre-processing step traverses the ontology graph visiting each concept node within the graph. For each concept node, the ancestor and related concepts are gathered. The method utilizes a max traversal distance from the current visited concept. The intuition for limiting the distance is based on the observations that all concepts eventually converge on a small number of common ancestors that do not provide any distinguishing clues about the concepts. For example, *Chemical Entity* concept is the root for most all concepts in ChEBI and does not provide any informative value. Instead, the focus is placed on the concepts most proximal to the target concept. Using a recursive graph traversal and dynamic programming algorithm that tracks distance and caches partial path results, the pre-processing step can quickly compute related concepts for each concept visited in the ontology. For this experiment, the entire ChEBI ontology (114,000 concepts) was processed.

Next, a shallow neural network model is designed with the goal of predicting for each given concept the related concepts that were preprocessed. Using an unsupervised training approach to fit the model in such a way that for each one-hot vector representation of a given concept the model predicts the likelihood of related concepts in the output. The model is trained using SGD with back-propagation. A single dense hidden layer is used with the identity function for its activation representing the embedding weights. Figure 39 shows the model architecture. Several experiments were conducted to evaluate the optimal embedding dimension width based on the related-concept task used to train the embedding. Table 4 shows that this method demonstrated the best performance for task 1 and task 2 with an F_1 of .35 and .200 average mean error. This approach was selected as the embedding method used for the remaining experiments in this study.

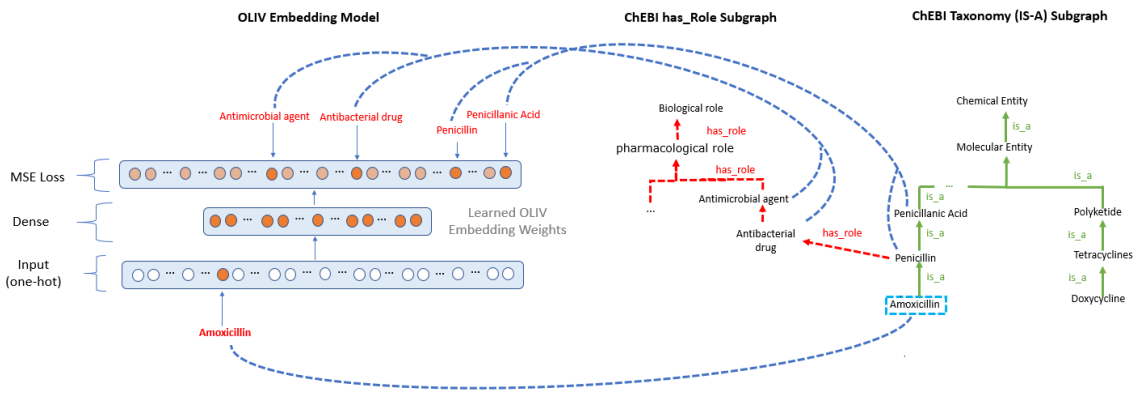

Figure 39 - OLIV Embedding Model

Ontological Path-encoded Vectors (OPEV)

This method uses a heuristic algorithm to construct a path-encoding vector representation for each concept. Each vector position represents a distance from the root concept. The vector values are normalized using min-max scaling. Figure 40 shows an example of the encoding for the Amoxicillin concept. During preliminary experimentation with this approach using the ChEBI ontology, several issues were encountered with the method. These issues included: multiple inheritance by some concepts, disjoint ontology graph where not all paths lead to a root concept, and cycles within some paths where a concept node can occur more than one tine providing multiple paths to the root node. Lastly, this approach requires one vector representation for each relation type within the ontology. For the ChEBI ontology, this is manageable since it only has 9 semantic relations in addition to the *is-a* taxonomic relation. However, other ontologies such as UMLS have hundreds of semantic relations requiring hundreds of path-encoded vectors per relation. During experimentation and metric gathering using ChEBI, it was found that the average number of relations that a concept participates in is 3. This means that most of the concept's vectors representing these relations will have zero values. This sparsity in values is known to have a negative impact to neural network activation and ability for the model to learn. As such, this approach was abandoned, and no further experimentation was performed.

Another drawback of using the OPEV method for concept embedding involves the additional complexity introduced to the final DDI model. The approach requires an additional input channel for each ontology relation with the input dimensionality of the

concept vector. The concept embedding layers would then require a merging strategy resulting in a large increase in the total number of trainable parameters required by the model. This increase in model capacity can lead to overfitting issues as well as a significant increase in training time.

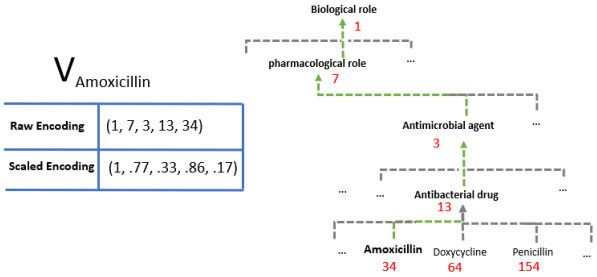

Figure 40 - OPEV Encoding

Ontology Embedding Tests

The following section provides an overview and description of the test methodology and algorithm used for comparatively evaluating the embedding methods. Two different tests were defined to assess the performance of each method. The results of each test were collected and analyzed to determine the best performing Ontology Embedding method used for the remainder of the study.

Task 1: DDI Detection Test

The first test is defined as *Task1: DDI Detection Test*. This test uses a subset of the SemEval 2013 DDI Benchmark dataset to evaluate the F_1 accuracy metric using only the embedding as feature inputs to a shallow neural network featuring a single Dense layer. This test evaluates the performance of the embedded drug entities towards the DDI relation extraction task. The F_1 metric is collected for each embedding method evaluated using this test.

Task 2: Ontological Distance Preservation Test

The second test is defined as *Task2: Ontological Distance Preservation Test*. This test measures how well the embedding layer can represent relative distances between the ontological space (pre-embedding) and the embedding space. This test measures the

average of the accumulated error (MSE) using two different distance functions compared to the ontological path distance between a pair of concepts. The first distance used is the Cosine Distance between a pair of concepts within the embedding space. The second is the Euclidean distance between a pair of concepts within the embedding space.

The test selects 100 random drug concepts mentioned in the DDI Training dataset. Each drug concept is paired with the other 99 concepts and three distances are computed: Ontological distance, Cosine distance, and Euclidean distance. The list of distances is sorted to produce a ranked list for each respective distance type (i.e. 3 sorted lists of concepts ranked by distance). The MSE is computed by using the ranked Ontological distance as a reference to compare to the corresponding ranked Cosine distance and ranked Euclidean distance. This error is accumulated and averaged over the total number of pair permutations produced from the pool of 100 randomly sampled concepts.

This test is repeated using the same pool of sampled concepts for each embedding method producing an average error based on the Cosine distance and Euclidean distance errors.

Appendix C

Experiment Results

The experiment results for the experiments conducted as part of this study have been compiled into an Excel Spreadsheet format. The Java DDI framework, Python models, and experiment results are available at: http://mjlorenzo.com/research/ddi/

Each sheet within the experiment spreadsheet includes the metrics with a description of each trial run along with model configuration and hyperparameters used. The table is organized by experiment numbers. Experiments are re-run multiple times to ensure reproducibility and to establish baseline averages to compare against. Blank or 'N/A' values imply that the parameter represented by the column does not apply to that experiment.

Appendix D

Source Code and Materials Availability

The source code developed for this study along with the preprocessed data for both Python and Java are available at: http://mjlorenzo.com/research/ddi/

The Python project includes the implementation of the two baseline control models used during the experiments. This includes the Zhang Y. (2018) and the Lamurias & Couto (2019) DDI Classification Models based on Keras 2.3.1 and Theano 1.0.5 computational backend. The project includes the enhanced version of these baseline models with the parameter option to run either the baseline or the augmented model which features the ChEBI pre-trained embedding layer presented in this study.

The Java project includes a port of the Zhang Y and the Lamurias & Couto models based on the DeepLearning4J and ND4J computational backend. Additionally, several variations of DDI Classification models, including different input channels, are included. The code structure is divided into model code, that represents the Neural Network Computation Graph along with its hyperparameters; Vectorizers, that transform the intermediate feature input format into a vectorized tensor for use with the corresponding model; Extractors, used to convert the DDI dataset into intermediate representation of the features; SemEval data model for parsing the train and test dataset; and utilities, for parsing and traversing the ChEBI and UMLS ontology, Concept Mapper implementation for concept matching, and NLP utility for parsing sentences using Stanford NLP Parser.

Additionally, the OLIV trained ChEBI concept embedding weights are available for download at http://mjlorenzo.com/research/ddi/chebi_300.bin along with the binary dictionary representation of ChEBI ontology for use with Concept Mapper at http://mjlorenzo.com/research/ddi/chEBI.dic.