Deep Learning for Information Extraction in the Biomedical Domain

by

Víctor Suárez-Paniagua

A dissertation submitted by in partial fulfillment of the requirements for the degree of Doctor of Philosophy in

Computer Science and Technology

Universidad Carlos III de Madrid

Advisor:

Isabel Segura-Bedmar

March 2019

Published and submitted content

The research publications realized by the author are partially included for the systems proposed in the present document and are listed as follows:

1. Journals:

   - Víctor Suárez-Paniagua, Isabel Segura-Bedmar (2018). Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction, Proceedings of the 11th International Workshop on Data and Text Mining in Biomedical Informatics (DTM-BIO 2017), June, 2018, BMC Bioinformatics, ISSN: 1471-2105, 19, 8, pp: 209, [Q1] partially included in Chapter 6.

     `https://doi.org/10.1186/s12859-018-2195-1`.

   - Víctor Suárez-Paniagua, Isabel Segura-Bedmar, Paloma Martínez, (2017). Exploring Convolutional Neural Networks for Drug-Drug Interaction Extraction, Database The Journal of Biological Databases and Curation, January, 2017, ISSN: 1758-0463, Volume 2017, 1, [Q1] partially included in Chapter 6.

     `https://doi.org/10.1093/database/bax019`.

2. Conferences:

   - Víctor Suárez-Paniagua, Isabel Segura-Bedmar, Akiko Aizawa, (2018). UC3M-NII Team at SemEval-2018 Task 7: Semantic Relation Classification in Scientific Papers via Convolutional Neural Network, Proceedings of The 12th International Workshop on Semantic Evaluation (SemEval), New Orleans, Louisiana, USA, June, 2018, Association for Computational Linguistics, pp: 793-797, partially included in Chapter 6.

     `http://www.aclweb.org/anthology/S18-1126`.

   - Víctor Suárez-Paniagua, Isabel Segura-Bedmar, Paloma Martínez, (2018). LABDA at TASS-2018 Task 3: Convolutional Neural Networks for Relation Classification in Spanish eHealth documents, Proceedings of TASS 2018: Workshop on Semantic Analysis at SEPLN, TASS@SEPLN 2018, co-located with 34nd SEPLN Conference (SEPLN 2018), Sevilla, Spain, September, 2018, ISSN: 1613-0073, TASS 2018: Workshop on Semantic Analysis at SEPLN, pp: 71-76, partially included in Chapter 7.

`http://ceur-ws.org/Vol-2172/p7-labda_tass2018.pdf`.

- Víctor Suárez-Paniagua, Isabel Segura-Bedmar, Paloma Martínez, (2017). LABDA at SemEval-2017 Task 10: Relation Classification between keyphrases via Convolutional Neural Network, Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval), Vancouver, Canada, August, 2017, Association for Computational Linguistics, pp: 938-941, partially included in Chapter 6.

  `https://www.aclweb.org/anthology/S17-2169`.

- Víctor Suárez-Paniagua, Isabel Segura-Bedmar, (2016). Using Recursive Neural Networks to detect and classify drug-drug interactions from biomedical texts, 22nd European Conference on Artificial Intelligence (ECAI 2016), The Hague, Holland, August, 2016, partially included in Chapter 5.

  `http://ebooks.iospress.com/volumearticle/44972`.

- Isabel Segura-Bedmar, Víctor Suárez-Paniagua, Paloma Martínez, (2015). Exploring Word Embedding for Drug Name Recognition, Sixth Workshop on Health Text Mining and Information Analysis, Lisboa, Portugal, September, 2015, partially included in Chapter 4.

  `https://aclweb.org/anthology/W/W15/W15-2608.pdf`.

- Víctor Suárez-Paniagua, Isabel Segura-Bedmar, Paloma Martínez, (2015). Word Embedding Clustering for Disease Named Entity Recognition, Proceedings of the Fifth BioCreative Challenge Evaluation Workshop (BIOCREATIVE V), Sevilla, Spain, September, 2015, partially included in Chapter 4.

  `http://www.biocreative.org/media/store/files/2015/BCV2015_paper_47.pdf`.

- Isabel Segura-Bedmar, Víctor Suárez-Paniagua, Paloma Martínez, (2015). Combining Conditional Random Fields and Word Embeddings for the CHEMDNER-patents task, Proceedings of the Fifth BioCreative Challenge Evaluation Workshop (BIOCREATIVE V), Sevilla, September, 2015, partially included in Chapter 4.

  `http://www.biocreative.org/media/store/files/2015/BCV2015_paper_51.pdf`.

Abstract

The main hypothesis of this PhD dissertation is that novel Deep Learning algorithms can outperform classical Machine Learning methods for the task of Information Extraction in the Biomedical Domain. Contrary to classical systems, Deep Learning models can learn the representation of the data automatically without an expert domain knowledge and avoid the tedious and time-consuming task of defining relevant features.

A Drug-Drug Interaction (DDI), which is an essential subset of Adverse Drug Reaction (ADR), represents the alterations in the effects of drugs that were taken simultaneously. The early recognition of interacting drugs is a vital process that prevents serious health problems that can cause death in the worst cases. Health-care professionals and researchers in this domain find the task of discovering information about these incidents very challenging due to the vast number of pharmacovigilance documents. For this reason, several shared tasks and datasets have been developed in order to solve this issue with automated annotation systems with the capability to extract this information. In the present document, the DDI corpus, which is an annotated dataset of DDIs, is used with Deep Learning architectures without any external information for the tasks of Name Entity Recognition and Relation Extraction in order to validate the hypothesis. Furthermore, some other datasets are tested to evidence the performance of these systems.

To sum up, the results suggest that the most common Deep Learning methods like Convolutional Neural Networks and Recurrent Neural Networks overcome the traditional algorithms concluding that Deep Learning is a real alternative for a specific and complex scenario like the Information Extraction in the Biomedical domain. As a final goal, a complete architecture that covers the two tasks is developed to structure the named entities and their relationships from raw pharmacological texts.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Artificial Intelligence (AI) has grown to perform human tasks automatically. In this effort, scientists had focused on building Machine Learning architectures in order to emulate human behaviour. The vast of these systems need tons of manually annotated data in order to learn the common patterns and extract information from these labelled datasets. Thus, the machines can finally predict a possible solution for new data through statistical analysis.

Previously, the systems used for Machine Learning were trained with predefined features by the researches that could contain the relevant information for each sample (feature engineering). This step is a vital process for the performance of the systems because an inadequate representation of the data may carry wrong predictions. For that reason, this domain requires expert knowledge in order to define the hand-crafted features carefully. This task is a very tedious and time-consuming task. Deep Learning started from the idea of automatically transforming the raw data into a feature representation (feature learning) (Goodfellow et al., 2016). To this end, Neural Networks (NNs) are used in Deep Learning for capturing the information from the data and trying to process it as the human brain does. Furthermore, multiple layers of NNs are applied in deep architectures to create a more abstractive representation of the inputs. This process is very natural in the human understanding where a problem is decomposed into sub-problems with multiple levels of representation in order to create a complete knowledge of it (Bengio, 2009).

Natural Language Processing (NLP) is the field of study that concerns with the interaction between a computer and human languages. The primary goal of NLP is to analyse and synthesize natural language data in order to communicate with machines and perform the following tasks among others: Natural language understanding, Natural language generation, Speech recognition, Text-to-speech, Information Extraction, Information retrieval, Machine translation, Question answering and Spoken dialogue system. The understanding and usage of human language is a complex process due to the high level of ambiguity and interpretability of communication. For that reason, creating a machine that computes natural language perfectly is an AI-complete task or strong AI (Yampolskiy, 2013), i.e. making computers as smart as humans (Shapiro, 1992). Thus, interpreting human communication by computers is a very complex topic because of its ambiguity and constant changes. Concretely, the goal of text mining is to understand the natural language from a collection of written documents as input data. Mainly, the topic of Information Extraction (IE) is to transforms unstructured data into structured data in order to extract the internal information from texts. Two basics processes are proposed for this end, Named Entity Recognition (NER), which gets the more relevant words of the sentences that are called entities, and Relation Extraction (RE), which defines the possible relationships between the entities. The source data for these tasks are usually taken from general purpose documents, such as letters, newspapers or reviews. However, one of the most valuable knowledge of humanity is medical knowledge, and it would be beneficial in using these techniques to interpret this information. The complete understanding of this kind of texts is difficult because they present very rare or unknown words and complex structures. Therefore, applying these tasks to biomedical texts is very challenging for the NLP community.

Nowadays there is a growing concern about Adverse Drug Reactions (ADRs) since they are a severe risk to patient safety (Bond and Raehl, 2006) as well as a cause of rising health care costs (van Der Hooft et al., 2006). A Drug-Drug Interaction (DDI), a type of Adverse Drug Reaction (ADR), occurs when a drug is co-administrate with another affecting their levels of activity. Unfortunately, most DDIs are not detected during clinical trials, mainly because these trials are designed to assess the effectiveness of drugs rather than their safety (Stricker and Psaty, 2004). The Institute Of Medicine (IOM) reported in 2009 that each year at least 1.5 million ADRs occur and over 4 billion dollars are spent in order to treat their preventable (Council and of Medicine, 2009). Concretely, ADRs cause more than 300,000 deaths in the USA and Europe per year (Lazarou et al., 1998). According to (Makary and Daniel, 2016), medical error has become the third most common cause of death in the USA being medication error the most significant medical error because of the adverse drug effects.

Ideally, prescribing information about a drug should list its potential interactions, together with the following information about each interaction: its mechanism, its relation to the doses of both drugs, its time course, the factors that alter an individual's susceptibility to it, its seriousness and severity, and the probability of its occurrence (Aronson, 2004; Ferner and Aronson, 2006). In practice, however, this information is rarely available (Aronson, 2007).

An enormous amount of the most current and valuable information is unstructured, written in natural language and hidden in published articles, scientific journals, books and technical reports. Drug interactions are bread and butter to journals of clinical pharmacology due to the vast number of interactions that can happen (Aronson, 2007). Concretely, the number of articles published in MedLine about the biomedical domain is increasing between 10,000 and 20,000 documents per week (Medline, 2007). Additionally, 300,000 articles are published each year within the pharmacology domain (Duda et al., 2005). For this reason, there are no drug-drug interactions databases up-to-date because biomedical scientists discover new drugs and publish new studies of their effects continuously. Thus, the management of DDIs is a critical issue due to the overwhelming amount of information available on them (Hansten, 2003). Consequently, health-care professionals have to spend a long time reviewing DDI databases as well as the pharmacovigilance literature in order to check if there is a new ADR.

The introduction of new technologies in primary care and hospitals has led to the development of electronic medical record systems, which has opened the opportunity of incorporating decision support systems to prevent DDIs and inform on possible actions to take. However, the deployment of these systems is not widespread yet (Rodríguez-Terol et al., 2009), and most systems in primary care do not support the management of DDIs. Therefore, clinicians and pharmacists must be able to manage by themselves the richness of information available on DDIs.

IE can be of great benefit in the pharmaceutical industry allowing identification and extraction of relevant information and providing an exciting way of reducing the time spent by health care professionals on reviewing the literature. Therefore, the development of an automated annotation system of DDIs which extracts the information of pharmacovigilance documents is vital for improving and updating the drug knowledge databases.

The DDIExtraction Task (Segura-Bedmar et al., 2013, 2011) was an NLP challenge to promote the development of techniques applied to the biomedical domain, in particular, to the pharmacovigilance subject and providing a common framework for the evaluation of the participating systems and other researchers interested in this topic. Besides, the significant contribution of DDIExtraction Task has been to provide the DDI corpus (Herrero-Zazo et al., 2013) as a benchmark for developing systems. This dataset is a valuable annotated corpus that provides gold standard data for training and evaluating supervised machine learning algorithms to extract DDIs form texts. It contains abstracts about DDIs from MedLine and comments from the DrugBank database (Wishart et al., 2006). The task is divided into two main subtasks the detection and classification of drugs and their relations in the dataset. The results obtained in the shared task showed that supervised machine learning techniques (Segura-Bedmar et al., 2014), such as Support Vector Machines (Cortes and Vapnik, 1995) with hand-crafted features and kernel-based methods are the techniques most used for the extraction and recognition of the drug names and the detection and classification of interactions between drugs. Most of the participating systems use large and rich sets of linguistic features, which have to be defined by domain experts and text miners, and which require considerable time and effort.

Deep learning methods can be an exciting alternative to the classical supervised machine-learning algorithms since they can automatically extract the most appropriate features from representing a problem for a given task (Goodfellow et al., 2016). Thus, the hypothesis is that Deep Learning architectures can outperform classical Machine Learning approaches without using a large predefined set of features for IE in the particular case of biomedical documents.

1.2 Objectives

The main objectives of this thesis are:

- Exploring state-of-the-art Deep Learning architectures with word embeddings, such as Recursive Neural Network (RecNN) (Socher, Pennington, Huang, Ng and Manning, 2011), Convolutional Neural Network (CNN) (Lecun et al., 1998) and Recurrent Neural Network (RNN) (Rumelhart et al., 1986).
- Proposing methods based on Deep Learning for NER and RE tasks in the biomedical domain.
- Studying their performance in the problem of extracting information of texts that contain interactions between drugs using the DDI Corpus.
- Building a complete system that involves the NER and RE tasks.
- Testing the language and domain independence of the models using other datasets.
- Identifying open issues from the conclusions in order to propose future researches.

1.3 Contributions

The present document explores novel neural models for the tasks of NER and RE in biomedical texts. Concretely, the main contributions of this thesis are:

- A complete description and review of the state-of-the-art systems for NER and RE with a special focus on the biomedical texts for DDI interactions.

- The exploration of the word embeddings and their clusters as features using a CRF classifier for recognizing mentions of drugs, chemical compounds, and diseases in biomedical documents.

- The use of the MV-RNN, which is the first Deep Learning system for RE, applied to the extraction of drug interactions.

- The proposition of the first CNN with attention pooling for the DDIExtraction Task.

- Applying the CNN architecture to deal similar task such as SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications and at SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers in order to check their efficiency.

- The definition of the first end-to-end model using the DDI Corpus, which involves the NER and RE from raw data.

- Testing the language and domain independence of the end-to-end model using another language dataset like the Semantic Analysis at SEPLN (TASS-2018) (Martínez-Cámara et al., 2018).

The study of these systems is a baseline for future works and could be implemented as a medical resource because of their excellent performance. This thesis demonstrates that Deep Learning architectures are a real alternative to classical Machine Learning methods since that they can learn the representation of the texts. Furthermore, these models can be used in other domains because no background knowledge of the topic is required to capture the relevant information of the texts.

1.4 Outline

The rest of this thesis is organized as follows:

- Chapter 2 introduces the theoretical background of the NLP, the different techniques for word representation, the most used Machine Learning algorithms for text mining, and the description of the main Deep Learning architectures and its precedents.
- Chapter 3 depicts an overview of the Information Extraction tasks in the biomedical domain including a detailed description of the dataset used for the proposed systems. Also, the section summarizes the state-of-the-art models for Named Entity Recognition and Relation Extraction subtasks with an emphasis on Deep Learning systems.
- Chapter 4 shows a CRF architecture using word embeddings for the case of DrugNER Task. Also, two other tasks focused on the disease and chemical mentions recognition are presented applying the same model.
- Chapter 5 presents the adaptation of the Matrix-Vector Recursive Neural Network for the extraction of drug-drug interactions.
- Chapter 6 shows an exploration of the Convolutional Neural Network architecture in order to set the best hyper-parameters for the DDIExtaction Task. Moreover, the section describes the performance in the extraction of drug interactions using the Convolutional Neural Network with different pooling operation such as maximum, average and attention mechanism. Furthermore, the architecture is also applied to solve two Shared Tasks at SemEval, SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications (Augenstein et al., 2017) and at SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers in order to check their efficiency (Gábor et al., 2018).
- Chapter 7 depicts a complete system that involves the Named Entity Recognition and Relation Extraction subtasks in biomedical texts from raw data as an end-to-end architecture. This model performs the tasks for English and Spanish documents, the DDI Corpus and the Semantic Analysis at SEPLN (TASS-2018) (Martínez-Cámara et al., 2018), in order to test its language independence capability.
- Chapter 8 draws some conclusions of the studies and proposes some future works to be

taken into consideration in order to improve the results that can be applied to real scenario systems.

1.5 Funding

This thesis has been supported by:

- Pre-doctoral research training scholarship of the Carlos III University of Madrid (PIF UC3M 02-1415) for four years.
- Research Program of the Ministry of Economy and Competitiveness - Government of Spain, (DeepEMR project TIN2017-87548-C2-1-R).
- Research Program of the Ministry of Economy and Competitiveness - Government of Spain, (eGovernAbility-Access project TIN2014-52665-C2-2-R).
- Doctoral stay TEAM - Technologies for information and communication, Europe - east Asia Mobilities project (Erasmus Mundus Action 2-Strand 2 Programme) funded by the European Commission realized in the University of Tokyo, Japan, for the Aizawa Laboratory in National Institute of Informatics (NII) for seven months.

Chapter 2

Background

Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed (Samuel, 1959). The main algorithms of machine learning are classified into four classes. Unsupervised learning generates knowledge from samples without a category assigned. Supervised learning learns from samples with labels that represent a category. Semi-supervised learning solves tasks that only a few samples are labelled taking knowledge from non-categorized samples. In Reinforcement learning, an agent takes actions depending on the actual state and a reward given from the environment.

Supervised learning models are subdivided in regression and classification techniques depending on what they predict, a real number or a set of predefined categories, respectively. Inside the classification systems, there is a distinction between models that learn the joint probability distribution of the samples and their labels (generative classifiers) and models that learn a function that separates the categories according to some weights (discriminative classifiers).

A classification model used in supervised learning takes a set of input variables $\mathbf{x}$ called observations or features and gives a set of output variables $\mathbf{y}$ called classes or labels. Formally, the goal of the supervised learning algorithms is to find a mapping function that predicts the correct outputs y of a set of inputs $\mathbf{x}$ as

$$y = f(\mathbf{x}) \tag{2.1}$$

From a practical point of view, the datasets to be used in supervised models contain M instances as pairs of information $(\mathbf{x}, y)$ where $\mathbf{x} \in \mathbb{R}^N$ is a vector of N attributes that define each sample and y represents the class of the sample.

2.1 Artificial Neural Networks

The basic architecture of the Artificial Neural Networks (ANN) is the single artificial neuron which is biologically inspired by the operations of these cells in the brain (McCulloch and Pitts, 1943). These units are simple pattern detectors that learn synaptic weights of each attribute to generate a prediction. Figure 2.1 represents the weighted function that an artificial neuron computes as

$$y = f(\sum_{i=1}^{N} w_i x_i + b) = f(\mathbf{w}^T \mathbf{x} + b) \tag{2.2}$$

where f is an activation function, $\mathbf{w} \in \mathbb{R}^N$ is a vector of the weights for the N attributes and b is an offset term called bias.

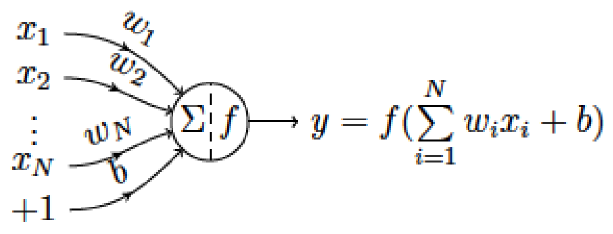
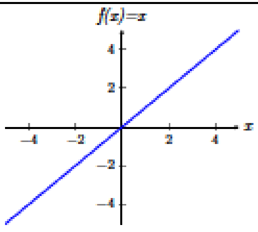
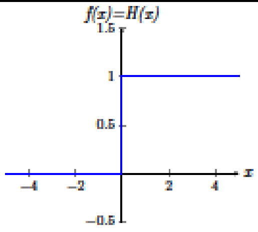
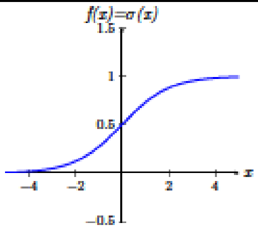
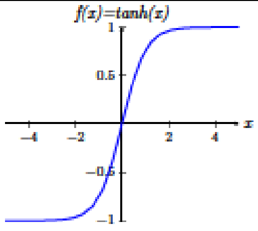
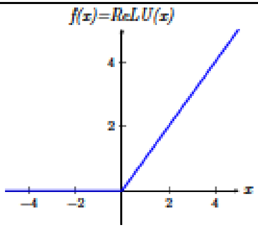


Figure 2.1: The architecture of an artificial neuron.

2.1.1 Activation Functions

The activation function defines the threshold where a neuron is working to generate the prediction in its output. Concretely, this function indicates how much information to transfer from a neuron in a range of activation. The most common activation functions are showed in Table 2.1. The majority of them are non-linear functions which are a desirable capability for computing inherently non-linear inputs.

Table 2.1: Most commonly used activation functions for the artificial neuron computation.

Activation Function	Equation	Derivatite	Plot
Identity	$f(x) = x$	$f'(x) = 1$	
Heaviside step	$H(x) = \begin{cases} 1, & x > 0 \\ 0, & otherwise \end{cases}$	$H'(x) = 0, x \neq 0$	
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	$\sigma'(x) = \sigma(x)(1 - \sigma(x))$	
Softmax	$Softmax(x) = \frac{e^{x^i}}{\sum_{i=1}^{K} e^{x^i}}$	$Softmax'(x) = Softmax(x)(1 - Softmax(x))$	Multi-class Sigmoid
Tanh	$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$tanh'(x) = 1 - tanh(x)^2$	
ReLu	$ReLU(x) = \begin{cases} x, & x > 0 \\ 0, & otherwise \end{cases}$	$ReLU'(x) = \begin{cases} 1, & x > 0 \\ 0, & otherwise \end{cases}$	

2.1.2 Multilayer Perceptron

The simplest activation function to create a linear separation of binary categories is the Heaviside step function. This function implemented with an artificial Neuron is known like Perceptron (Rosenblatt, 1958). The architecture can compute the different boolean functions as a weighted combination of two binary inputs $(\mathbf{A}, \mathbf{B})$ as $y = H(\mathbf{A}w_1 + \mathbf{B}w_2 + b)$ where b is the bias term and w_1 and w_2 are the weights of A and B, respectively (see Figure 2.2).

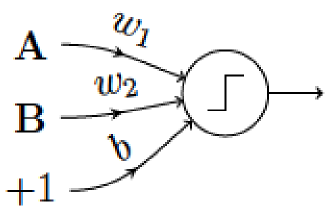


Figure 2.2: A Perceptron which represents the different boolean functions with two binary inputs and a bias term.

It is not possible for this architecture to perform a function for the **XOR** boolean function with the Perceptron due to the fact that two boundaries are required in order to classify the points. However, **XOR** is defined by the combination of other boolean functions. Thus, the solution to perform this boolean operation is to build a concatenation of two functions with the Perceptron and to compute their outputs with another Perceptron. This network was the first implementation of a Multilayer Perceptron (MLP) whose main idea is to connect neuron outputs to other neurons like the human brain does to generate more complex functions.

The general architecture for this model with an arbitrary activation function and an intermediate layer of neurons is known as Neural Network with a hidden layer. The hidden layer performs a computation with the weighted values of the inputs in each neuron. Then, a fully connected layer combines the outputs of all the neurons to a final neuron to give a prediction. Figure 2.3 represents the feed-forward computation as:

$$\begin{aligned} \mathbf{a} &= f(\mathbf{W}^T\mathbf{x} + \mathbf{b}) \\ y &= f(\mathbf{u}^T\mathbf{a} + b_u) \end{aligned} \tag{2.3}$$

where $\mathbf{W} \in \mathbb{R}^{N \times H}$ is the matrix of weights for the inputs and the H neurons, $\mathbf{b} \in \mathbb{R}^{H}$ is the offset of the hidden layer, $u \in \mathbb{R}^{H}$ is the vector of weights for the neuron output and b_u is the offset of the final neuron.

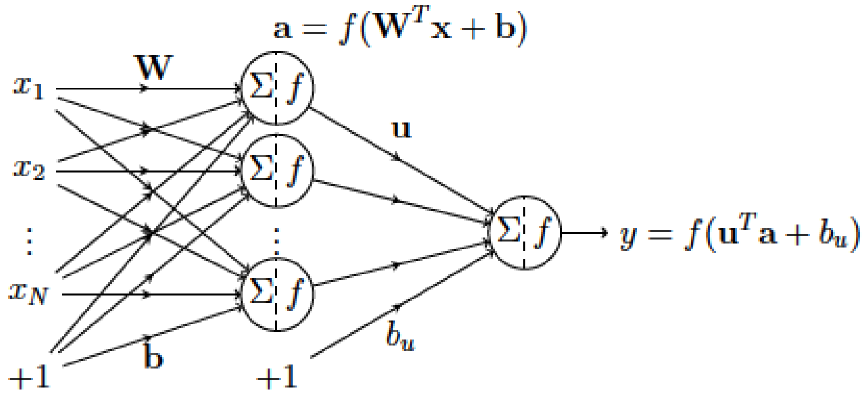

$\mathbf{a} = f(\mathbf{W}^T\mathbf{x} + \mathbf{b})$
x_1
x_2
x_N
$+1$
$\mathbf{W}$
$\mathbf{b}$
Σ
f
$+1$
$\mathbf{u}$
b_u
$y = f(\mathbf{u}^T\mathbf{a} + b_u)$

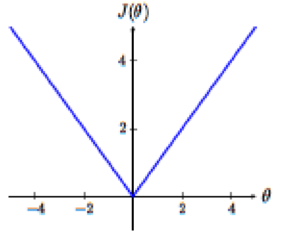
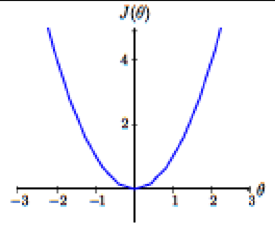
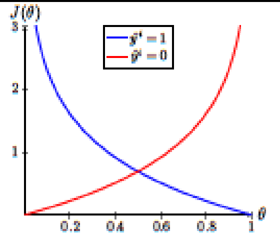
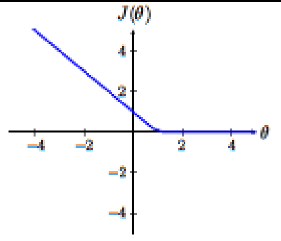
Figure 2.3: A concatenation of Perceptrons for building a Multilayer Perceptron with a hidden layer.

The universal approximation theorem states that a Neural Network using a single hidden layer with a finite number of neurons with arbitrary activation function and a linear neuron in the output layer can approximate any continuous function using the appropriate parameters (Cybenko, 1989; Hornik, 1991). However, the Neural Networks require a loss function, a search optimization algorithm and a model validation in order to measure the error made during the training, find the optimal values of the weights, and fit the optimal number of neurons, respectively.

2.1.3 Loss Functions

Loss functions, cost functions or objective functions $J(\theta)$ are used in the optimization problems to estimate the parameter θ of the model that generate the least losses. Concretely, it measures the error between the prediction of the model y and the expected value $\hat{y}$ in order to fit the parameters in the training phase. Table 2.2 shows the common loss functions to solve supervised optimization problems.

Table 2.2: Most commonly used loss functions for supervised machine learning.

Loss Function	Equation	Derivative	Plot
Mean Absolute Error	$J(\theta) = \frac{1}{M}\sum_{i=1}^{M} \lvert\hat{y}_i - y_i\rvert$	$J'(\theta) = \begin{cases} 1, & \hat{y} - y > 0 \\ -1, & \hat{y} - y < 0 \end{cases}$	
Mean Squared Error	$J(\theta) = \frac{1}{2M}\sum_{i=1}^{M} \lvert\lvert\hat{y}_i - y_i\rvert\rvert^2$	$J'(\theta) = \hat{y} - y$	
Cross-entropy	$J(\theta) = \frac{-1}{M}\sum_{i=1}^{M} \hat{y}_i ln(y_i) + (1 - \hat{y}_i) ln(1 - y_i)$	$J'(\theta) = \hat{y} - y$	
Negative Log Likelihood	$J(\theta) = \frac{-1}{M}\sum_{i=1}^{M} \hat{y}_i ln(y_i)$	$J'(\theta) = \hat{y} - y$	Multi-class Cross-entropy
Hinge	$J(\theta) = \frac{-1}{M}\sum_{i=1}^{M} max(0, 1 - \hat{y}_i y_i)$	$J'(\theta) = \begin{cases} -\hat{y}_i x_i, & \hat{y}_i y_i < 1 \\ 0, & otherwise \end{cases}$	

2.1.4 Learning algorithm

The back-propagation (Bryson et al., 1963) is the learning algorithm that updates the weights of the ANN models according to the error from the loss function. Contrary to the forward computation to generate the prediction, this method propagates the error in a backward direction. Thus, the parameters $\boldsymbol{\theta}$ of the model will be updated with an optimization algorithm called Gradient Descent (GD) that assures the convergence to the local minimum of the loss function as:

$$\theta^{NEW} \leftarrow \theta^{OLD} - \alpha \frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \tag{2.4}$$

This method reduces the error with a step called learning rate decay α in the direction given by the gradient. For this end, derivatives of the loss function with respect to the weights need to be calculated for applying the learning process. Each time of training in this stage is called an epoch, and the weights are updated in order to decrease the loss function. Besides, it is a requirement to back-propagate that activation and loss functions are differentiable. An example of minimizing a function with a GD is showed in Figure 2.4 where a first initial point is corrected each step (3 times or epochs in the example) to reach the minimum value of the function.

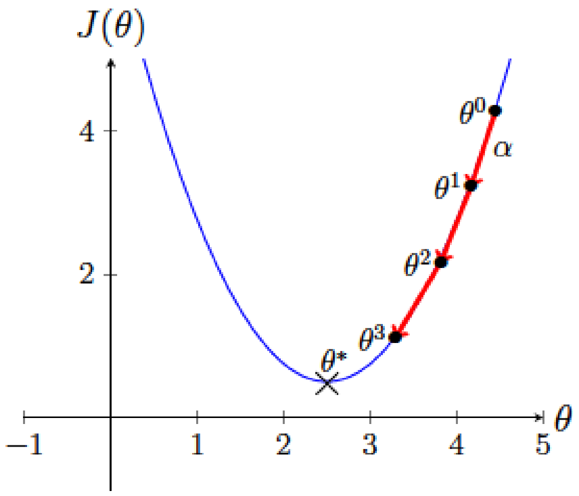


Figure 2.4: 3 steps of a Gradient Descent with learning decay rate α starting from the initial point θ^0 where θ^* represents the local minimum.

Figure 2.5 shows the derivative of each parameter in a MLP where $s = \mathbf{u}^T\mathbf{a} + b_u$ and $\mathbf{z} = \mathbf{W}^T\mathbf{x} + \mathbf{b}$. The activation function outputs are firstly taken from the feed-forward computation using the Equation 2.3, then the gradient for each parameter is calculated and finally the error is propagated from the derivative of the loss function. Thus, the derivative of a parameter can be seen as three parts multiplication, the forwarded signal, the local gradient and the back-propagated error.

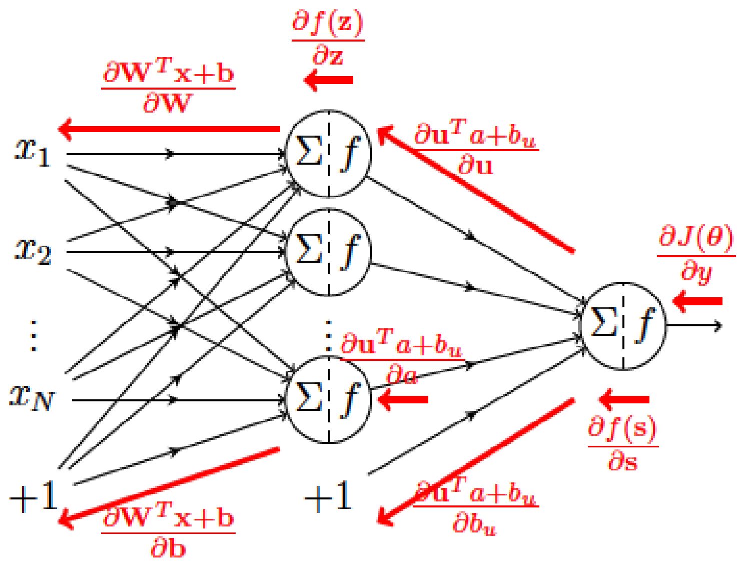

$\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}}$
$\frac{\partial \mathbf{W}^T\mathbf{x}+\mathbf{b}}{\partial \mathbf{W}}$
x_1
x_2
$\vdots$
x_N
$+1$
$\Sigma \mid f$
$\frac{\partial \mathbf{u}^T a+b_u}{\partial \mathbf{u}}$
$\frac{\partial J(\boldsymbol{\theta})}{\partial y}$
$\frac{\partial \mathbf{u}^T a+b_u}{\partial a}$
$\frac{\partial f(\mathbf{s})}{\partial \mathbf{s}}$
$+1$
$\frac{\partial \mathbf{u}^T a+b_u}{\partial b_u}$
$\frac{\partial \mathbf{W}^T\mathbf{x}+\mathbf{b}}{\partial \mathbf{b}}$

Figure 2.5: The derivatives of each parameter of a MLP for the back-propagation algorithm.

The back-propagation algorithm needs the derivatives of the parameters that depends on previous layer derivatives. For this reason, the chain rule can propagate the error in a backward manner starting from the loss function derivative until the weights of the first layer. Following the architecture of Figure 2.5, the first derivative to be calculated is the loss function output as:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial y} = J'(\boldsymbol{\theta}) \tag{2.5}$$

The output of the activation function takes the derivative of the loss function and computes the local gradient given by:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial s} = \left[\frac{\partial J(\boldsymbol{\theta})}{\partial y}\right]\frac{\partial f(s)}{\partial s} = [J'(\boldsymbol{\theta})] \times f'(s) \tag{2.6}$$

Then, the previous derivative is propagated to the output layer weights, bias and the output of the hidden layer for calculating the derivatives and updating their values with the GD as follows:

$$\begin{aligned}\frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{u}} &= \left[\frac{\partial J(\boldsymbol{\theta})}{\partial s}\right]\frac{\partial \mathbf{u}^T a + b_u}{\partial \mathbf{u}} = [J'(\boldsymbol{\theta}) \times f'(s)] \times a \\ \frac{\partial J(\boldsymbol{\theta})}{\partial b_u} &= \left[\frac{\partial J(\boldsymbol{\theta})}{\partial s}\right]\frac{\partial \mathbf{u}^T a + b_u}{\partial b_u} = [J'(\boldsymbol{\theta}) \times f'(s)] \times 1 \\ \frac{\partial J(\boldsymbol{\theta})}{\partial a} &= \left[\frac{\partial J(\boldsymbol{\theta})}{\partial s}\right]\frac{\partial \mathbf{u}^T a + b_u}{\partial a} = [J'(\boldsymbol{\theta}) \times f'(s)] \times \mathbf{u}\end{aligned} \tag{2.7}$$

Similarly, the derivative of the output in the hidden layer is taken to propagate the error to the activation function of the hidden layer as:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{z}} = \left[\frac{\partial J(\boldsymbol{\theta})}{\partial a}\right]\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} = [J'(\boldsymbol{\theta}) \times f'(s) \times \mathbf{u}] \times f'(\mathbf{z}) \tag{2.8}$$

Finally, this derivative is used in the hidden layer weights and bias to obtain their partial derivatives and update their values with the learning algorithm with the following equations:

$$\begin{aligned}\frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{W}} &= \left[\frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{z}}\right]\frac{\partial \mathbf{W}^T\mathbf{x} + \mathbf{b}}{\partial \mathbf{W}} = [J'(\boldsymbol{\theta}) \times f'(s) \times \mathbf{u} \times f'(\mathbf{z})] \times \mathbf{x}^T \\ \frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{b}} &= \left[\frac{\partial J(\boldsymbol{\theta})}{\partial \mathbf{z}}\right]\frac{\partial \mathbf{W}^T\mathbf{x} + \mathbf{b}}{\partial \mathbf{b}} = [J'(\boldsymbol{\theta}) \times f'(s) \times \mathbf{u} \times f'(\mathbf{z})] \times 1\end{aligned} \tag{2.9}$$

2.1.5 Optimization

The GD has three variants depending on the number of data used in each step:

- Batch Gradient Descent updates the parameters of the network $\boldsymbol{\theta}$ using the entire dataset.
- Stochastic Gradient Descent computes the derivative of the loss function with respect to the parameters with each sample $(\mathbf{x}_i, y_i)$.

- Mini-Batch Gradient Descent is an intermediate solution which takes m samples of the dataset to compute the learning algorithm of GD each time.

The choice of the learning step is critical for the training because small α makes the GD very slow because many steps are needed to reach the minimum, at the contrary, large α makes the GD jump over the local minimum and the minimization gets stuck. More importantly, the learning algorithm could reach a suboptimal local minimum or non converge to an optimal solution due to a highly non-convex loss function. To solve these issues some improvements to the GD are proposed:

- Momentum (Qian, 1999): accelerates GD to the gradient direction in order to gain faster convergence and prevent the oscillations around local optima. The Equation 2.10 adds a momentum parameter γ that takes a percentage of the previous momentum term v.

$$\begin{aligned} v^{NEW} &\leftarrow \gamma v^{OLD} - \alpha \frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \\ \theta^{NEW} &\leftarrow \theta^{OLD} - v^{NEW} \end{aligned} \tag{2.10}$$

- Nesterov accelerated gradient (NAG) (Nesterov, 1983): anticipates where the parameters are going to be in the momentum term calculating the gradients with respect to the next position $\theta^{OLD} - \gamma v^{OLD}$ (see Equation 2.11). Thus, it can speed up the GD by an adaptation according to the slope position.

$$\begin{aligned} v^{NEW} &\leftarrow \gamma v^{OLD} - \alpha \frac{\partial J(\boldsymbol{\theta})}{\partial (\theta^{OLD} - \gamma v^{OLD})} \\ \theta^{NEW} &\leftarrow \theta^{OLD} - v^{NEW} \end{aligned} \tag{2.11}$$

- Adaptive Gradient (Adagrad) (Duchi et al., 2011): updates the learning rate size automatically depending on the commonness of the data. It makes larger updates if the features are infrequent and smaller updates if the features are more frequent. In Equation 2.12, $\mathbf{G}$ represents a diagonal matrix with the sum of the squares of the gradient of each parameter $\sum_{\tau}^{t}\left(\frac{\partial J(\theta_\tau)}{\partial \theta_\tau^{OLD}}\right)^2$ and ϵ is a small number that avoids the division by zero.

$$\begin{aligned} \alpha^{NEW} &\leftarrow \frac{\alpha^{OLD}}{\sqrt{\mathbf{G}}+\epsilon} \\ \theta^{NEW} &\leftarrow \theta^{OLD} - \alpha^{NEW}\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \end{aligned} \tag{2.12}$$

- Root Mean Square Prop (RMSProp): reduces the uniformly decreasing of the learning rate in Adagrad with the average of all past squared gradients s multiplied by a parameter γ in the same way that the momentum (see Equation 2.13).

$$\begin{aligned} s^{NEW} &\leftarrow \gamma s^{OLD} + (1-\gamma)\left(\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}}\right)^2 \\ \alpha^{NEW} &\leftarrow \frac{\alpha^{OLD}}{\sqrt{s^{NEW}}+\epsilon} \\ \theta^{NEW} &\leftarrow \theta^{OLD} - \alpha^{NEW}\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \end{aligned} \tag{2.13}$$

- Adaptive Learning Rate (Adadelta) (Zeiler, 2012): proposes to add the r term to the RMSProp that multiplies the average of all past updates rule with the learning rate as the Equation 2.14.

$$\begin{aligned} s^{NEW} &\leftarrow \gamma s^{OLD} + (1-\gamma)\left(\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}}\right)^2 \\ \alpha^{NEW} &\leftarrow \alpha^{OLD}\frac{\sqrt{r^{OLD}}+\epsilon}{\sqrt{s^{NEW}}+\epsilon} \\ r^{NEW} &\leftarrow \gamma r^{OLD} + (1-\gamma)\left(\alpha^{NEW}\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}}\right)^2 \\ \theta^{NEW} &\leftarrow \theta^{OLD} - \alpha^{NEW}\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \end{aligned} \tag{2.14}$$

- Adaptive Moment Estimation (Adam) (Kingma and Ba, 2014): keeps the average of all past gradient v and squared gradients s estimated with the mean and the variance, respectively. In Equation 2.15, γ_1 and γ_2 represents the percentage of decaying average of past gradients and squared gradients and bias corrected for v and s, respectively.

$$\begin{aligned}
v^{NEW} &= \gamma_1 v^{OLD} + (1-\gamma_1)\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}} \\
s^{NEW} &= \gamma_2 s^{OLD} + (1-\gamma_2)\left(\frac{\partial J(\boldsymbol{\theta})}{\partial \theta^{OLD}}\right)^2 \\
\hat{v}^{NEW} &= \frac{v^{NEW}}{1-\gamma_1} \\
\hat{s}^{NEW} &= \frac{s^{NEW}}{1-\gamma_2} \\
\theta^{NEW} &\leftarrow \theta^{OLD} - \alpha\frac{\hat{v}^{NEW}}{\sqrt{\hat{s}^{NEW}}+\epsilon}
\end{aligned} \tag{2.15}$$

2.1.6 Weight Initialization

Neural networks need to initialize the learning algorithm from an initial value for each weight. Choosing the starting point is crucial for the convergence of the gradient descent for reaching a local minimum in the error function. One possible consideration is starting with all the weights at zero. However, this makes the derivatives and the weights updates be the same in all the neurons of the Neural Network. Thus, all the hidden unit are symmetric, and the performance of the Neural Network would be the same as a single neuron and no better than a linear model. Should be noted that there is no problem setting biases to zero because the values in the neurons will be different.

For breaking this symmetry, the best solution is to initialize the weights randomly very close to zero, and thereby each neuron can compute different functions. The most common choice is to use a probability distribution with zero mean, such as uniform or normal distribution. In addition, two initialization methods can adjust the variance of th according to the selected non-linear activation function.

- Xavier initialization (Glorot and Bengio, 2010), defined as $\sqrt{\frac{1}{s}}$, is recommended for tanh or sigmoid functions.
- He initialization (He et al., 2015), defined as $\sqrt{\frac{2}{s}}$, is recommended for ReLU functions.

For both cases, s is the size of the previous layer. This initialization is the main source of randomness of Neural Networks. For this reason, it is highly recommended in practice the use of random seeds in order to obtain the same results.

2.1.7 Validation

The definition of a stopping rule is vital to determinate when a Neural Network has to stop training. This stopping point must assure the generalization of the model without fitting the training data perfectly in order to avoid capturing the noisy information. To this end, it is very convenience to define a validation set that tests the machine each epoch. In the cases where this set is not predefined, the original training set can be split to form the new training set and the validation set which must be as representative as the training set.

The early stopping criteria is a rule that finds the optimal number of epochs for training a Neural Network. In each epoch, the validation set tests the performance of the trained model. Thus, the minimum value of error in the validation set gives the optimal number of epochs for training the network. Before this point, the model is considered not totally trained (under-fitting), and after this point, the model cannot generalize well (over-fitting).

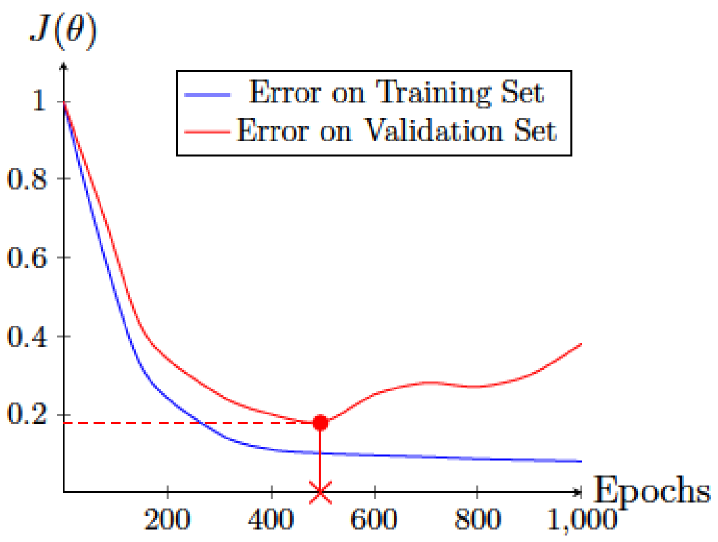


Figure 2.6: The training stage of a Neural Network where the loss function is decreasing in each epochs. The optimal number of epochs (red cross) is located by the minimum value in the validation set (red point).

2.1.8 Regularization

Regularization is a mathematical technique applied to machine learning models to prevent over-fitting in the training phase. The most common of these techniques is using a regularization term in the objective function for making the learning process more flexible and avoiding capturing the irrelevant data. Concretely, we add a tuning parameter λ that multiplies and constraints the values of the weights in order to penalize the complexity of the model and make smoother functions. Two different regularization term can be applied:

- L1-norm, which use the absolute mean of the weights as $J(\theta) + \lambda \sum |w|$, also known as LASSO estimator.
- L2-norm, which use the squared mean of the weights as $J(\theta) + \lambda \sum w^2$, also known as Ridge regression.

The main differences are that L1 regularization produces a sparse output making some weights to zero if they are not relevant in contrast, L2 regularization has analytical solutions and is computationally more efficient.

Another regularization method specially for Neural Network is Dropout (Hinton et al., 2012). This technique prevents the co-adaptations of neurons on the training stage by randomly setting the neurons to zero with a probability p following a Bernoulli distribution. In the test stage, p is zero for using the whole layer of neurons.

2.2 Classical Information Extraction classifiers

The most common machine learning methods for IE tasks with labelled exampled in a finite number of classes are the classification models with supervised learning. Figure 2.7 illustrates some other classical machine learning systems used for these tasks presented as following:

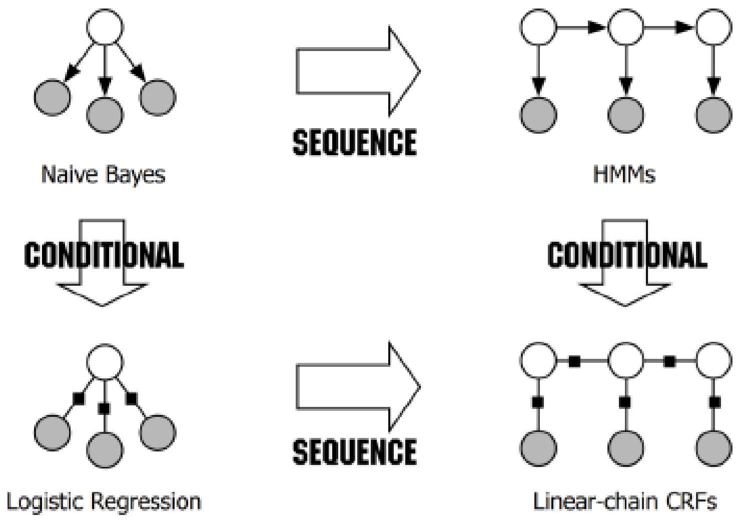


Figure 2.7: Diagram of traditional supervised classification system which shows the relationships between them. Image extracted from (Sutton et al., 2012).

2.2.1 Naive Bayes

Naive Bayes is a generative model that describes the joint distribution $p(y, \mathbf{x})$ by the prior probability $p(y)$ and the likelihood function $p(\mathbf{x}|y)$ as Equation 2.16. This model predicts a class label $\hat{y}$ given a vector of independent features $\mathbf{x}$ using the maximum a posteriori rule $\hat{y} = \text{argmax}_{k\in\{1,\ldots,K\}}\, p(y, \mathbf{x})$.

$$p(y, \mathbf{x}) = p(y) \prod_{m=1}^{M} p(x_m|y) \tag{2.16}$$

2.2.2 Hidden Markov Models

Hidden Markov Models (HMM) (Baum et al., 1970) is the sequential version of the Naive Bayes that describes the join distribution $p(\mathbf{y}, \mathbf{X})$ as Equation 2.17. Each time, the model predicts a class label $\hat{y}_n$ given a vector of independent features $\mathbf{x}_n$ of the current input and the output of the previous step y_{n-1} like a Markov Chain using the maximum a posteriori rule $\hat{y_n} = \text{argmax}_{k\in\{1,\ldots,K\}}\, p(y, \mathbf{x}_n)$.

$$p(\mathbf{y}, \mathbf{X}) = \prod_{n=1}^{N} p(y_n|y_{n-1}) p(\mathbf{x}_n|y_n) \tag{2.17}$$

2.2.3 Logistic Regression

Logistic Regression is a discriminative model that describes the conditional probability of $p(y|\mathbf{x})$ as Equation 2.18. This model uses the sigmoid function with the weights λ_m of features f_m defined with respect to y and $\mathbf{x}$ to generate a class prediction. The features are defined for state-observation pairs $f_m(y, \mathbf{x})$.

$$p(y|\mathbf{x}) = \frac{exp\left(\sum_{m=1}^{M} \lambda_m f_m(y, \mathbf{x})\right)}{\sum_{y'} exp\left(\sum_{m=1}^{M} \lambda_m f_m(y', \mathbf{x})\right)} \tag{2.18}$$

2.2.4 Conditional Random Fields

Conditional Random Fields (CRF) (Lafferty et al., 2001) is the sequential version of the Logistic Regression that describes the conditional probability $p(\mathbf{y}|\mathbf{X})$ as Equation 2.19. This model uses the sigmoid function with the weights λ_m of features f_m defined with respect to y_n, y_{n-1} and $\mathbf{x}_n$ to generate a class prediction like a Markov Chain. It is needed to defined at least one feature for each state transition $f_m(y_n, y_{n-1})$ and one for each state-observation pair $f_m(y_n, \mathbf{x}_n)$.

$$p(\mathbf{y}|\mathbf{X}) = \frac{exp\left(\sum_{m=1}^{M} \lambda_m f_m(y_n, y_{n-1}, \mathbf{x}_n)\right)}{\sum_{y'} exp\left(\sum_{m=1}^{M} \lambda_m f_m(y'_n, y'_{n-1}, \mathbf{x}_n)\right)} \tag{2.19}$$

2.2.5 Support Vector Machines

Support Vector Machines (SVM) (Cortes and Vapnik, 1995) is a supervised machine learning algorithm for data classification. It defines a hyper-plane $\mathbf{wx} + b = 0$ that separates two classes which obtains the largest distance or maximum margin between the nearest data points to the other class (support vectors). Figure 2.8 shows data points with two variables $\mathbf{x} = [x_1, x_2]$ of two classes $y = -1, 1$ where the maximum margin is defined by the margins $\mathbf{wx} + b \leq 1$ and $\mathbf{wx} + b \geq -1$ which are the boundaries of the classes $y_i = 1$ and $y_i = -1$, respectively.

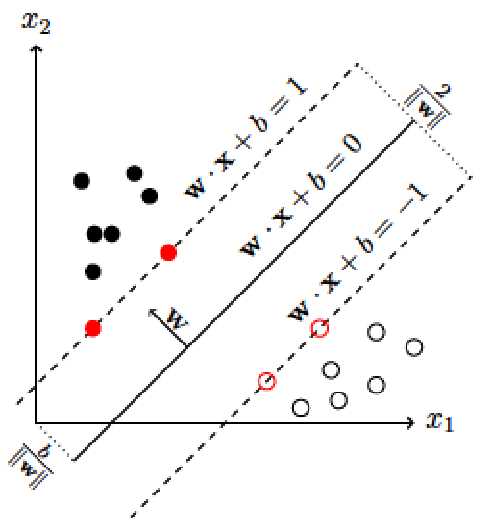

x_2
x_1
$\mathbf{w}\cdot\mathbf{x}+b=1$
$\mathbf{w}\cdot\mathbf{x}+b=0$
$\mathbf{w}\cdot\mathbf{x}+b=-1$
$\mathbf{w}$
$\frac{2}{\|\mathbf{w}\|}$
$\frac{b}{\|\mathbf{w}\|}$

Figure 2.8: An SVM separating two classes by an hyperplane $\mathbf{wx} + b = 0$.

The parameter $\frac{b}{||\mathbf{w}||}$ represents the offset of the hyperplane and the margin distance is $\frac{2}{||\mathbf{w}||}$. In order to maximize this distance that represents the margin, it is necessary to minimize $||\mathbf{w}||$. Thus, the maximum margin can be rewritten to get an optimization problem as the Equation 2.20, called Hard-margin.

$$\begin{aligned} \text{minimize } & \frac{1}{2}||\mathbf{w}||^2 \\ \text{subject to } & y_i(\mathbf{wx}_i + b) \leq 1 \end{aligned} \tag{2.20}$$

Considering cases where the data cannot be linearly separable, aggregating a slack variable ξ_i to the margins allows some deviation in order to classify the non-separable samples. Equation 2.21 represents the Soft-margin minimization function where C is the penalty parameter that balances the increasing of the margin size and the wrong side data. One possible choice for the slack variable is the Hinge Loss, $\xi_i = \max(0, 1 - y_i(\mathbf{wx}_i + b))$, which is zero if the data is not in the correct boundary.

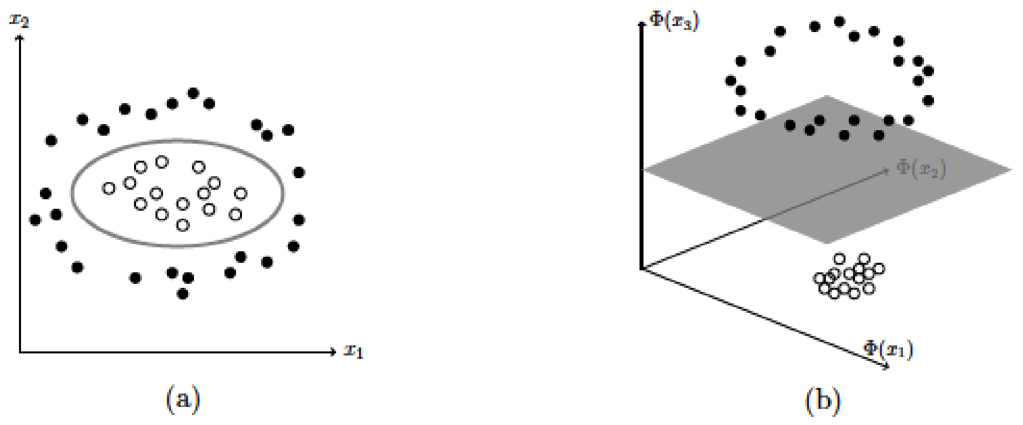


Figure 2.9: Kernel trick applied to a non-linear decision problem.

$$\begin{aligned} &\text{minimize } \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}\xi_i \\ &\text{subject to } y_i(\mathbf{w}\mathbf{x}_i + b) \leq 1 - \xi_i \\ &\xi_i \geq 0;\ C \geq 0 \end{aligned} \tag{2.21}$$

SVM cannot build a non-linear decision boundaries like Figure 2.9a. However, the kernel trick (Aizerman et al., 1964) maps the data $\mathbf{x}$ into a higher dimensional space with a kernel function defined as $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j)$ to create a linear separating hyperplane in this space (see Figure 2.9b). There are four basic kernels defines as follows:

- Linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$
- Polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\mathbf{x}_i^T\mathbf{x}_j + r)^d$
- Gaussian radial basis function (RBF): $K(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma||\mathbf{x}_i - \mathbf{x}_j||^2}$ for $\gamma > 0$
- Hyperbolic tangent or sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = tanh(\gamma\mathbf{x}_i^T\mathbf{x}_j + r)$ for $\gamma > 0$ and $r < 0$

2.3 Deep Learning

Deep Learning is a set of techniques for machine learning that uses Neural Networks architectures (Goodfellow et al., 2016). Currently, they provide the best performance to many data processing tasks (LeCun et al., 2015). The original idea was to create MLPs connected with more than one hidden layer in order to generate higher abstraction levels with deeper architectures (see Figure 2.10) (Bengio, 2009). Similarly, to the human brain, the information is processed by multiples transformations and learned higher and complex representations of the input in each hidden layer.

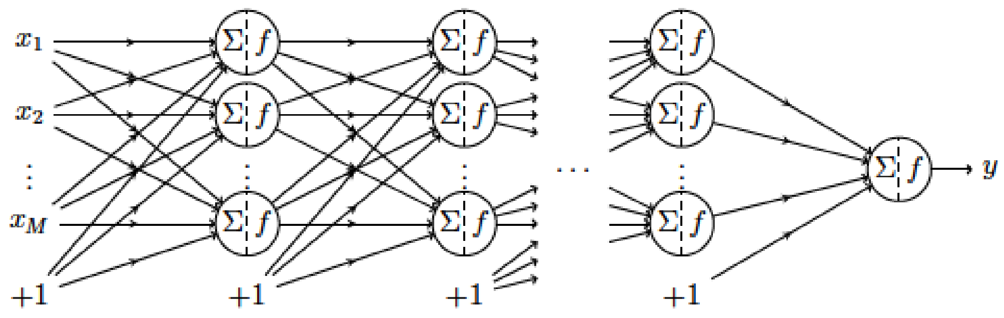


Figure 2.10: A Deep Neural Network with multiple layers.

Historically, the focus of data processing has moved from manually rule-based systems to automated systems. Figure 2.11 shows that classical machine learning approaches need hand-designed features while the representation learning models extract the representation of these features automatically. Currently, Deep Learning algorithms create complex knowledge from simplex feature representations using more layers that can compute functions with different levels of abstraction.

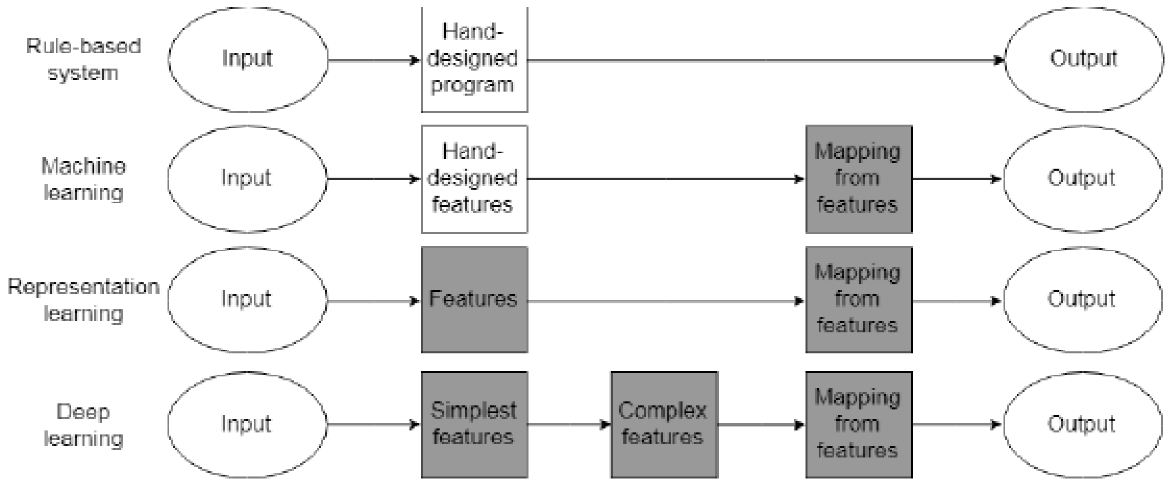


Figure 2.11: Historical view of the data processing.

The main advantages of Deep Learning are that the universal approximation theorem also applies to take the advantage that the functions can be approximated using fewer neurons because the network computes a new representation of the outputs for each layer by composition (Goodfellow et al., 2016). Additionally, the training of deep architectures is made by the back-propagation as in Artificial Neural Networks. Thus, the chain rule can transfer the higher layers derivatives to the first layers like in MLPs.

Deep Learning represents a set of techniques based on Neural Networks. The main two architectures designed for classification are the Convolutional Neural Networks and Recurrent Neural Networks. These two techniques differ in the kind of input they predict, Convolutional Neural Networks are prepared to classify spatial signals, and Recurrent Neural Networks are prepared to classify temporal signals. The following sections describe the details of these networks.

2.3.1 Convolutional Neural Networks

In the cases where inputs are high dimensional, MLPs have tons of training parameters. The main idea to overcome this problem is to take the local representation which describes the whole input instead of taking the global representation. This local information is represented in Convolutional Neural Network (CNN) (Lecun et al., 1998) using a Neural Network with little parts of the input as Equation 2.22,

$$h(c, r) = f(\mathbf{W}x(c - \Delta c : c + \Delta c, r - \Delta r : r + \Delta r) + b) \tag{2.22}$$

where $h(c, r)$ represents the output taking a region of shape $(2\Delta \times 2\Delta)$ in the position (c, r) of the input x, f is a non-linear function, $\mathbf{W}$ and b are the weights and bias of the Neural Network, respectively. The same Neural Network is applied over the entire input sliding the region as a filter. Furthermore, a pooling layer applied to the convolution output simplifies and reduces the information of the filters. In this stage, the layer needs to capture the more relevant features to preserve the information of the input.

CNN was firstly proposed for handwritten character recognition in Computer Vision. Thus, Figure 2.12 represents the proposed CNN taking a three channel image (RGB) where the convolution generates different filters and pooling is applied to extract the relevant parts of the inputs. Moreover, a new convolution and pooling layers are applied to the previous output for generating more complex features and create a better representation of the raw image. Ideally, the deeper layers of the network capture complex shapes of the image starting from straight lines until detecting complete forms like faces. Once the input is simplified to a vector, a Softmax layer performs the classification of the picture.

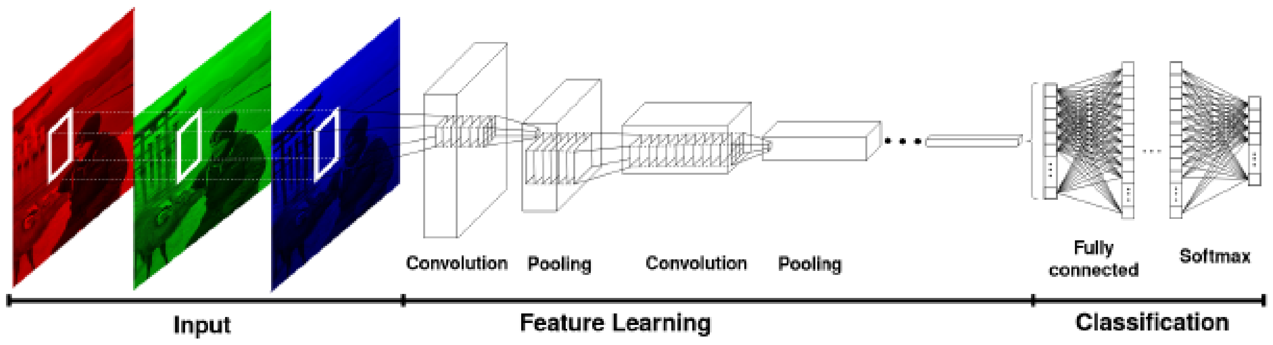


Figure 2.12: Convolutional Neural Network applied to a RGB image for its classification.

Similarly, CNN can be applied to NLP tasks where the inputs are the vector representation of each word in a sentence (Kim, 2014). However, the shape of the filters has to be the dimension of the word vector and a predefined context window for computing the convolution operations. Then, the same operation is applied sliding the filter for the context window of each word in the sentence. Different sizes of context window could be selected for discovering new feature representations. After the convolution layer calculates the resulting matrices, the pooling and classification layers perform the same process previously defined.

2.3.2 Recurrent Neural Networks

In the cases where inputs are sequences, MLPs cannot capture the temporal dependencies of the input dynamically. The main idea to overcome this problem is to take the information given by the previous step in order to compute the current input in a sequential manner. Recurrent Neural Network (RNN) (Rumelhart et al., 1986) uses two different weights for the input and for the previous output as Equation 2.23,

$$h(t) = f(\mathbf{W}x(t) + \mathbf{U}h(t-1) + b) \tag{2.23}$$

where $h(t)$ represents the output in the time t of the input x, f is a non-linear function, $\mathbf{W}$ are the weights for the current input, $\mathbf{U}$ are the weights for the previous output, and b the bias term of the Neural Network. Concretely, the weights and bias are shared over the entire input. Optionally, a pooling layer can simplify all the time-step outputs and reduce the global representation into a vector. The output of the pooling layer needs to capture the more relevant features to preserve the input information.

The image on the left of Figure 2.13 represents the definition of a simple RNN where the current state is computed with the input and the previous output. The image on the right of Figure 2.13 represents the unfolded RNN version where each time-step the system performs an output.

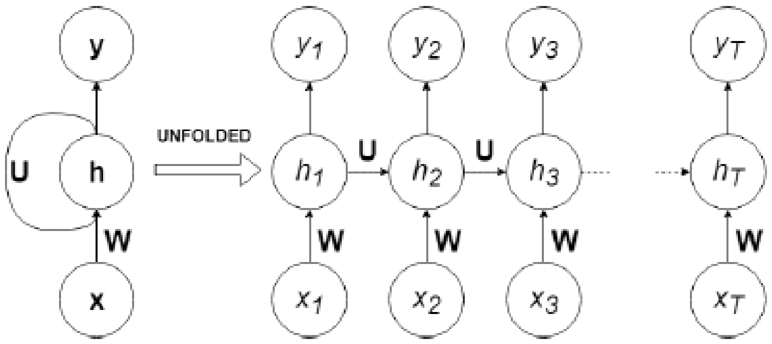
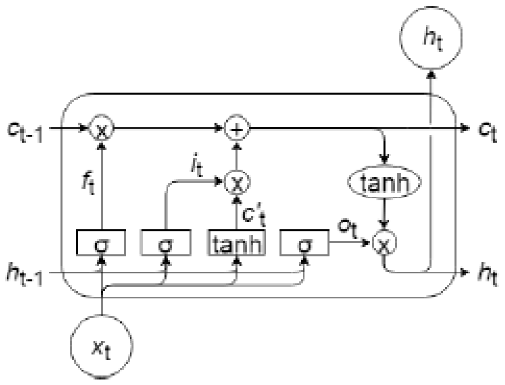


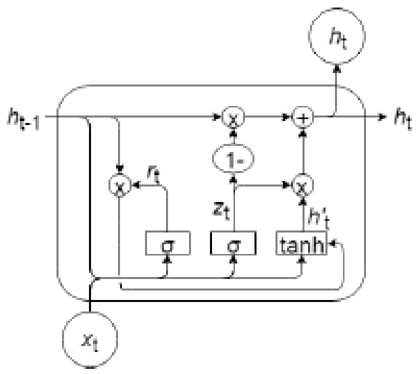
Figure 2.13: Recurrent Neural Network and its unfolded version.

The simple RNN cannot capture the long dependencies because it loses the information of the gradients as long as the back-propagation is applied to the previous states. For this reason, the incorporation of cell units into the RNN computation solves the long propagation of the gradient problem. These unit cells introduce a gating mechanism to remember the values of previous intervals (see Figure 2.14).

The Long Short-Term Memory cell (LSTM) (Hochreiter and Schmidhuber, 1997) defines three gates in order to control the information that flows into the cell and a cell state that is transferred to the next step. The input gate i_t, the forget gate f_t and the output gate o_t for the



(a) Long Short-Term Memory cell.



(b) Gated Recurrent Unit cell.

Figure 2.14: Recurrent Neural Networks cells.

current t step transform the input vector x_t taking the previous output h_{t-1} using its corresponding weights and bias computed with a sigmoid function. The cell state c_t takes the information given from the previous cell state c_{t-1} regulated by the forget cell and the information given from the current cell c'_t regulated by the input cell using the element-wise represented as $*$ in the Equation 2.24. Finally, the current output h_t is defined by the hyperbolic function of the cell state and regulated by the output gate.

$$\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
c'_t &= tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\
c_t &= f_t * c_{t-1} + i_t * c'_t \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
h_t &= o_t * tanh(c_t)
\end{aligned} \tag{2.24}$$

The Gated Recurrent Unit cell (GRU) (Cho et al., 2014) defines the input gate z_t and the reset gate z_t which take the input vector x_t of the current t time and the previous output h_{t-1} using its corresponding weights and bias computed with a sigmoid function. The current cell information is captured by the vector h'_t using the input and the previous output regulated by the reset gate with the hyperbolic function as Equation 2.25. Then, the output vector h_t is regulated by the reset gate choosing the most significant information between the current and previous cells.

$$\begin{aligned} z_t &= \sigma(W_z \cdot [h_{t-1}, x_t] + b_z) \\ r_t &= \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \\ h'_t &= tanh(W_h \cdot [r_t * h_{t-1}, x_t] + b_h) \\ h_t &= (1 - z_t) * h_{t-1} + z_t * h'_t \end{aligned} \tag{2.25}$$

On the one hand, LSTM has a memory unit and a cell state that retain the information of the previous steps. On the other hand, GRU is more efficient because it has fewer parameters to be trained. However, the performance of both cells is very similar to (Chung et al., 2014).

2.3.3 Vanishing gradient and Exploding gradient

The main problem in Deep Learning architectures is the computation of the back-propagation. The multiplication by the chain rule of the derivatives of previous layers obtained from higher layers makes gradients unstably smaller or larger at each step (Nielsen, 2015).

Commonly, the small derivatives multiply the new derivatives and make them smaller as the back-propagation is performed losing the information of the error made by the first layers. This problem is known as the vanishing gradient and makes Deep Neural Networks converge very slowly (Hochreiter et al., 2001). The ReLU activation function solves this issue because it makes gradient zero for negative values.

Contrary, the multiplication of large derivatives with the weights makes that the cost function changes in long ranges. This problem is known as exploding gradient and makes the Deep Neural Networks oscillates with big steps before converging (Pascanu et al., 2013). This issue is solved trimming the gradients obtained by each layer with a predefined threshold (gradient clipping).

In addition, choosing the correct weight initialization, optimization and regularization for the specific problem can prevent these problems.

Chapter 3

Related Work

This chapter describes the systems proposed for NER and RE in biomedical texts. Concretely, the description of the methods will be focused on the pharmacovigilance domain. Current attempts to address these tasks will be presented taking into account the core architecture they use. Thus, the description of the systems for the two NLP tasks are divided by Feature engineering and Deep Learning models.

Moreover, the DDIExtraction Shared Task will be presented together with the dataset of this task, the DDI Corpus, which will be the benchmark for comparing the performance of the models and establishing which ones are better and why. The DDIExtraction Shared Task is composed of NER and RE tasks, the Drug Name Recognition (DrugNER) and the Drug-drug Interaction Extraction (DDI), respectively. At the end of this chapter, a final remark of the current techniques and a discussion about them will be presented.

3.1 SemEval-2013 Task 9: DDIExtraction Shared Task

The main goal of the pharmacovigilance is the early detection of adverse drug reactions, which is a vital process in order to prevent the increasing drug safety incidents as well as their high associated costs for drug agencies and pharmaceutical companies (Bond and Raehl, 2006).

Drug-drug interactions are alterations in the level of activity and the effects of a drug when another drug is co-administrated. IE techniques can be applied to medical literature extracting the important information about the drug entities and their possible relationships in the sentences and reducing the time-consuming of reviewing these documents by healthcare professionals.

NLP challenges are usually organized to develop IE system in the biomedical domain and to provide a common framework where the different techniques can be evaluated and compared. Concretely, the DDIExtraction Shared Task 2013 (Segura-Bedmar et al., 2013, 2014) is the second edition of the DDIExtraction Shared Task series, a community-wide effort to promote the implementation and comparative assessment of NLP techniques focused on the field of the Pharmacovigilance domain. This competition proposes two main subtasks: the recognition of pharmacological substances (DrugNER task) and the detection and classification of drug-drug interactions (DDI task) from biomedical texts.

The DDIExtraction Shared Task provides a benchmark dataset, the DDI Corpus (Herrero-Zazo et al., 2013), which is a manually annotated text collection. This corpus is a gold standard data for training and evaluating supervised machine-learning algorithms to extract DDIs from texts. The DDI Corpus contains 233 selected abstracts about DDIs from MedLine (DDI-MedLine) as well as 792 other texts from the DrugBank database (DDI-DrugBank) for 18,502 pharmacological substances and 5,028 interactions. Table 3.1 shows the basic statistics on the training and test datasets for the DrugNER and DDI tasks.

The creation of guidelines guaranteed the quality of the annotations. Besides, the inter-annotator agreement (IAA) measures the consistency of two annotators, which is an upper bound for the performance of the automatic systems. The agreement of the annotators was very high for the DDI-DrugBank dataset (91.04% for the entities and 83.85% for the relationships) and moderate for the DDI-MedLine (79.62% for the entities and 65.13% for the relationships) because MedLine abstracts have a higher complexity than DrugBank texts that usually have simpler sentences.

The recognized drugs in the sentences are categorized into four different entity types. The description of each type and an example are shown as follows:

Table 3.1: Statistics on the training and test dataset for the DDIExtraction Shared Task.

		Training	Test for DrugNER task	Test for DDI task
DDI-DrugBank	documents	572	54	158
	sentences	5675	145	973
	drug	8197	180	1518
	group	3206	65	626
	brand	1423	53	347
	drug_n	103	5	21
	mechanism	1260	-	279
	effect	1548	-	301
	advice	819	-	215
	int	178	-	94
DDI-MedLine	documents	142	58	33
	sentences	1301	520	326
	drug	1228	171	346
	group	193	90	41
	brand	14	6	22
	drug_n	401	115	119
	mechanism	62	-	24
	effect	152	-	62
	advice	8	-	7
	int	10	-	2

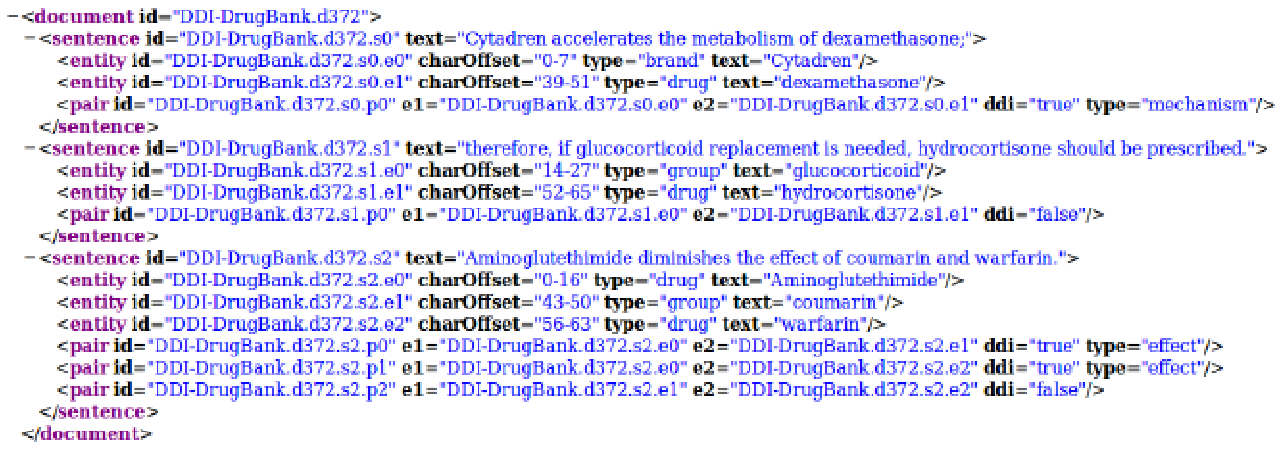
- *drug* type describes the generic name of human medicines (e.g. Penicillin).
- *brand* type represents human medicines known by a trade or brand name (e.g. ERGO-MAR).
- *group* type includes the groups of drugs (e.g. Bacteriostatic antibiotic).
- *drug_n* type is an active substance not approved for human use (e.g. Heroin).

Additionally, the DDI Corpus describes four different types of DDI relationships. Bellow, some sentences of the DDI Corpus illustrates the different types of DDIs:

- *mechanism*: This DDI type describes a pharmacokinetic (PK) mechanism (e.g. Grepafloxacin may inhibit the metabolism of theobromine).
- *effect*: This DDI type represents an effect (e.g. In uninfected volunteers, 46% developed rash while receiving SUSTIVA and clarithromycin) or a pharmacodynamic (PD) mechanism (e.g. Chlorthalidone may potentiate the action of other antihypertensive drugs).

- *advice*: This type of relationship includes a recommendation or advice about a drug interaction (e.g. UROXATRAL should not be used in combination with other alpha-blockers).
- *int*: This type of relationship is a DDI without any additional information (e.g. The interaction of omeprazole and ketoconazole has been established).

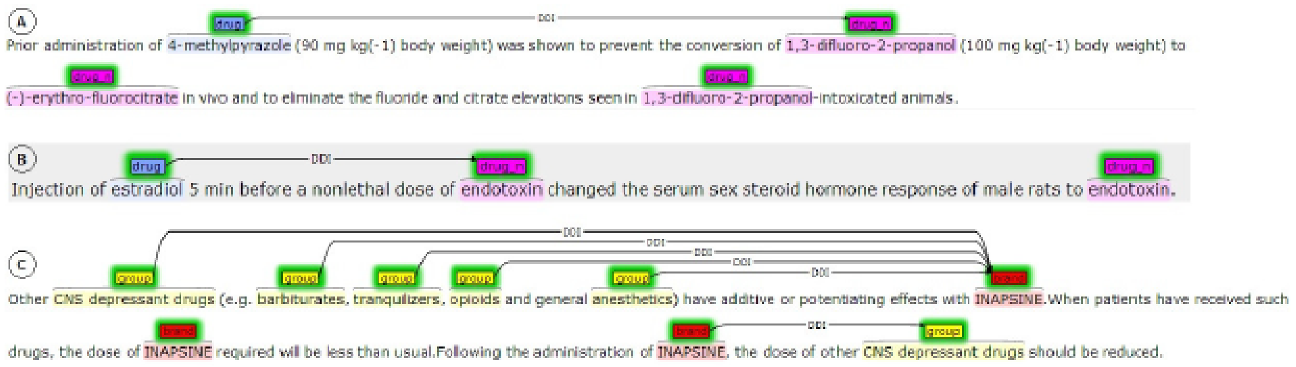
The corpus was split into the training and test datasets in order to evaluate the different participating systems. The training dataset was taken randomly from around 77% of the documents, and the remaining was selected for the test dataset. These documents were annotated in XML following the unified format for PPI corpora proposed in (Pyysalo et al., 2008) (see Figure 3.1). (Herrero-Zazo et al., 2013) presents a detailed description and analysis of the DDI Corpus and its methodology.



```
- <document id="DDI-DrugBank.d372">
  - <sentence id="DDI-DrugBank.d372.s0" text="Cytadren accelerates the metabolism of dexamethasone;">
      <entity id="DDI-DrugBank.d372.s0.e0" charOffset="0-7" type="brand" text="Cytadren"/>
      <entity id="DDI-DrugBank.d372.s0.e1" charOffset="39-51" type="drug" text="dexamethasone"/>
      <pair id="DDI-DrugBank.d372.s0.p0" e1="DDI-DrugBank.d372.s0.e0" e2="DDI-DrugBank.d372.s0.e1" ddi="true" type="mechanism"/>
    </sentence>
  - <sentence id="DDI-DrugBank.d372.s1" text="therefore, if glucocorticoid replacement is needed, hydrocortisone should be prescribed.">
      <entity id="DDI-DrugBank.d372.s1.e0" charOffset="14-27" type="group" text="glucocorticoid"/>
      <entity id="DDI-DrugBank.d372.s1.e1" charOffset="52-65" type="drug" text="hydrocortisone"/>
      <pair id="DDI-DrugBank.d372.s1.p0" e1="DDI-DrugBank.d372.s1.e0" e2="DDI-DrugBank.d372.s1.e1" ddi="false"/>
    </sentence>
  - <sentence id="DDI-DrugBank.d372.s2" text="Aminoglutethimide diminishes the effect of coumarin and warfarin.">
      <entity id="DDI-DrugBank.d372.s2.e0" charOffset="0-16" type="drug" text="Aminoglutethimide"/>
      <entity id="DDI-DrugBank.d372.s2.e1" charOffset="43-50" type="group" text="coumarin"/>
      <entity id="DDI-DrugBank.d372.s2.e2" charOffset="56-63" type="drug" text="warfarin"/>
      <pair id="DDI-DrugBank.d372.s2.p0" e1="DDI-DrugBank.d372.s2.e0" e2="DDI-DrugBank.d372.s2.e1" ddi="true" type="effect"/>
      <pair id="DDI-DrugBank.d372.s2.p1" e1="DDI-DrugBank.d372.s2.e0" e2="DDI-DrugBank.d372.s2.e2" ddi="true" type="effect"/>
      <pair id="DDI-DrugBank.d372.s2.p2" e1="DDI-DrugBank.d372.s2.e1" e2="DDI-DrugBank.d372.s2.e2" ddi="false"/>
    </sentence>
  </document>
```

Figure 3.1: Example of an annotated document of the DDI Corpus extracted from (Segura-Bedmar et al., 2014).

Figure 3.2 shows some examples in brat format (Stenetorp et al., 2012) of annotated texts extracted from the DDI Corpus. The first example (A), taken from the MedLineDDI dataset, describes a DDI of mechanism type between a drug (named using a synonym different from its most common generic name, fomepizole) that inhibits the metabolism of a substance not approved for humans (1,3-difluoro-2-propanol). The second example (B) is also a sentence taken from MedLine and describes the consequence of a DDI (effect type) between estradiol (a generic drug) and endotoxin (a substance not approved to for humans) in an experiment performed in animals. The last example (C) is a paragraph from the DDI-DrugBank dataset. Its first sentence describes the consequence of the interaction (effect type) of a drug, denominated by its brand name (Inapsine) when it is co-administered with five different groups of drugs. The third sentence in C shows a recommendation to avoid these DDIs (advice type).



Prior administration of 4-methylpyrazole (90 mg kg(-1) body weight) was shown to prevent the conversion of 1,3-difluoro-2-propanol (100 mg kg(-1) body weight) to (-)-erythro-fluorocitrate in vivo and to eliminate the fluoride and citrate elevations seen in 1,3-difluoro-2-propanol-intoxicated animals.

Injection of estradiol 5 min before a nonlethal dose of endotoxin changed the serum sex steroid hormone response of male rats to endotoxin.

Other CNS depressant drugs (e.g. barbiturates, tranquilizers, opioids and general anesthetics) have additive or potentiating effects with INAPSINE.When patients have received such drugs, the dose of INAPSINE required will be less than usual.Following the administration of INAPSINE, the dose of other CNS depressant drugs should be reduced.

Figure 3.2: Some examples of sentences in the DDI Corpus extracted from (Segura-Bedmar et al., 2014).

A common evaluation metric must be defined in order to measure the performance of the participant systems and make a top ranking leaderboard from this official score. Besides, the performance of each class individually is also measured for the two subtasks. Taking the right labels and the predictions generated by the models, the measures applied for the classification are defined as follows:

- True Positives (TP) are the true instances predicted as positives.
- True Negatives (TN) are the false instances predicted as negatives.
- False Positives (FP) are the false instances predicted as positives.
- False Negatives (FN) are the true instances predicted as negatives.
- Accuracy (Acc) is the proportion of true results among the total of results calculated as Equation 3.1.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

- Precision (P) is the proportion of true positives among the total of predicted as positives calculated as Equation 3.2.

$$P = \frac{TP}{TP + FP} \tag{3.2}$$

- Recall (R) is the proportion of true positives among the total of true instances calculated as Equation 3.3.

$$R = \frac{TP}{TP + FN} \tag{3.3}$$

- F-measure (F1) is the harmonic mean of precision and recall calculated as Equation 3.4.

$$F1 = 2\frac{R * P}{R + P} \tag{3.4}$$

Concretely, the official score for the DrugNER task is the Strict evaluation, which includes the recognized mentions with the same boundaries and class. In the case of the DDIExtraction subtask, the evaluation metrics must score the ability of the models for detecting the possible interactions between drugs and the class they belong. To do so, the Micro-Average is chosen as the official score whose metrics are calculated globally adding the individual statistics of all the classes (TP, FP, FN).

3.2 State-of-the-art for Named Entity Recognition in the biomedical domain

This section shows the related works for the DrugNER task of the DDIExtraction Shared Task with a description of the proposed systems and the results obtained and the conclusions of the study. Firstly, the methods based on feature engineering are described afterwards, the Deep Learning systems are presented.

3.2.1 Feature Engineering systems

The winner system of the DrugNER subtask of DDIExtraction Shared Task 2013 was the system of (Rocktäschel et al., 2013). The model uses a CRF with general domain features, which was applied to many other NER tasks (Klinger et al., 2008; Lin and Wu, 2009; Sha and Pereira, 2003). However, the authors conclude that adding specific domain features can improve the results. Firstly, the authors create an independent-domain feature setcollected from (Klinger et al., 2008; Leaman and Gonzalez, 2008; Settles, 2005). In addition, the four preceding tokens, the four following tokens and the appearance of upper-case characters in the sentenceare extracted to compose the general feature set for the first run. Secondly, a domain-specific feature set that contains information from the ChemSpot system (Rocktäschel et al., 2012), Jochem dictionary (Hettne et al., 2009), PHARE (Coulet et al., 2011) and ChEBI (de Matos et al., 2010) ontologiesis added to the previous set for the second run. 10-fold cross-validation is performed for training the CRF model with the DrugNER dataset. A third system is trained using the best features given by cross-validation that contains the datasets of the DrugNER and DDI subtasks. The performance of the configuration with the selection of features given by the cross-validation reaches 71.5% in F-measure. The main conclusion of this work is that defining a specific domain feature set extracted from health resources and adding it to a general purpose feature set boost the performance in the DrugNER task.

The system of (Liu, Tang, Chen, Wang and Fan, 2015) uses 8 types of manually defined features with a CRF for the DrugNER task. The feature set includes the words, POS tags, chunks, orthographical, affixes (prefixes and suffixes of lengths of 3, 4 and 5), word shapes (generalized and brief word class), dictionary frequencies (using DugBank, Drugs@FDA and Jochem) and word embeddings. This method is the first CRF model that takes the word embedding of each word in the sentence as the features for the extraction of drug entities. The skip-gram model of Word2vec tool is trained over MedLine abstract in order to extract a 50-dimensional vector representation of each word. After that, a k-means model performs the clustering in 400 semantic classes of these vectors to be used as the features. Experiments with different features conclude that using a combination with pairs of features obtains 78.37% in F-measure and using the feature selection reaches to 79.36%, which chooses the best feature subset of the conjunctions. Concretely the best feature selection method is given by the Information Gain method when the number of features is reduced to 40% (from 294782 to 117913). The authors provide results for the whole DDI Corpus divided by type of entities obtaining a state-of-the-art performance of 79.36% in F-measure for the classification of drug entities. However, the performance of their system on each dataset of the corpus is unknown and whether this system can overcome the system of (Rocktäschel et al., 2013) on the MedLine dataset.

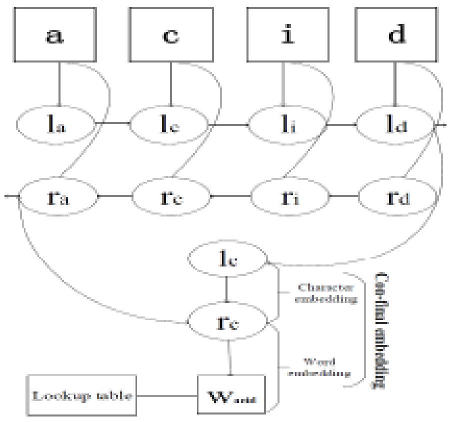
The authors of (Liu, Tang, Chen, Wang and Fan, 2015) presented a work of the same architecture without using the feature selection (Liu, Tang, Chen and Wang, 2015). The article presents the performance of 89.7% and 68.25% in F-measure for the datasets of DrugBank and MedLine, respectively. Despite being a later work, there is no novelty in this system. The main idea of the work of (Zeng et al., 2016) is to extend drug dictionaries with a semi-supervised learning technique in order to enrich the dictionary features used for the classification of drug entities with a CRF. For this end, a CRF with features extracted from the words, such as POS tags, affixes, affixes that appear in the dictionary, one-word drug name, word shape, word embedding as (Liu, Tang, Chen, Wang and Fan, 2015) and the enlarged dictionaries of DrugBank, Drugs@FDA and Jochem performs the classification of named entities in DrugNER of DDIExtraction Shared Task. The system reaches 89.26 in F-measure using the external features, adding the dictionary features and the word embedding feature. However, it is not clear if the results are given only in the type of entity *Drug* because the system performs higher than the (Liu, Tang, Chen, Wang and Fan, 2015) system, which includes the same feature set with the same classifier. A proof of this is that the comparison table used in their study for the results of (Liu, Tang, Chen, Wang and Fan, 2015) and (Rocktäschel et al., 2013) are the same as they individually reported for the *Drug* type. Thus, it is not possible to claim that this system is the state-of-the-art in DrugNER.

3.2.2 Deep Learning systems

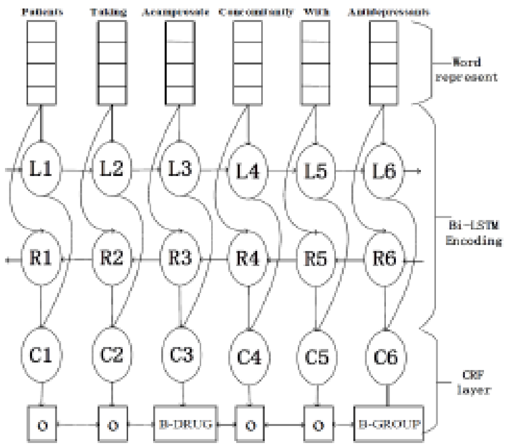
The authors of (Sadikin et al., 2016) propose three different data representation taking the information about the distribution and similarities of the word embeddings. Then, each data representation is evaluated with an MLP, and three different deep learning techniques Deep Belief Network (DBN), Stacked AutoEncoder (SAE) and RNN-LSTM. This article is the first work that uses deep learning techniques for the DrugNER task. All the architectures use word embeddings from Word2vec using CBOW in a context window length 5 and a vector dimension 100. The LSTM is better than the MLP, DBN or SAE systems for extracting drug names and has 78.59% and 94.3% in F-measure over MedLine and DrugBank datasets, respectively. However, the authors claim that they only extract the types of *Drug* and *Drug_n* to evaluate the experiments, whereby the results are only given for the *Drug* type. The final results are measured with the average of the MedLine and DrugBank which is not a fair comparison with the rest of state-of-the-art techniques.

The paper (Chalapathy et al., 2016) explores the use of three RNNs architectures for the task of drug name recognition, Elman network (Elman, 1990), Jordan network (Jordan, 1997) and a bidirectional RNN with LSTM cells together with a CRF classifier (Lample et al., 2016). This is the first work that applied the RNN model to the whole corpus of DDI giving the results for each drug type in DrugBank and MedLine. The authors split the dataset using 30% of the training set for validation and fine-tuning the hyper-parameters of the networks for obtaining the number of hidden layers nodes 25, 50, 100, the context window size 1, 3, 5, the dimension of the embeddings 50, 100, 300, 500, 1000 and the dropout rate sampled from an uniform distribution within the range [0.05, 0.1]. Early stopping criteria over 100 epochs was used to determinate the best performing model on the validation set. The bidirectional LSTM-CRF is significantly better than the other two RNN models obtaining 52.75% in F-measure for the MedLine dataset and 85.19% for the DrugBank set. The results for both datasets are very close to the task winner system in the DrugNER Task (Rocktäschel et al., 2013) without the external knowledge and random embedding initialization.

The system proposed in (Zeng et al., 2017) defines the character embeddings of each word together with the word embeddings for the DrugNER task. Concretely, a bidirectional RNN with LSTM extracts a vector representation of the words taking their characters as the input of the network. Firstly, the system takes each sentence and divides each word into characters. These characters are the input of a bidirectional LSTM whose final computed vectors of both directions are concatenated to form the char level representation of the word (see Figure 3.3a). The authors collect a character set that consists of 26 upper and lower case letters, 10 numbers and 33 punctuation. Then, the imput of the bidirectional LSTM is the concatenation of the character embeddings of each word and the word embeddings. Finally, the output of the bidirectional LSTM is the input for a CRF classification layer in order to identify those tokens which are drugs and classify their types (see Figure 3.3b). The performance of the model obtains 79.26% in F-measure being higher than the winner system in the DrugNER Task (Rocktäschel et al., 2013). However, the authors do not provide results for DrugBank and MedLine individually.



(a) Character embedding representation of the word *"acid"*.



(b) Bidirectional LSTM with a CRF classifier.

Figure 3.3: A bidirectional LSTM with word and character embedding with a CRF classification layer. Images extracted from (Zeng et al., 2017).

The performance of the model obtains 79.26% in F-measure being higher than the winner system in the DrugNER Task (Rocktäschel et al., 2013). However, the authors do not provide results for DrugBank and MedLine individually.

The model described in (Unanue et al., 2017) is an extension of the work of (Chalapathy et al., 2016). Apart from a basic bidirectional LSTM-CRF, the authors explore the use of a CRF and an LSTM with a Softmax classification layer incorporating character representation learnt by a bidirectional LSTM applied to each word. Figure 3.3 represents the whole framework of the bidirectional LSTM-CRF. This system concatenates the word embedding and a character embedding represented by the two output vectors of a bidirectional LSTM model. As word features, the word embedding model is trained using Glove over the general purpose corpus, the Common Crawl (cc) (Pennington et al., 2014). Additionally, the model is extended with the concatenation of cc with the health domain dataset MIMIC-III embeddings (cc/mimic) (Johnson et al., 2016). Furthermore, the input embedding of the model incorporates morphological, semantic and clustering features of the words giving a 146-dimensional vector (features).

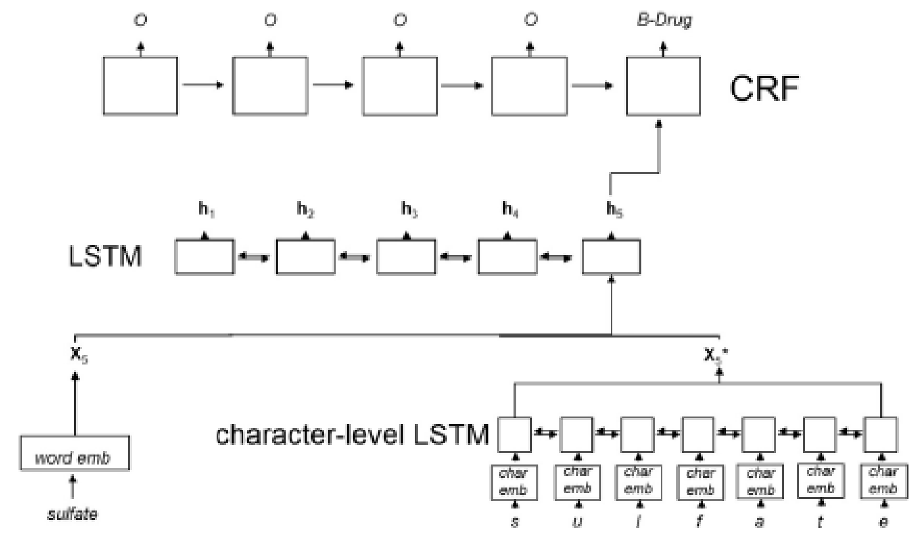


Figure 3.4: The drug names detection system using a bidirectional LSTM-CRF and character embedding information learnt by a bidirectional LSTM. Image extracted from (Unanue et al., 2017).

The experimental results suggest that the bidirectional LSTM-CRF model outperforms the others being 60.66% and 88.38% its performance in F-measure MedLine and DrugBank datasets, respectively. It seems that adding the features do not improve the results. Moreover, the use of the extended model with MIMIC embedding increases the results of the LSTM models. Furthermore, it is better to aggregate character embeddings than only using the word embeddings. Contrary to (Chalapathy et al., 2016), the model does not perform a concatenation of the word embeddings in a window of the context. Besides, the authors test the pre-trained word embedding models over general and specific domain corpus, and they evaluate the performance of aggregating the character embeddings. Consequentially, these changes improve the results in the task of classifying drug name entities.

3.3 State-of-the-art for Relation Extraction in the biomedical domain

This section shows the related works for the DDI task of the DDIExtraction Shared Task with a description of the proposed systems and the results obtained and the conclusions of the study. Firstly, the methods based on feature engineering are described afterwards, the

Deep Learning systems such as Convolutional Neural Networks, Recurrent Neural Networks and Hybrid Networks are presented.

3.3.1 Feature Engineering systems

The winner system of the DDI subtask of DDIExtraction Shared Task 2013 was the model of (Chowdhury and Lavelli, 2013*b*). The authors create a two-stage system for recognizing and classifying drug-drug interaction in biomedical documents. The main ideas are to take the scope negation cues and the semantic roles information in order to discard the negative instances, and to use two SVM models with a hybrid kernel that performs the recognition and the classification sub-sequentially.

As a preprocessing step, the sentences are tokenized, tagged and parsed using the Charniak-Johnson parser (Charniak and Johnson, 2005) with a biomedical parsing model from (McClosky et al., 2010). Furthermore, the syntactic dependencies are extracted from the parse tree with the Stanford parser (Manning et al., 2014). Firstly, the scope of negation discards the less informative sentences (LIS) and the semantic roles, and contextual evidence discards the less informative instances (LII). Secondly, a hybrid kernel, which is defined as $K_{Hybrid}(R1, R2) = K_{HF}(R1, R2) + K_{SL}(R1, R2) + w * K_{PET}(R1, R2)$, determines the representation of a relation between entities. This kernel is composed by a feature based kernel K_{HF} from (Chowdhury and Lavelli, 2013*a*), the Shallow Linguistic kernel K_{SL} from (Giuliano et al., 2006), and the Path-enclosed Tree kernel K_{PET} from (Moschitti, 2004) where w is a weight that assigns the information taken from this kernel. The model computes the SVM-Light-TK toolkit (Joachims, 1999) in order to obtain the kernel. Then, cross-validation of 5-fold tunes the SVM hyper-parameters for performing the detection of the interactions between drugs. Finally, four separate one vs all SVM recongnizes and classifies each drug relationship.

The system outperforms the other participants in the DDIExtraction Task (Segura-Bedmar et al., 2014) obtaining 65.1% in F-measure. The work shows that the multi-phase RE system performing the identification and classification of DDIs with different trained model overcomes the single step systems. Moreover, the proposed hybrid kernel is very useful for this specific task.

The work of (Kim et al., 2015) explores the detection and classification of DDIs with two SVMs with linear kernel functions. For this end, a rich set of features are designed to capture the lexical and syntactic information of the sentences. Additionally, the models implement the *one-vs-one* strategy for the classification of DDIs, which is very effective for dealing with imbalanced datasets.

In the first step, a preprocessing phase is performed over the DDI collection that involves sentence segmentation, tokenization, POS tagging and syntactic parsing that generates constituent parses tree and dependency graphs. As part of the preprocessing, numbers are replaced to the tag "NUM", and the drug mentions are anonymized to ensure generalization fo the features using a common name for the two target drugs, "DRUG", and another name for the rest of the named drugs in the sentence, "DRUG_OTHER". Furthermore, some instances are removed whether the two target drugs refer to the same name and the detailed description sentences where a drug is named followed by a colon and a description, in this cases the left part of the colon is removed. The features used for the representation of each interaction are Word feature, Word pair feature, Dependency graph feature, Parse tree feature and Noun phrase-constrained coordination feature.

An SVM classifier with a linear kernel performs the recognition of drug pairs that interact taking the set of features for each instance. Sub-sequentially, the pairs detected are classified by a multi-class SVM with *one-against-one* strategy. Contrary to the *one-against-all* strategy where a class is compared with the remaining classes, *one-against-one* strategy compares only two classes and the predicted class is determined by a majority voting of all the classifiers. *One-against-one* strategy produces better performance in the fewer representative classes than the *one-against-all* strategy obtaining 67% in the classification of DDIs, which is higher than the winner of the DDIExtraction Shared Task (Chowdhury and Lavelli, 2013*b*). The authors conclude that the linear kernel with the rich feature set is enough to perform the detection and classification of drug-drug interaction and the use of *one-against-one* strategy solves the imbalanced problems of this dataset.

The system of (Zheng et al., 2016) creates a new type of nonlinear kernel, called context vector graph kernel. This graph kernel extracts the surrounding information of each token in the dependency graph and the linear sequence of the sentences.

At first step, the texts are cleaned and normalized filtering out some negative instances like relationships that involve the same entity name, replacing the multi-token entities by a single token and numbers to "NUM". Moreover, the target drugs are replaced to "DRUG1" and "DRUG2", and the remaining entity mentions to "DRUG0". Once the sentences are preprocessed, two graphs representations extract the contexts of each token in different distances. On the one hand, the Stanford parser gives the dependency subgraph 0f the sentence and the POS tags of the words. The edges in the shortest path between the interacting drugs are weighted to 0.9 and the remaining edges are set to 0.3. On the other hand, the sequential order of the words and their POS tags in the sentence are taken as a linear subgraph. The three edges from the three previous tokens of "DRUG1" until the three following tokens of "DRUG2" are weighted to 0.9, and the remaining are set to 0.3. After taking these subgraphs, a graph kernel based on the context of each node is computed based on the similarity of each graph. This similarity is the sum of the weights of all paths of context vector pairs of all layers.

Two SVMs with the computed kernels performs the detection and the classification of DDI sentences, which are trained by 5-fold cross-validation to adjust the parameter of the models. The performance of this model reaches 68.4% in F-measure for the classification of DDIs. The contextual information extracted from the proposed kernel based on graphs is more effective than the performance of previous kernel approaches.

The authors of (Raihani and Laachfoubi, 2016) define a two-stage system that firstly, performs the detection using an SVM with lexical features, correction patterns and trigger words, and secondly, performs the classification with multiple *one-vs-all* SVM which work sequentially and take the lexical features of the sentence as input.

This system proposes three enhancements with respect to the system of (Bui et al., 2014) for the detection of the DDIs. A title correction preprocessing is defined with rules to identify correctly the sentences structures that involve the definition of drugs as (Kim et al., 2015). In addition, 203 new trigger words are added to the previous list increasing to 501 the number of trigger words. Finally, the definition of new features helps to capture more information about possible drug interactions. Unigrams and bigrams of lemmatized tokens and grammatical tags are extracted as features. In addition, some features are added if any dependency type describes a negation, if a path exists between the interacting drugs and if the target drugs belong to the same coordinate structure.

By analysing the special lexical field comparing a class with other classes, the authors suggests to build three subsequent *one-vs-all* classifiers for the classification of each detected DDI: *Advice-vs-all* SVM with RBF kernel, *Mechanism-vs-(Effect+Int)* SVM with linear kernel and *Effect-vs-Int* SVM with linear kernel (System_1). In addition, a fourth *Int-vs-all* SVM classifier with linear kernel separates the Int class at the beginning in order to have high precision in this class (System_2).

The results suggest that the subsequent architectures (System_1 and System_2) are better than the multi-class SVM model and that the performance of the high precision classifier at the beginning of the architecture obtains better results with 71.14% in F-measure.

The system proposed in (Raihani and Laachfoubi, 2017) is an adaptation of their previous work (Raihani and Laachfoubi, 2016). The main difference is that the authors create a new set of features for the detection of sentences involving drug interactions.

The preprocessing of texts is very similar to the previous system, but the two interacting drugs are replaced to the string "ARG" in the drug blinding step. Compared to the previous work, this system defines a different feature set taking the unigrams and bigrams of the lemmatized tokens and the main-verb with the position information, the surrounding words of each drug mention, and the n-grams windows. In addition, some features are added if any dependency type describes a negation and if the target drugs are in the same chunk.

The DDI classification is a cascade of *one-against-all* SVM classifiers which is the same architecture defined in (Raihani and Laachfoubi, 2016). The experimental resultsshow that the rich feature set defined by this work outperforms the previous system based on kernels. The two-stage method reaches 71.79% which is state-of-the-art for these techniques in the DDI classification task.

The works for extraction of DDIs only exploit the information about drug interactions in the sentence level. However, some drugs in the DDI sentences, such as branded name or generic name, appear frequently and their information is very valuable for other sentences. For this reason, the study presented in (Tu et al., 2017) takes corpus-level features to perform the classification in the DDI Corpus.

The proposed system involves a drug name normalization and the calculation of odds ratio, which is the number of occurrences of a drug divided by the times this drug interacts with other drugs. For the classificaiton step, 5 SVM classifiers with RBF kernel includes basic features together with template feautes from the odds ratio calculation. A majority voting of the classifier outputs gives the final class prediction.

This model achieves an F-measure of 65.3% in the classification subtask and is better than the winner system of the DDIExtration Task (Chowdhury and Lavelli, 2013*b*). The work shows that the corpus-level features are important and could be applied for future works, but the experimental results are quite far from the state-of-the-art results on this task.

3.3.2 Convolutional Neural Network systems

The authors of (Liu, Tang, Chen and Wang, 2016) explore a Convolutional Neural Networks (Lecun et al., 1998) for DDIExtraction Task, which consists of four layers: a look-up table layer, a convolutional layer, a max pooling layer, and a Softmax layer.

The sentences of the corpus are tokenized, converted to lower-case and preprocessed using drug blinding, which replaces mentions by labels. Additionally, padding tokens are added until reach the maximal length of tokens given by the longest sentence in the corpus. Furthermore, a negative instance filtering discards the non-interacting drug pairs in the cases that two drugs have the same name, one drug is the abbreviation or acronym of the other, two drugs appear in the same coordinate structure that has more than two drugs as elements or one drug is a special case of another one.

After the preprocessing phase, the look-up table generates the representation of the DDI instances concatenating the word embeddings and two position embeddings which describes the distances with respect to each drug of interest. In the convolutional layer, the matrix of a DDI sentence generates features in a context window of k words with filters of different size. Then, these filters generate a feature vector in all the context windows. In the max pooling layer, the maximum value of the features for each filter creates the new vector of features. Thus, the dimensionality of the vector is reduced to the length of filters.

Afterwards, in the Softmax classification layer, each feature is selected randomly with a Bernoulli probability in order prevent over-fitting during the training phase. Once the training has finished, the feature vector of the max pooling layer is fed directly to the Softmax layer without dropout in the test phase.

The model obtains 65.00% without position embeddings and negative instance filtering. The architecture that aggregates both techniques achieves a 69.75% in F-measure. This work demonstrates that Deep Learning techniques can outperform feature engineering system like SVM for DDIExtraction Task.

The major limitation of CNN is the lack of good representation in long-distance dependencies for DDI instances. The capture of this representation is helpful for a good performance where the interaction has long distances between the drugs in the sentence. To overcome this problem, the work of (Liu, Chen, Chen and Tang, 2016) proposes a Dependency-based Convolutional Neural Network (DCNN) to improve the CNN model using convolution operation over the dependency parse tree of the DDI instances.

The same preprocessing of (Liu, Tang, Chen and Wang, 2016) is defined including the padding of the sentences, the drug blinding, the negative instance filtering and the position embedding. The tokenizer of NLTK (Bird, 2006) performs the dependency parse tree of a sentence, then the Charniak-Johnson Parser (Charniak, 2000) generates the constituent parse tree and then the PyStandfordDependencies of the Stanford CoreNLP (Manning et al., 2014) creates the dependency parse tree from the constituent parse tree.

The system is based on their previous work (Liu, Tang, Chen and Wang, 2016), where a CNN with pre-trained word embedding classifies the DDI sentences. Besides, three parallel convolutional layers are defined in this model regarding the concatenation of the word embeddings. The first operation takes the embeddings of the k adjacent words in the same way as (Liu, Tang, Chen and Wang, 2016), the second operation takes the embeddings of the word and their a ancestor nodes in the dependency parse tree and third operation takes the embeddings of the word, their ancestor and their b sibling nodes in the dependency parse tree. In case the number of ancestors and siblings are less than the size of the window, the model appends to the concatenation the words *ROOT* and *SIBLING* respectively until filling the window.

The dependency parser produces wrong trees for long DDI instances which affects the results in performance. To overcome this problem, the authors also propose a combined method taking the outputs of CNN and DCNN. The CNN classifies the sentences longer than 30 words, while the DCNN classifies the sentences shorter than this threshold. The results show that DCNN obtains 70.19% in F-measure and 70.81% for the combination system.

In (Zhao et al., 2016), the authors create a system called Syntax Convolutional Neural Network (SCNN). In this article, a new syntax word embedding is proposed to capture the syntactic information of the DDI sentences.

The word embeddings of the previous works are based on the surrounding words in the linear order of the sentence. Therefore, these embeddings ignore the syntactic information of the words. For that reason, the authors of (Zhao et al., 2016) propose to pre-train the word embeddings with the Word2vec tool (Mikolov, Sutskever, Chen, Corrado and Dean, 2013) of the words in the shortest path from the dependency parse tree (SDP) given for the Enju parser (Miyao and Tsujii, 2008). Thus, the syntax word embedding captures the syntactic structure of the sentence in a short syntax word sequence. The processes of this model include a negative instance filtering, a biomedical tokenizer (Jiang and Zhai, 2007), and a transformation of numbers regarding if they are integers or decimals.

Feature extraction phase generates two kinds of representation for each DDI sentence. Firstly, a CNN with max-pooling computes an output vector with the concatenation of syntax word embeddings, the relative distances of each word with respect to the two interacting drugs and POS tag embedding from Enju parser. Additionally, a hidden layer generates the final vector for the representation of the DDI sentence. Secondly, a single hidden layer creates other representation taking the concatenation of the syntax vectors given by the two target drug names, their predicate words, their surrounding words, the predicate of their surrounding words, the words in the SDP and their dependencies types. Finally, these representations are concatenated to be classified by a Softmax layer. Furthermore, the authors compare the one-stage classification, which only performs the detection and classification of drug interactions with one classifier, and the two-stage classification, which performs the two subtasks with two classifiers.

The experimental results show that models reach 67% and 68.6% in F-measure using the SCNN with the one-stage method and the SCNN with the two-stage method, respectively. Despite these complex methods, the model does not reach the performance of a simple CNN like in (Liu, Tang, Chen and Wang, 2016). However, this article shows that a two-stage classification method improves the results against a one-stage classification method in the DDI Corpus because of their imbalanced instances.

The work of (Quan et al., 2016) proposes a CNN model that takes multiple channels for representing each word in the sentences (MCCNN). Based on the idea of adding the three parallel channels of the image (RGB) to their classification with CNNs, the system integrates multiple word embeddings from different sources to increase the semantic information of each word.

Five different word embedding models are extracted using the articles of PubMed, PMC, MedLine, and Wikipedia from (Pyysalo et al., 2013) and another word embedding using MedLine with CBOW model. Figure 3.5 shows the CNN architecture for the five channels.

Like the previous works the model uses a negative instance filtering step and replaces the

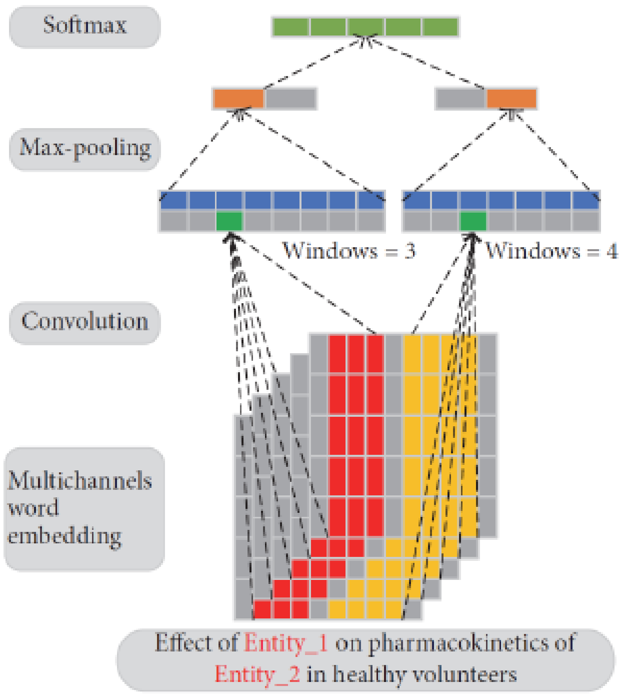

Softmax
Max-pooling
Windows = 3
Windows = 4
Convolution
Multichannels word embedding
Effect of Entity_1 on pharmacokinetics of Entity_2 in healthy volunteers

Figure 3.5: The Multichannel CNN architecture with five different word embedding. Image extracted from (Quan et al., 2016).

two target entities and the remaining entities with *"Entity1"*, *"Entity2"* and *"EntityOther"*, respectively. The MCCNN system with the five channels provides an F-measure of 70.21%, a 10% higher than the randomly initialized word embeddings. This model obtains a good performance without the position embeddings or syntactic information like in the previous works (Liu, Chen, Chen and Tang, 2016; Liu, Tang, Chen and Wang, 2016; Zhao et al., 2016).

Some previous works apply the attention mechanism with RNN, but the performance with CNN has not been explored for the DDIExtraction Task. Similarly to (Wang, Cao, de Melo and Liu, 2016*a*), the authors of (Asada et al., 2017) apply an input attention mechanism to the representation of the embeddings for the DDI sentences before a CNN model.

The whole system is a basic CNN, with embedding layer, convolution layer, max-pooling and classification layer. Texts are preprocessed with drug blinding to replaces the mentions by a predefined name. The model pre-trains the word embeddings with Skip-gram using PubMed articles. In contrast to previous works (Liu, Chen, Chen and Tang, 2016; Liu, Tang, Chen and Wang, 2016; Quan et al., 2016; Zhao et al., 2016), the embedding representation is transformed by an attention mechanism before the convolutional operation. In this case, each word in the sentence receives an attention weights based on the two target drug entity embeddings $\{e_1, e_2\}$, which are

$$\alpha_i = \frac{\alpha_{i,1} + \alpha_{i,2}}{2} + b_\alpha \qquad \text{where, } \alpha_{i,j} = \begin{cases} Softmax(\beta_{i,j}), & \text{if} \quad i \notin \{e_1, e_2\} \\ a_{drug}, & \text{otherwise} \end{cases} \tag{3.5}$$

$\beta_{i,j}$ is the dot multiplication of the word embeddings i and j, a_{drug} is an attention parameter for the entities and b_α is the bias term. After that, each word embedding is multiplied by their corresponding attention weight given by α and fed to the convolutional layer (see Figure 3.6).

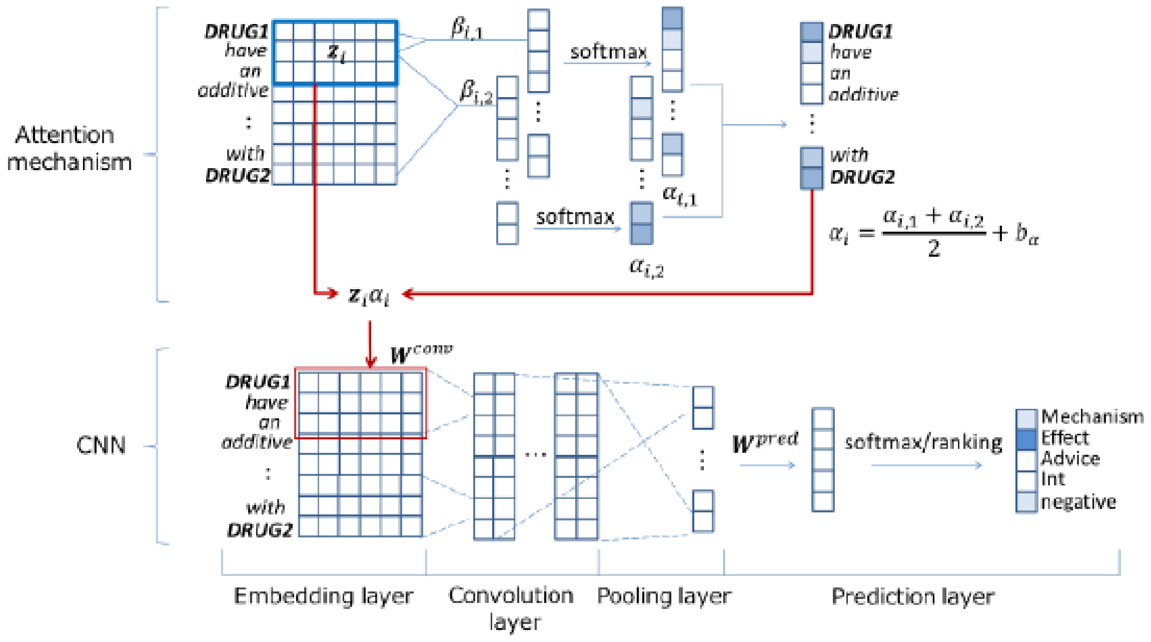


Figure 3.6: The proposed model of CNN with an input attention applied to the word embeddings. Image extracted from (Asada et al., 2017).

Additionally, the authors define two different objective functions for generating the predictions, the traditional Softmax layer and the ranking-based proposed by (dos Santos et al., 2015). The ranking CNN is higher than the Softmax CNN with 69.12% and 67.94% in F-measure, respectively. This system is very competitive to the state-of-the-art-techniques without the negative instance filtering. Concretely, this preprocessing method could increase the results for this model in the classification of DDI sentences.

Few works are focused on deep architectures with multiple layers for the DDIExtraction Task, such as (Huang et al., 2017; Kavuluru et al., 2017; Raj et al., 2017). These authors explore the addition of two or three layers in order to generate the vector representation of the current sentence. However, the authors of (Dewi et al., 2017) propose a deep architecture of 10 CNN layers together with multichannel word embeddings (DeepCNN) in order to learn higher abstraction representation for the DDI instances than previous works.

Firstly, the sentences are converted to lower-case, drug blinded and negative instance filtered. Five-word embeddings are included in the model as the representation of each word, which includes PMC, PubMed, PMC and PubMed, Wikipedia and PubMed, Medline. Then, a CNN with a max-pooling layer generates a vector for a fixed window length from these embeddings. All the resulting vectors for the different window sizes are concatenating as a matrix to be the input for a new CNN with a max-pooling layer. The Softmax layer transforms the last matrix into a vector which represents the probability of each DDI class for the current instance. Figure 3.7 shows an overview of the model for one channel and one window size.

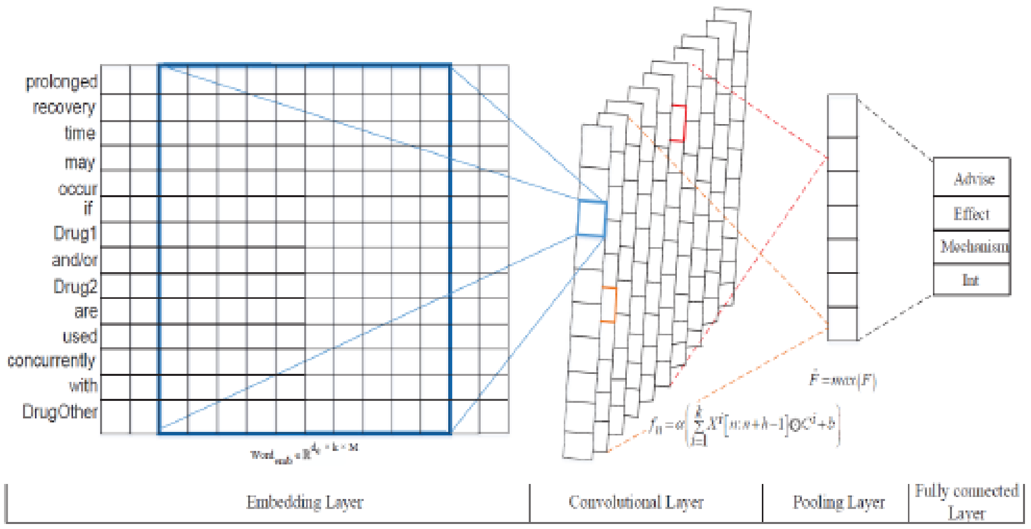


Figure 3.7: The deep architecture of a CNN, where the filters are concatenated to be the input of a second CNN layer. Image extracted from (Dewi et al., 2017).

The study concludes that 10 layers with dropout reach 86.27%, which is highly superior to the state-of-the-art techniques. This work shows that adding more layers to generate more abstraction for DDI instances boost the performance for their classification.

Apart from the feature vector generated by Deep Learning models, external knowledge can be added before the classification layer to extend the vector information representing each DDI sentence. The authors of (Asada et al., 2018) propose to take the molecular graph structure of the two interacting drugs into a Graph Convolutional Network (GCN) in order to encode their internal chemical compounds information.

In order to transform the molecular structure information into a vector, two kinds of GCN are proposed: CNNs for fingerprints (NFP) (Duvenaud et al., 2015) and Gated Graph Neural Networks (GGNN) (Li et al., 2015). Furthermore, an interaction pairs dataset is created from DrugBank for training these methods. Figure 3.8 shows the molecular graph structures of a DrugBank entry given the transformation to the RDKit (Landrum, 2013) of the SMILES encoding (Weininger, 1988). The resulting vector is fed to a fully connected layer and a Softmax layer which predicts if two drugs interact from their molecular structure information.

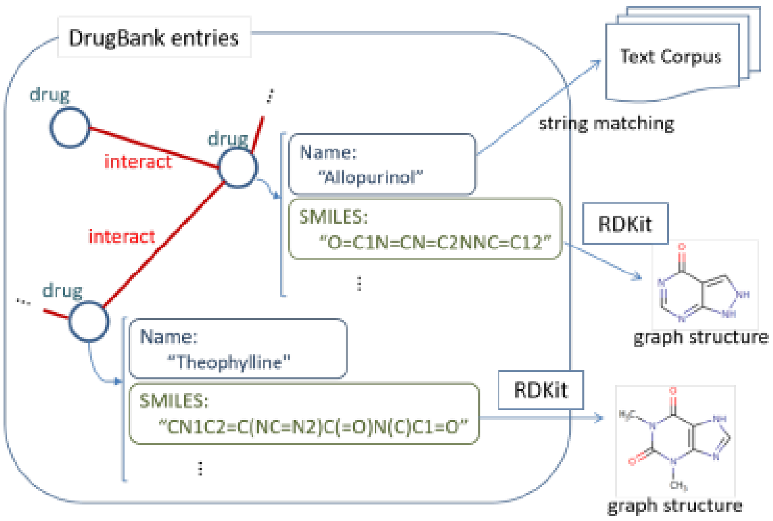


Figure 3.8: Extracting the molecular graph structures from the DrugBank entries. Image extracted from (Asada et al., 2018).

Figure 3.9 shows an overview of the proposed model. Firstly, sentences are preprocessed by the drug blinding and are tokenized by Genia tagger (Tsuruoka et al., 2005). Each word is described by its word embeddings pre-trained with Word2vec on MedLine articles and position embeddings, whose concatenation is fed to a CNN. The max-pooling operation is applied to the resulting matrix together with the GCN output. Then, a fully connected layer aggregates more abstraction to the representation of this vector. Finally, a Softmax layer predicts a DDI class taking the fully connected output.

NFC method obtains 72.21%, while GGNN obtains 72.55%, both in F-measure, which are better than only taking the text information. The molecular structure information has shown

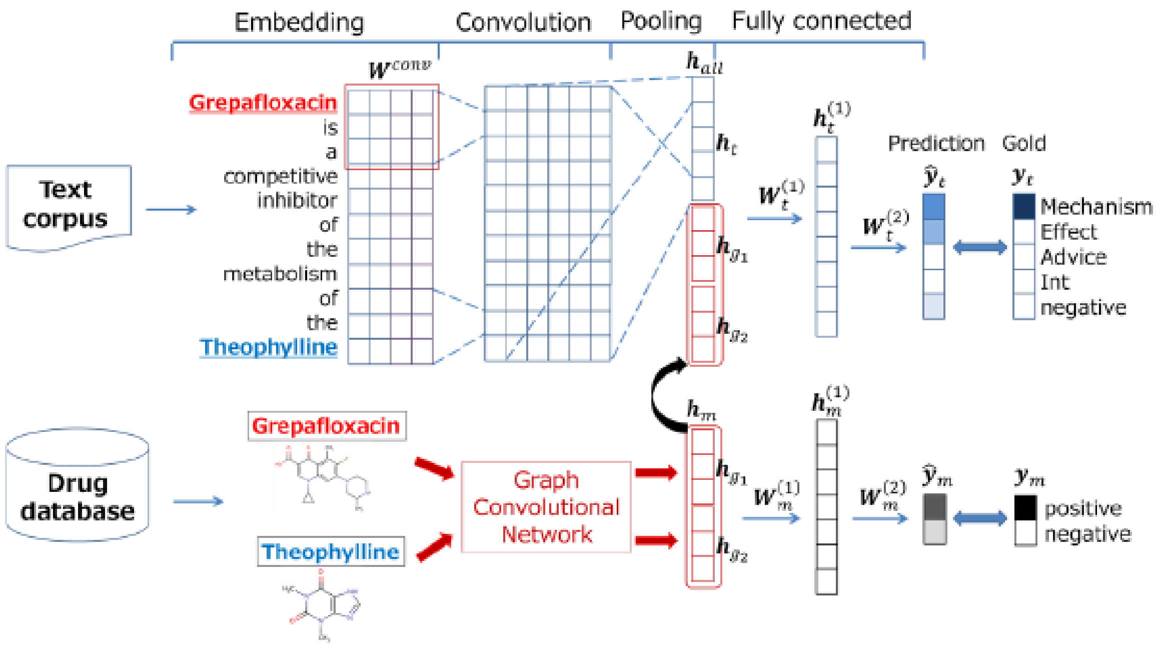


Figure 3.9: The architecture of the Graph Convolutional Network. Image extracted from (Asada et al., 2018).

very valuable and useful in the DDI classification that should be included in future works. The main improvement of this system is to reach a state-of-the-art technique using a single layer of convolution without negative instance filtering.

The Turku Event Extraction System (TEES) (Björne et al., 2014) is a framework for relation and event extraction for the biomedical domain. This system represents the entities and event triggers as nodes and relations and event arguments as edges in a graph. For this end, the model includes a pipeline of four classification tasks, named entity detection, relationship detection of the detected entities, unmerging of overlapping entities and the modifier detection such as speculation or negation. In the original model, a multi-class SVM performs the classification of each pahse. The work of (Björne and Salakoski, 2018) proposes an extension of TEES replacing the SVM classifiers by the CNN architecture.

The four CNN classifiers have the same architecture (see Figure 3.10). The input of the model is composed by the word embeddings, POS tag embeddings, entity type embeddings, distances to the tokens of interest, relative locations in the sentence (Before, Middle or After), dependency path type embeddings, SDP sequence words and event argument. Firstly, sentences are centred with padding from both sides. Later, the words of the sentence are transformed into vectors of 200-dimensional from the word embeddings concatenated with 8-dimensional for each embedding given by POS tag, entity type and position. After that, four convolution layers of 512 filters are applied with filter sizes of 1, 3, 5 and 7 giving a vector of size 2048. A max-pooling layer reduces the resulting matrix of features to a vector. Additionally, a dense layer of 400 neurons is built to this vector before the Softmax classification layer.

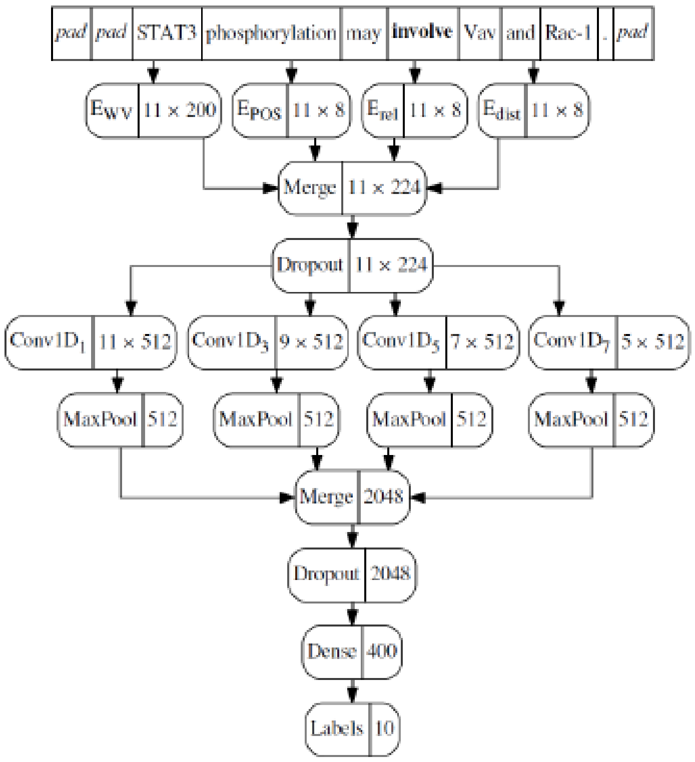


Figure 3.10: Turku Event Extraction System implemented with CNN classifiers. Image extracted from (Björne and Salakoski, 2018).

The authors evaluate the system on 12 different NER, RE and event extraction shared tasks corpora including the DDI Corpus. The implementation of the TEES with CNNs obtains 68% and 73.51% in F-measure for DrugNER Task and DDIExtraction Task, respectively. This system achieves the state-of-the-art results on several corpora and high performance for the DDIExtraction Task.

Similarly to (Dewi et al., 2017), the system of (Sun et al., 2018) explores the addition of multiple layers for a CNN model in order to classify the DDI sentences (DDNet). Following the idea of deep architectures in the field of computer vision, the output of a convolution operation is the input of the next layer.

The authors of the model include a preprocessing of drug blinding and convert to lower-case the sentences from the DDI Corpus. The word embeddings are pre-trained with 300 dimensions using Glove where the unknown words are initialized as an all-zero vector. The matrices that represent the DDI sentences according to the embeddings of their words are fed into a hierarchical CNN with convolutional blocks applied sequentially. A convolutional block is composed by a convolutional layer applied to the input with an activation layer and a batch normalization layer that scales the inputs and accelerate the training of the network. Then, a Softmax layer performs the classification of the sentences over a max-pooling layer applied to the last convolutional block. Figure 3.11 represents the complete framework for the DDNet system.

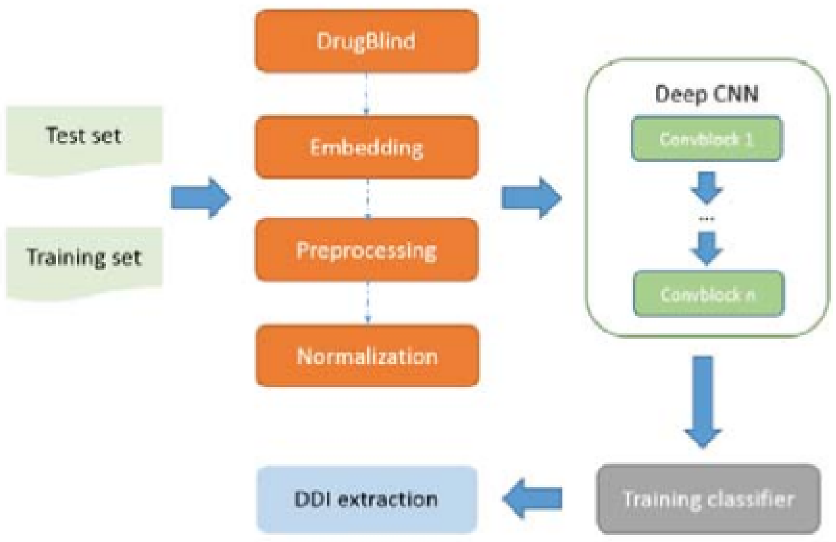


Figure 3.11: The whole system of the DDNet with convolutional blocks applied sequentially. Image extracted from (Sun et al., 2018).

The network with 16 layers of depth obtains the best performance with an F-measure of 84.5% in the classification of DDI sentences. This study and (Dewi et al., 2017) validates that increasing the number of layers used sequentially in the classification networks overcomes the performances of the shallow Deep Learning techniques.

3.3.3 Recurrent Neural Network systems

The work of (Ebrahimi and Dou, 2015) demonstrates that using dependency parse instead of constituency parse in a Recursive Neural Network (RecNN) model (Socher, Lin, Ng and Manning, 2011) improves the performance and reduces the training time for the task of relation classification. The main hypothesis is that dependency graphs can give a more compact representation of the relationship. The architecture is a modification of a RecNN to incorporate dependency graph (DG) nodes where a relationship between entities has a unique common ancestor. Thus, this chain is transformed into a binary tree structure and can apply compositions on the words that belong to the shortest path between entities. In Figure 3.12a, the sentence *"The child was carefully wrapped into the cradle"*, which contains a relationship between the entities *child* and *cradle*, is transformed into a DG.

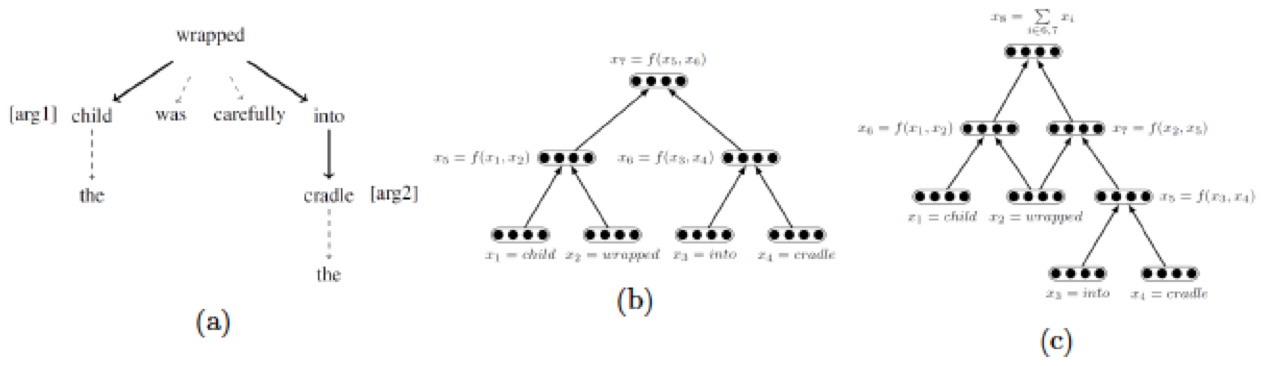


Figure 3.12: The Directed Acyclic Graph representation for the dependency graph of the sentence *"The child was carefully wrapped into the cradle"* in (a): the tree based chain RNN (C-RNN) in (b) and the DAG based chain RNN (DC-RNN) in (c). Images extracted from (Ebrahimi and Dou, 2015).

To this end, two models are proposed with a fixed structure as well as a model with a predicted tree structure. The models with a fixed structure are built with two Directed Acyclic Graph (DAG) structures by heuristics, according to the treatment of a word with two dependencies. The representation of this word is included in one composition, tree-based chain RNN (C-RNN) (see Figure 3.12b), or two compositions and then sum their results, DAG based chain RNN (DC-RNN) (see Figure 3.12b). Moreover, a Recursive Autoencoders (RAE) (Socher, Pennington, Huang, Ng and Manning, 2011) predicts the best tree structure of the chain (C-RNN-RAE).

For all the models, some internal features are added to the classification layer: the depth of the tree, the distance between entities, context words and the type of dependencies. The authors evaluate the systems for the DDIExtraction Task where C-RNN obtains a 68.64% in F-measure versus 67.68% obtained by MV-RNN.

The work of (Sahu and Anand, 2018) proposes three different models with RNNs (Rumelhart et al., 1986) based on LSTM and explores the use of the attention pooling. Unlike CNN, RNN captures the information of data in sequences receiving the output of the previous computation in each step. Concretely, LSTM cells deal with the long dependencies defining a memory mechanism.

As a preprocessing, Genia tagger tool (Tsuruoka et al., 2005) is performed to tokenize the sentences, all digits are replaced with a special token, and all letters are converted to lowercase. Moreover, the drug blinding process to transform the drug names into common names and a similar negative instance filtering is defined as in the previous works. Later, the system creates a layer with the word embeddings and the position embeddings to describe each word in a DDI sentence. Then, a bidirectional LSTM layer (Bi-LSTM) extracts a vector in each word taking the two directions of the sentence, forward and backward. The model simplifies this set of resulting vectors with a pooling layer. The authors evaluate two different pooling layers: the traditional max-pooling layer (Figure 3.13a describes the model B-LSTM), which takes the maximum value of each feature dimension, and a novel attentive pooling, which has a parallel NN to take a weighted linear combination of the feature vectors (Figure 3.13b describes the model AB-LSTM). The final vector is fed to a Softmax layer to generate a prediction of a DDI class. Furthermore, the model Joint AB-LSTM consists of two parallel architectures using the two kinds of pooling and concatenating the outputs for the classification layer (Figure 3.13c).

For the experiments, word embeddings are pre-trained with GloVe method (Pennington et al., 2014) on PubMed articles (TH et al., 2015) with a length of 100. The dimension of the position embeddings are set to 10 and are randomly initialized. The experimental results shows that the joint model performs better than the winner system in the DDIExtraction Task (Chowdhury and Lavelli, 2013*b*) with a 71.48% in F-measure, concluding that attention models should be beneficial for the DDIExtraction Task.

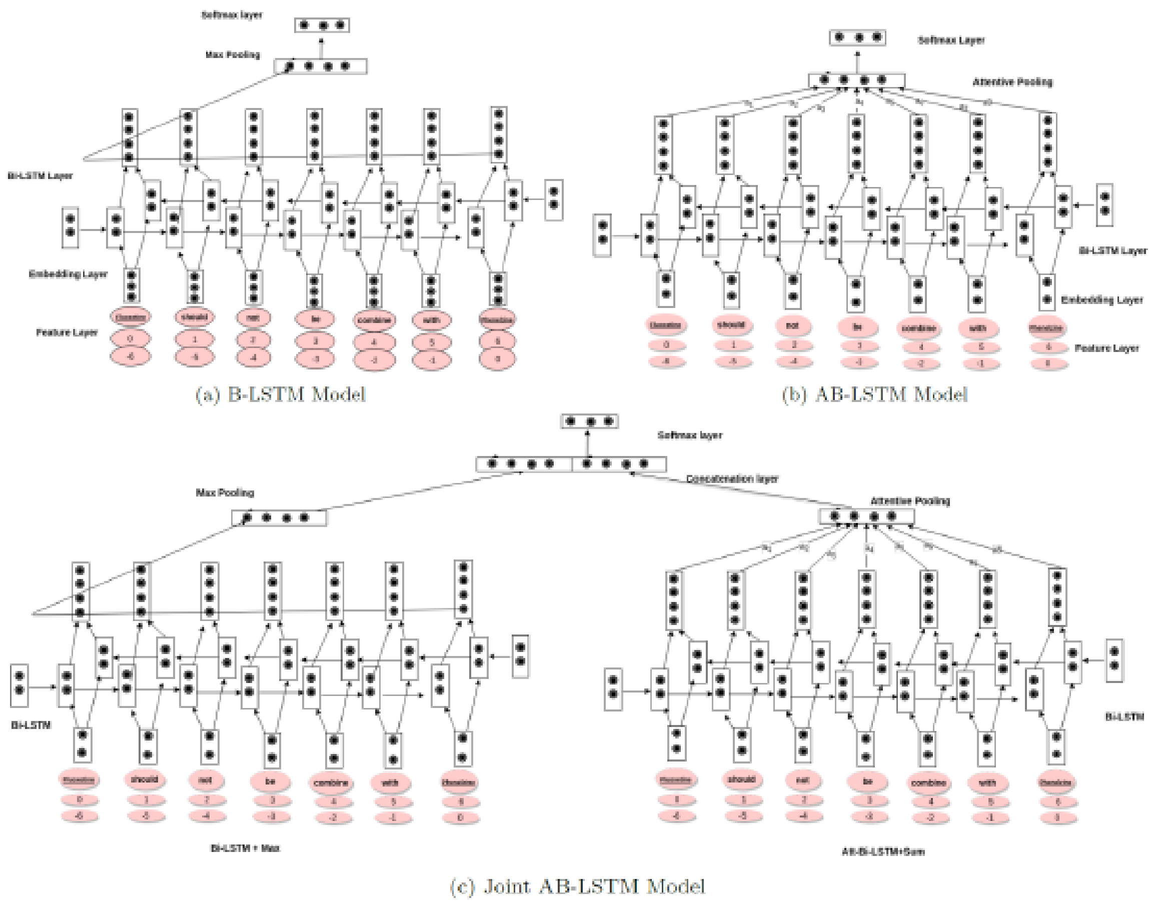


Figure 3.13: The RNN with LSTM cells architectures with max-pooling and attentive pooling. Image extracted from (Sahu and Anand, 2018).

Attention models have shown good performance in Relation Extraction tasks such as (Wang, Cao, de Melo and Liu, 2016*b*; Zhou et al., 2016). Concretely, the work of (Lin et al., 2016) proposes a selective attention layer over instances which exploits the information of all sentences that share the same entity pair. Based on this work, the authors of (Yi et al., 2017) propose the same attention mechanism applied to the DDIExtraction Task.

Figure 3.14 shows an overview of the proposed model. Firstly, each word of the DDI instances are represented by their pre-trained word embeddings with GloVe (Pennington et al., 2014) and the position embeddings. Then, these embeddings are fed to a bidirectional RNN with GRU cells to generate the representation of each word with the addition of the forward and the backward output vectors. After that, an attentive-pooling layer is applied to create the best linear combination of the resulting GRU outputs. Another attention layer is applied to capture the features of sentences that have the same drug pair. Finally, a Softmax classifier calculates the probability of each class for the current DDI instance.

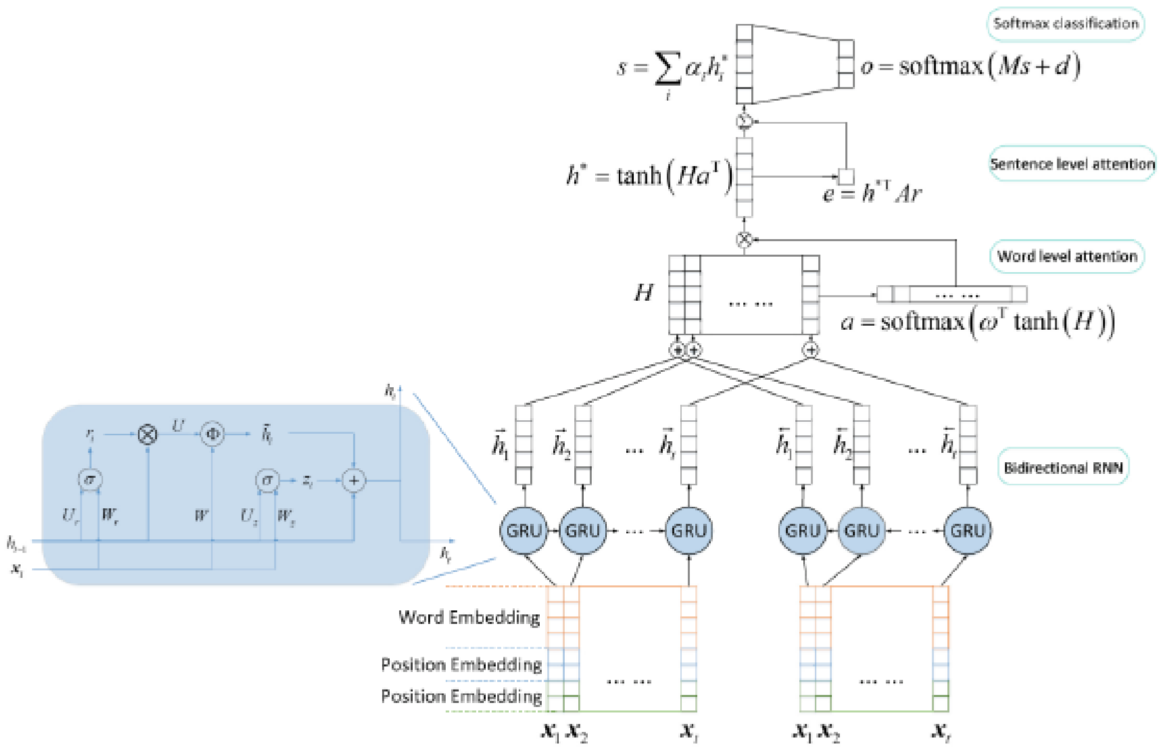


Figure 3.14: The bidirectional RNN model proposed with multiple attentions. Image extracted from (Lin et al., 2016).

The experimental results presented for this model reach to 72.2% in F-measure. However, the prediction results presented in a confusion matrix from the article obtains 67,58% in F-measure with micro-average, which are fewer than other state-of-the-art techniques.

The work of (Zheng et al., 2017) applies the input attention mechanism over word embeddings of an RNN with LSTM cells for the DDIExtraction Task. The system is very similar to the model of (Asada et al., 2017), but they concatenate the position embeddings and POS tag embedding after the performing of the attention mechanism.

The system performs a basic preprocessing of the DDI sentences which include: drug entity blinding, replacing each digit string with a special token and removing bracket characters. Two patterns are defined to solve the co-reference resolution, which contributes to decreasing the number of False Negatives. Besides, the sentences are pruned to a fixed input length longer than the maximal separation between drugs, except the text between them, in order to avoid the redundant information. Shorter sentences are padded with a special token until filling this length.

After the preprocessing, all the words in the pruned sentences are represented by their word embeddings, position embeddings and POS tag embeddings obtained with Stanford Parser. In order to determine which words are the most representative for the DDI relationship, the model applies an input attention mechanism to the word embeddings matrix calculated as

$$\alpha_i = \frac{\alpha_{i,1} + \alpha_{i,2}}{2} \tag{3.6}$$

where $\alpha_{i,j} = Softmax(score(\mathbf{u}_{w_i}, \mathbf{u}_{e_j}))$ being $\mathbf{u}_{w_i}$ and $\mathbf{u}_{e_j}$ the word embeddings of the word w_i and the target entity drug e_j, respectively. However, the attention parameter for entities and the bias term are not used in this article like in (Asada et al., 2017). The output of the attention layer is multiplied to the word embedding vector and is concatenated to the position embeddings and POS tag embeddings which are the input of the neural architecture.

The model defines a bidirectional RNN with LSTM cells to generate two vectors at the end of the forward and the backward directions. Finally, a Softmax layer generates a prediction of a DDI class with the concatenation of the vectors (see Figure 3.15). The pooling layer in this system is avoided for calculating the vector representation of each sentence.

For the experiments, word embeddings are pre-trained using Word2vec with a corpus obtained from PubMed. Furthermore, POS tag embeddings are also pre-trained using Word2vec with the DDI Corpus, while position embeddings are randomly initialized. The experimental results suggest that the co-reference resolution rules and the pruned sentences increase the results. Compared with other works, this model reaches 77.3% in F-measure being the highest performance in the state-of-the-art for the DDIExtraction Task applying the attention mechanisms.

The system of (Wang et al., 2017) integrates the dependency parse tree generated with Stanford Parse of the sentences into an RNN to perform the classification of DDI instances. The authors propose three channels for the model: the linear channel, which takes the sequence of the sentence, the DFS channel, which goes through the dependency parse tree using the Depth First Search algorithm, the BFS channel, which goes through the dependency parse tree using

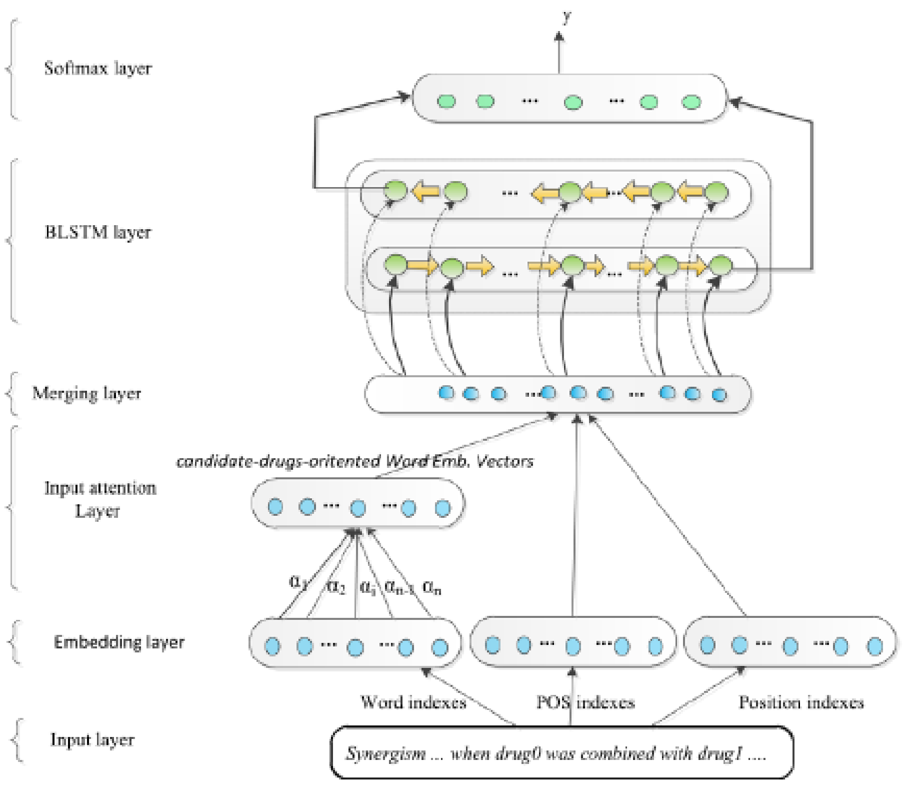

y
Softmax layer
BLSTM layer
Merging layer
Input attention Layer
candidate-drugs-oritented Word Emb. Vectors
α_1 α_2 α_i α_{n-1} α_n
Embedding layer
Word indexes
POS indexes
Position indexes
Input layer
Synergism ... when drug0 was combined with drug1 ....

Figure 3.15: The overall system of a bidirectional RNN with LSTM and an input attention. Image extracted from (Zheng et al., 2017).

the Breadth First Search algorithm.

The embedding layer applies the word embeddings randomly initialized (DLSTM^1) or the syntax word embedding based on Word2vec by the Skip-gram algorithm (DLSTM^2) for each word in the sequence. In addition, the linear channel concatenates the position embeddings of each word, while the embedding layer of the DFS and BFS channels concatenate two distance embeddings that represent the differences between the distance to the root and the distances to the target drug entities of the current word. Similarly, to (Zhao et al., 2016), the position and the distance embeddings, are represented by a ten-bit binary vector. Each channel has a separated bidirectional RNN with LSTM cells which takes the embedding layer as input. The resulting vectors of the two directions RNNs are averaged instead of concatenating them. Then, a max-pooling layer is applied to the averaging outputs to build the vector representation of each channel. A Softmax layer predicts the DDI class from the concatenation of the three channels outputs. Figure 3.16 illustrates the framework of the model.

Due to the imbalanced dataset, a training set sampling is performed to increase the number of

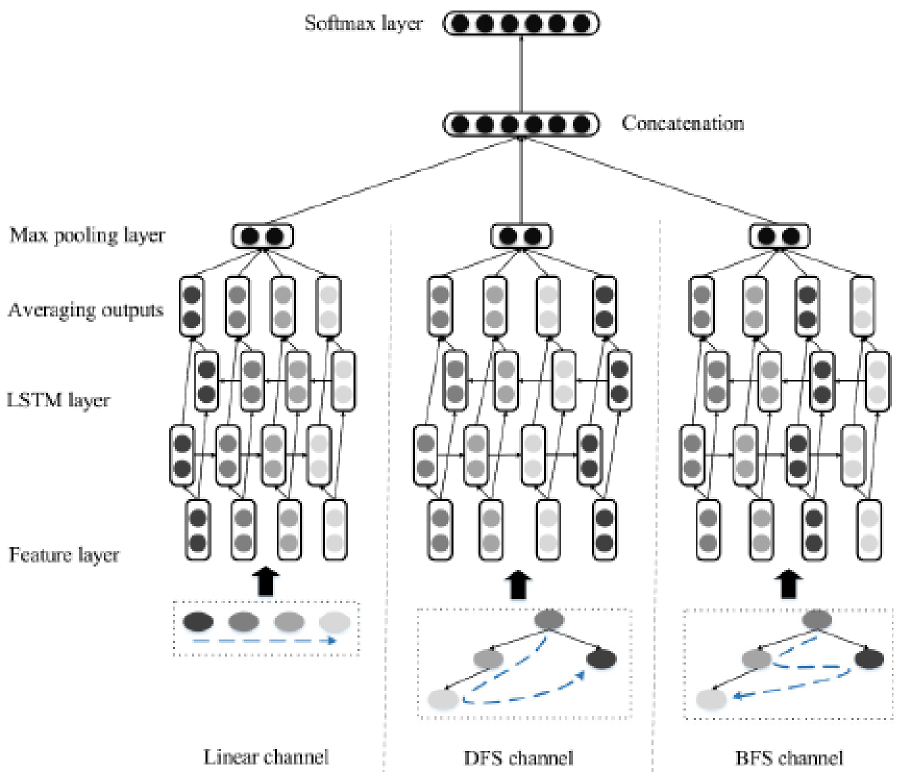


Figure 3.16: The framework of the three channels bidirectional LSTM model. Image extracted from (Wang et al., 2017).

instances for the class *Int* and decrease the instances for the class *None*. Fundamental sampling techniques to this end are the Random under-sampling and the Random over-sampling, which take a fixed number of samples chosen randomly in the sets and discard or repeat them, respectively. Additionally, the preprocessing step includes drug blinding and negative instance filtering as in previous works. The dimension for word embeddings is 100, for the distance embeddings is 10 and for the hidden layer in the RNN is 300. The percentages chosen for the Random under-sampling is 50% and for the Random over-sampling is 50%.

DLSTM^1 obtains 72%, while DLSTM^2 obtains 71.37% in F-measure showing that random word embeddings are better than syntax word embeddings. The experimental results also show that the sampling techniques are very effective over DDI Corpus due to its imbalanced nature. Moreover, averaging the outputs of the bidirectional RNN with LSTM cells is better than concatenating them. Furthermore, DFS and BFS channels increase the results almost 3% points, showing that the dependency information is also valuable for the DDIExtraction Task.

The article (Jiang et al., 2017) proposes a method to define the structure of the DDI instances, which is integrated with an LSTM called Skeleton-LSTM. Four kind of "bones" are defined for each word in the sentence: red bone (R) whether the token is a target drug, blue bone (B) for words between the interacting drugs, green bone (G) for surrounding words of the two drugs of the DDI, and blue-green bone (BG) which are the surrounding words between the interacting drugs.

Each word is embedded in a real value vector using Word2vec. The concatenation of the position embeddings and the "bone" embeddings creates the skeleton embeddings $x_t^{(skeleton)}$ of the word t. Thus, an RNN with LSTM cells computes the concatenation of the word embedding, the hidden layer output of the previous word, and the output of a skeleton gate defined as $S_t = g(\mathbf{W}^S x_t^{(skeleton)} + \mathbf{b}^S)$. Figure 3.17 shows an example of the system.

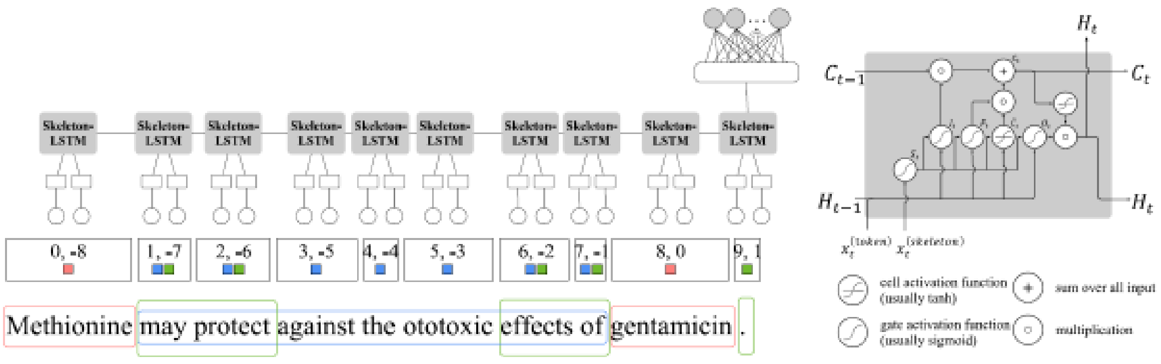

Skeleton-LSTM
0, -8
1, -7
2, -6
3, -5
4, -4
5, -3
6, -2
7, -1
8, 0
9, 1
Methionine may protect against the ototoxic effects of gentamicin .
H_t
C_{t-1}
C_t
H_{t-1}
H_t
$x_t^{(token)}$ $x_t^{(skeleton)}$
cell activation function (usually tanh)
sum over all input
gate activation function (usually sigmoid)
multiplication

Figure 3.17: The skeleton structure included into an RNN with LSTM cells. Image extracted from (Jiang et al., 2017).

This system performs 71.4% in F-measure using a simple RNN with LSTM cells defining an embedding that defines the structure of each DDI sentence.

Most of the previous works take the word embeddings as the representation of each word in the sentence. However, models could exploit lower level information provided by the characters of each word. Thus, the authors of (Kavuluru et al., 2017) investigate the effects of an RNN that takes the information from the character level (Char-RNN) for the DDIExtraction Task. Besides, they propose a bootstrapping method to average multiple models and increase their performance jointly.

Texts are preprocessed converting all the letter into lower-case and performing the drug blinding. Moreover, the sentences are tokenized and represented as a sequence of words. The sentences are zero padded to the longest sentence in the dataset. Two different embeddings are defined to represent each word, the word embedding and the character embedding. The word embeddings are pre-trained using Word2vec with MedLine citations, while 128 ASCII characters are transformed into a vector, and then they are fed to a simple RNN with LSTM cells in order to create the vector taking the output of the last character. Additionally, a position embedding is concatenated to the final vector. After that, a bidirectional RNN with LSTM cells and a max-pooling layer generate the vector that represents the sentence. Finally, a Softmax layer performs the classification (see Figure 3.18).

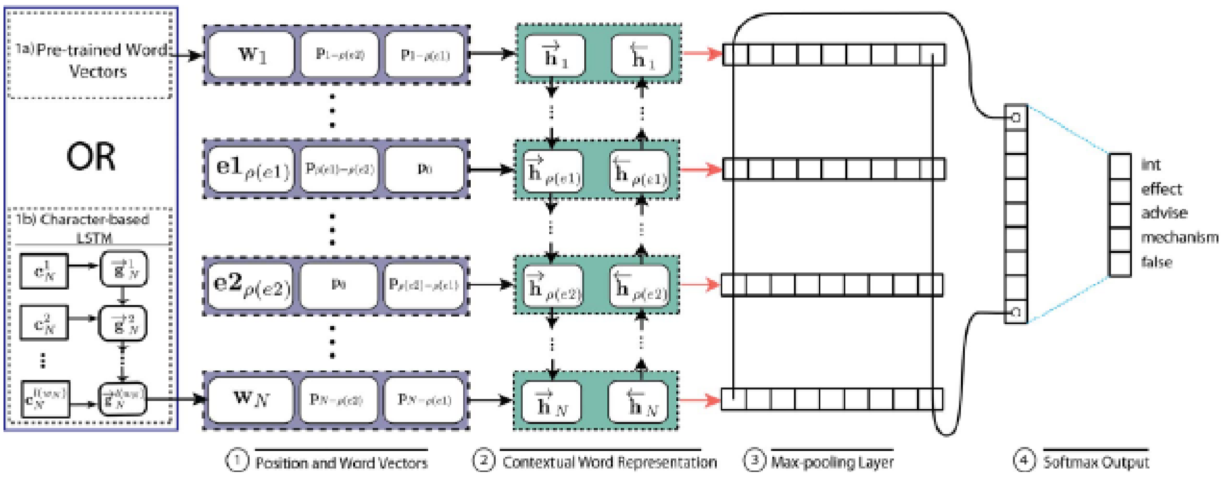


Figure 3.18: Char and word embeddings as the input of a bidirectional LSTM layer for the classification in a DDI type. Image extracted from (Kavuluru et al., 2017).

Additionally, to prevent the model reaches a local minimum, 10,000 ensembles of 10 models (5 of each group) are chosen from 20 RNN with word embeddings and 20 RNN with char embeddings trained with different parameter initialization with 10% of the training set as the validation set.

The ensemble method with negative instance filtering obtains the best performance with 72.13% in F-measure. This article is the first work that explores the character-level information and the aggregation of multiple models knowledge for the classification in the DDI sentences. Furthermore, it demonstrated that 128 characters embeddings are enough to represent the words and reach a similar performance of state-of-the-art techniques. Moreover, another conclusion is that ensemble methods should be considered to increase the results in the DDI Corpus.

The article (Zhang, Zheng, Lin, Wang, Yang and Dumontier, 2018) describes a model that integrates the sentences and its SDP sequence to a hierarchical RNN with an input attention mechanism. Similarly to (Xiao and Liu, 2016), the sentence is divided into five parts: the subsequence of words before the first drug target ($S0$), the first target drug name ($e1$), the subsequence of words between the target drugs ($S1$), the second target drug name ($e2$), and the subsequence of words after the second drug target ($S2$).

Figure 3.19 presents the overall system architecture of the hierarchical model proposed. Each word in the sentence is represented by its word embeddings trained on PubMed abstracts with Word2vec, POS tag embeddings and position embeddings. The POS tags of the entities are set to Noun, and the word embedding for multi-word entities is the mean of the word embeddings of all their words. The model gets the dependency parse tree and the POS tag sequence of the sentence from the Stanford parser. After that, an input attention mechanism transforms the representation of the entire sentence and their SDP. Then, the resulting word embeddings with attention in the sentence is divided into the five parts, where the three subsequences and the SDP sequence are fed to four different bidirectional RNN layers. The concatenation of the two directions output vectors of the RNN creates the vector representation of the sequences. Another bidirectional RNN is applied to the output vectors of $S0$, the word embedding of $e1$, $S1$, the word embedding of $e2$, $S2$ and SDP sequence. The two directions output vectors of this RNN are concatenated and fed to a Softmax classification layer to generate a prediction.

The authors perform experiments with different RNN architectures such as simple, LSTM and GRU. The hierarchical bidirectional RNN with LSTM applied to the SDP sequence and the sentence with the attention mechanism obtains a performance of 72.9% in F-measure. The results of the models conclude that SDP contributes valuable syntactic information and reduces the complexity of the sentence.

Most of the works for the DDIExtraction Task are focused on Deep Learning techniques, concretely in two of them, RNN and CNN. RecNN models are less used due to the difficulties of the parsers for capturing the information in biomedical sentences. However, the authors of (Lim et al., 2018) develop a system based on the parse tree structure of the instances with LSTM cells and perform an ensemble method to improve the performance in the DDIExtraction Task.

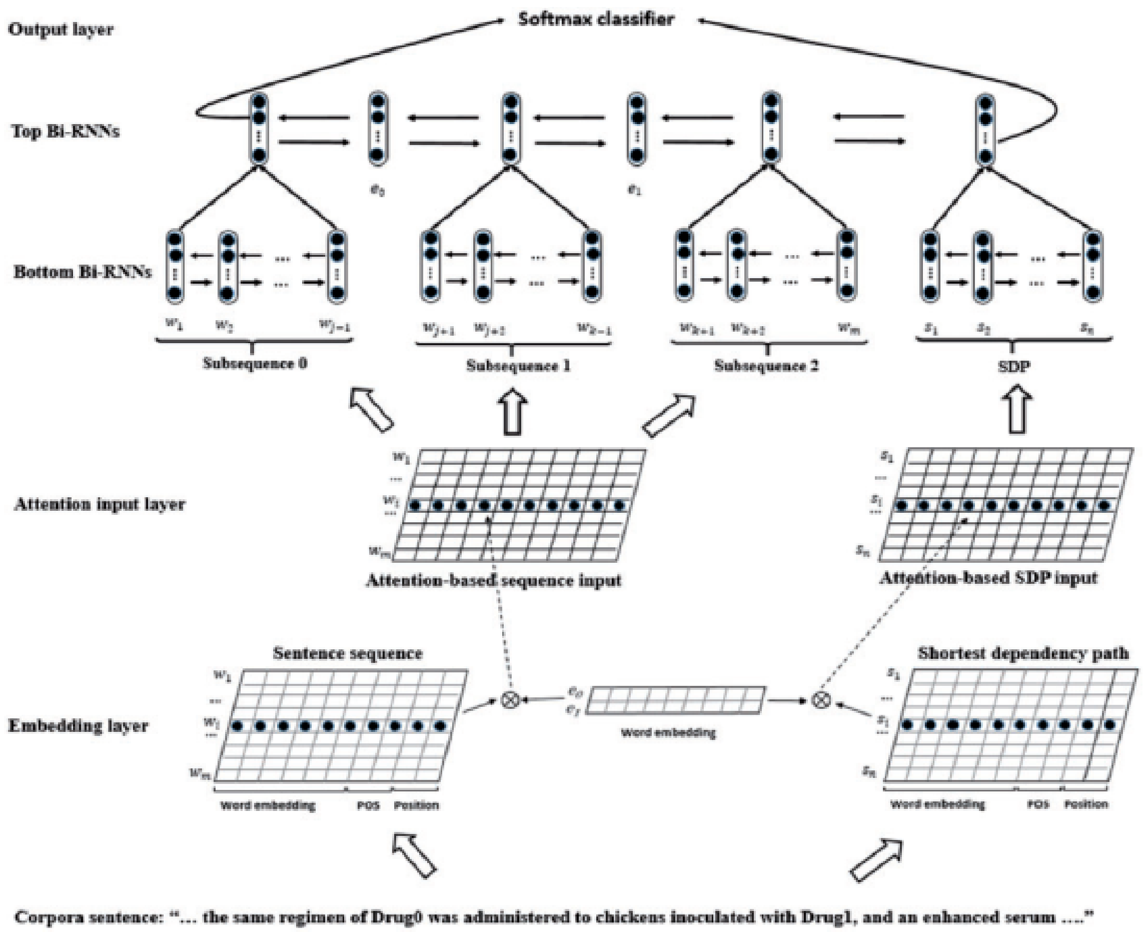


Figure 3.19: The overview of the system taking the SDP and the five parts of the sentences divided by the two entities. Image extracted from (Zhang, Zheng, Lin, Wang, Yang and Dumontier, 2018).

Firstly, the sentences are tokenized, filtered out some negative instances and replaced entities with drug blinding, and numbers with the special token "#". In addition, the Biomedical Entity Search Tool (BEST) (Lee et al., 2016) finds all entities without annotation in the sentences. The Stanford Parser gets the binarized constituency parse tree, which is the structure of the model for each sentence. Word embeddings are pre-trained using Word2vec of gensim tool (Řehůřek and Sojka, 2010) with PubMed-and-PMC-w2v dataset from (Pyysalo et al., 2013).

The architecture of the model is based on the Child-Sum Tree-LSTM from (Tai et al., 2015) (see Figure 3.20). Concretely, if a node is not a leaf, its word representation is randomly initialized, otherwise its word embedding is taken. Each node in the tree structure computes an LSTM cell for the two children, and their outputs are combined in the next node. Finally, the DDI class prediction is generated by a Softmax layer that takes the resulting vector of the root in each tree. Similarly to the model of (Zhao et al., 2016), position embeddings are a bit vector regarding some position ranges. In addition, a subtree containment feature is defined, which indicates whether the current node contains a target drug in its subtree. This model is a two-stage method, which uses two classifiers for the detection and classification sub-sequentially, and as a one-stage method, which uses one classifier for both tasks. Furthermore, an ensemble method generates a prediction aggregating the output probabilities of 10 models trained with different initialization.

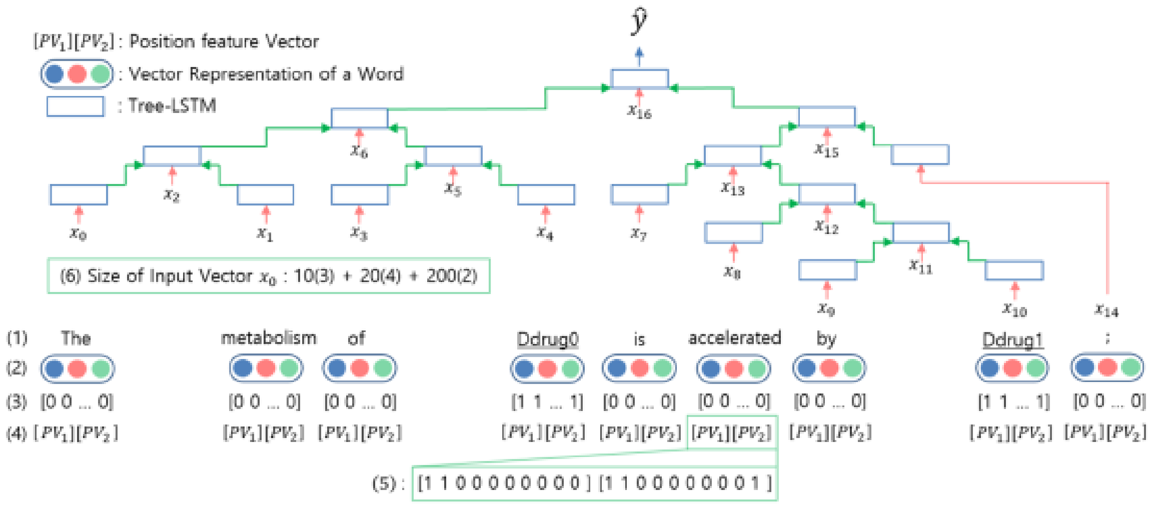

$[PV_1][PV_2]$: Position feature Vector
: Vector Representation of a Word
: Tree-LSTM
$\hat{y}$
x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12} x_{13} x_{14} x_{15} x_{16}
(6) Size of Input Vector x_0 : 10(3) + 20(4) + 200(2)
(1) The metabolism of Ddrug0 is accelerated by Ddrug1 ;
(2)
(3) [0 0 ... 0] [1 1 ... 1]
(4) $[PV_1][PV_2]$
(5) : [1 1 0 0 0 0 0 0 0 0] [1 1 0 0 0 0 0 0 0 1]

Figure 3.20: Illustration of the tree architecture of RNN with LSTM cells. Image extracted from (Lim et al., 2018).

The experimental results report the F-measure for the one-stage method using a single model, 71.7%, and the ensemble model, 73.5%, and for the two-stage method using a single model, 71.4%, and the ensemble model, 72.7%. This article confirmes that the ensemble model outperforms the single model because it prevents to reach in a local minimum. However, contrary to previous studies (Chowdhury and Lavelli, 2013*b*; Huang et al., 2017; Kim et al., 2015; Raihani and Laachfoubi, 2016, 2017; Zhao et al., 2016), the one-stage method for this architecture is better than the two systems for the two tasks. The Tree-LSTM does not seem to be better than the RNN with LSTM cells for the DDI Corpus.

Position embeddings and the input attention mechanism increase the results in most of the previous articles on DDI Corpus. For this reason, the model described in (Zhou et al., 2018) is the first work that combines the attention mechanism over the position embeddings and the outputs of a bidirectional RNN with LSTM cells. Furthermore, the system performs the detection and the classification of DDI classes with two parallels classification layers. Thus, a multi-task learning framework is learned at the same time in order to take inter-tasks knowledge.

Texts are preprocessed with drug blinding and negative instance filtering. Word embeddings pre-trained using Word2vec with PubMed and position embeddings define the vector representation of each word in a DDI sentence. A bidirectional LSTM is applied to these vectors, and the concatenation of the last vectors in the two directions are taken as the feature vector of the sentence. An attention mechanism performs a weighted combination of all the hidden states and the position embedding. Two Softmax layers performs the detection and the classification separately over the resulting vector together with the RNN representation. Figure 3.21 shows the complete system for a DDI instance.

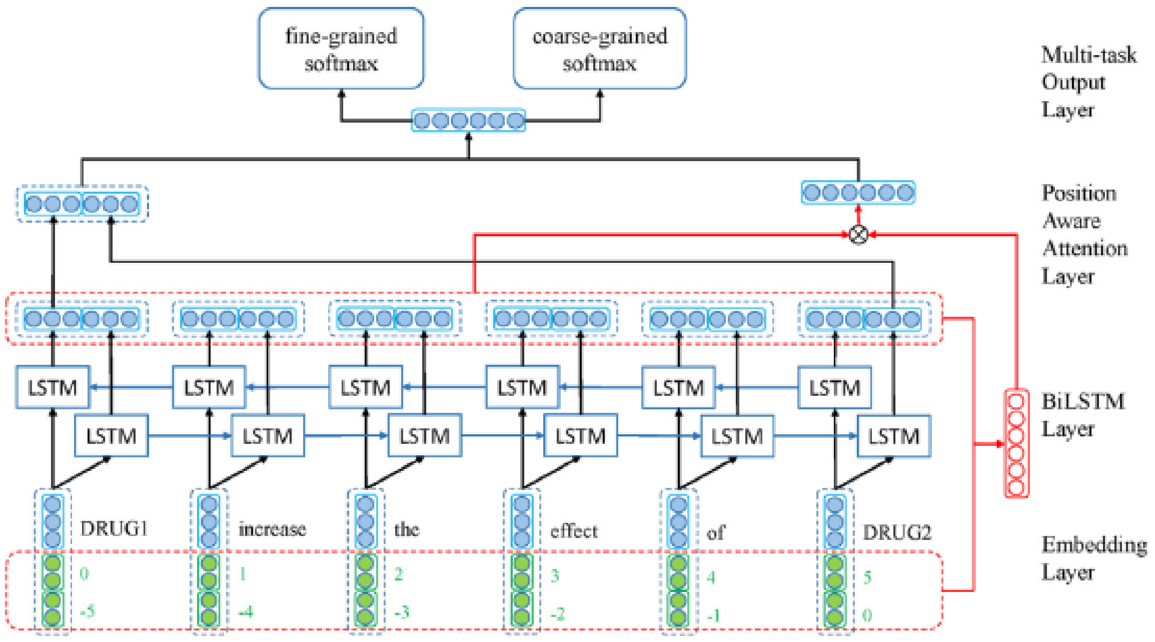


Figure 3.21: Overview of the whole process for the position-aware multi-task bidirectional LSTM. Image extracted from (Zhou et al., 2018).

The experimental results of three variants of the model, the position-aware attention mechanism (P-BLSTM), the multi-task learning (M-BLSTM) and the combination of both techniques (PM-BLSTM) obtains 72.25%, 71.75% and 72.99% in F-measure, respectively. The position-aware attention mechanism and the multi-task learning contribute to increasing the performance obtaining a 72.99% in F-measure.

The recent deep learning techniques applied to biomedical texts do not consider the valuable information that ontologies contain about the specific words in these documents. The system proposed in (Lamurias et al., 2018) integrates the knowledge of each entity as the sequence of its ancestors from a domain-specific ontology. Concretely, the authors implement an RNN with LSTM cells using word embeddings of each concept extracted from WordNet (Fellbaum, 1998) and Chemical Entities of Biological Interest (ChEBI) (Degtyarenko et al., 2008).

The architecture is very similar to the SDP-LSTM model of (Xu et al., 2015), where the information of the SDP sequence are included into multiple channels, such as word embedding, WordNet hypernyms, POS tags and the grammatical relations. Firstly, texts from DDI Corpus are tokenized and parsed with Spacy library (Explosion AI, 2017). Then, the dependency parser gives the SDP of the sentence, and a drug blinding process replaces the target entities by generic strings. The WordNet hypernym is extracted from each token in the SDP with the tool developed in (Ciaramita and Altun, 2006). Moreover, some rules filter out the most common negative instances, such as candidate entities with the same name, candidates that are enumerated in a list with punctuations between them, and the case where both entities have anti-positive governors as in (Chowdhury and Lavelli, 2013*b*).

Figure 3.22 shows the different processes followed to build the system from the SDP sequence. Figure 3.22 (A) takes the words embedding of 200 dimensions and Figure 3.22 (B) transforms the WordNet representation of each token into a 50 dimensional vector. Two representations for the ChEBI ancestors are considered, and their concepts from the ontology are embedded to a 50-dimensional vector. Figure 3.22 (C) represents the concatenation of the ancestors for the two target entity concepts and Figure 3.22 (D) takes the common ancestors of the two target entity concepts. All of them use an RNN with LSTM cell of 200 dimensions and a max-pooling layer to encode the vector representation of the DDI sentence. After that, a dense layer of 50 units and a Softmax layer performs the final classification.

The experimental results demonstrate that the addition of ChEBI ancestor information and WordNet hypernyms increases the performance for the DDIExtraction Task obtaining a 57.49% in F-measure against 55.42% taking only the word embeddings. However, these results are lower

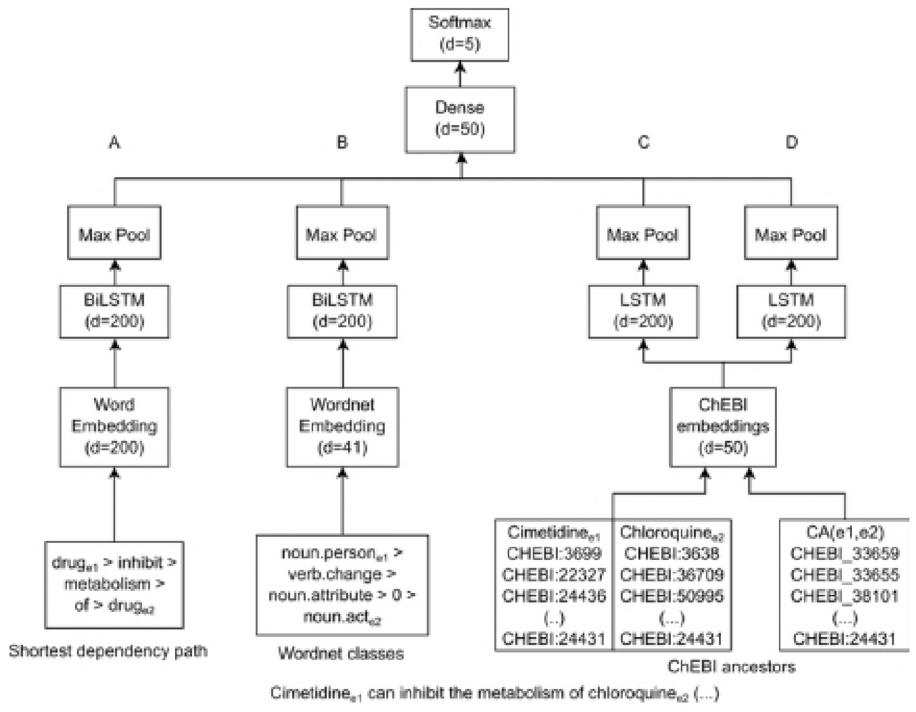

Softmax (d=5)
Dense (d=50)
A
B
C
D
Max Pool
BiLSTM (d=200)
LSTM (d=200)
Word Embedding (d=200)
Wordnet Embedding (d=41)
ChEBI embeddings (d=50)
drug$_{e1}$ > inhibit > metabolism > of > drug$_{e2}$
Shortest dependency path
noun.person$_{e1}$ > verb.change > noun.attribute > 0 > noun.act$_{e2}$
Wordnet classes
Cimetidine$_{e1}$ CHEBI:3699 CHEBI:22327 CHEBI:24436 (..) CHEBI:24431
Chloroquine$_{e2}$ CHEBI:3638 CHEBI:36709 CHEBI:50995 (...) CHEBI:24431
CA(e1,e2) CHEBI_33659 CHEBI_33655 CHEBI_38101 (...) CHEBI:24431
ChEBI ancestors
Cimetidine$_{e1}$ can inhibit the metabolism of chloroquine$_{e2}$ (...)

Figure 3.22: The systems proposed for the extraction of DDIs using BO-LSTM. Images extracted from (Lamurias et al., 2018).

than the state-of-the-art techniques that use an RNN.

The authors in (Xu et al., 2018) propose to add a Biomedical Resource Concept embeddings into the representation of each word. Similarly to the model proposed in (De Vine et al., 2014), a biomedical resource information is mapped into embeddings using the Skip-gram model with the collections of OHSUMED (MedLine abstracts) (Hersh et al., 1994) and MedTrack (clinical patient records) (Voorhees, 2013), which are converted to UMLS concept sequence with Metamap (Aronson, 2001) (see Figure 3.23a).

The preprocessing step includes negative instance filtering, drug blinding and tokenization with Genia tagger (Tsuruoka et al., 2005). Word embeddings pre-trained using Word2vec with PubMed and PMC, and position embeddings are taken together with the concept embedding to create the input of a bidirectional RNN with LSTM cells. The two direction outputs are the vector representation of the sentences and are classified by a Softmax layer to generate a prediction (see Figure 3.23b).

The model reaches 71.15% in F-measure, which is a similar performance in comparison with the state-of-the-art systems being a simple approach. Experimental results suggest that taking

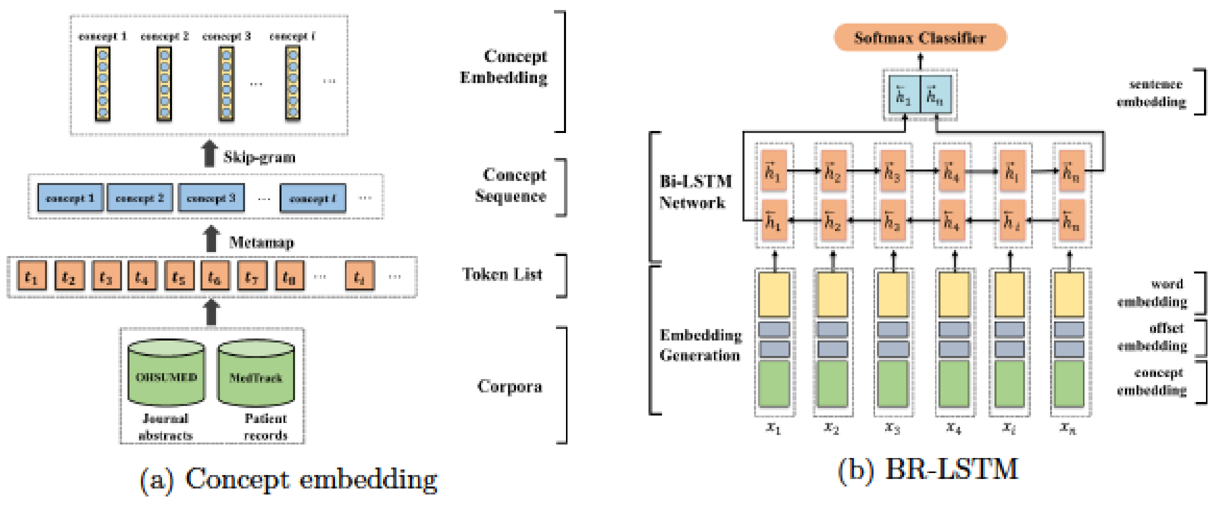


Figure 3.23: The process of extracting the concept embeddings and the architecture of the BR-LSTM. Images extracted from (Xu et al., 2018).

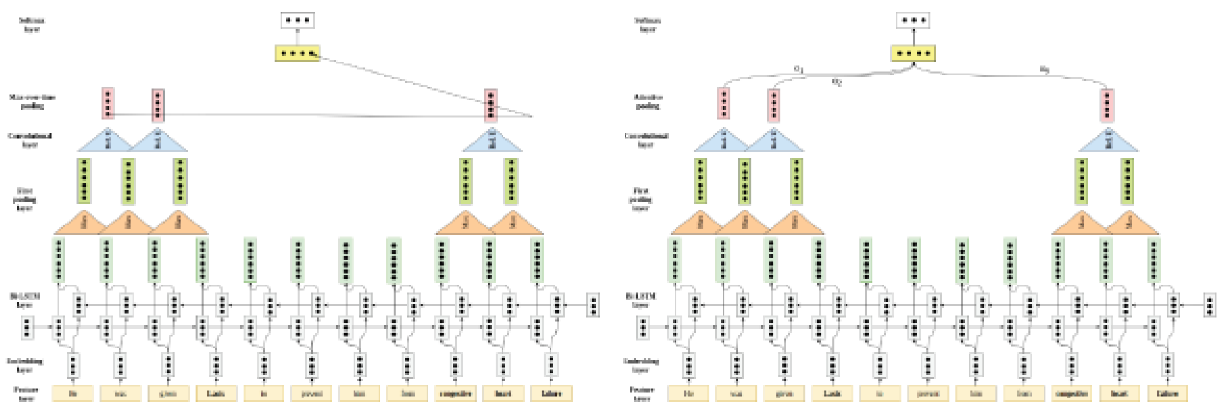
static concept embeddings can decrease 2.45% points in F-measure against taking dynamic concept embeddings, but the optimization speed is faster due to the fact that these vectors have not to be updated.

3.3.4 Hybrid Neural Network systems

The paper of (Raj et al., 2017) introduces the idea of mixing two layers of CNN and RNN in sequence (CRNN) for the DDIExtraction Task. This architecture is created to extract local and global contexts of the sentences with CNN and RNN, respectively. Following their previous work (Sahu and Anand, 2018), two kinds of pooling layer are tested: the max-pooling (CRNN-Max) and the attentive pooling (CRNN-Att).

Firstly, a bidirectional RNN with LSTM cells is applied to the vector representation of each word in the sentence. Then, a max-pooling reduces some output vectors of the bidirectional LSTM according to a window size. These vectors are the input to a CNN, and the outputs are fed to a max-pooling or an attentive pooling. Finally, a Softmax layer classifies each resulting vector into a DDI class (see Figure 3.24).

The preprocessing is similar to the work of (Sahu and Anand, 2018), including drug blinding, replacing numbers with the token *"NUM"* and a negative instance filtering. Moreover, the



(a) CRNN-Max (b) CRNN-Att

Figure 3.24: Architectures for the CRNN with max-pooling (CRNN-Max) and attentive pooling (CRNN-Att). Image extracted from (Raj et al., 2017).

authors train a 100-dimensional word embedding model.

The hybrid system with max-pooling, CRNN-Max, obtains 65.89% in F-measure being better than the attentive pooling, which obtains 63.24%. Furthermore, the authors try different initialization and update of the word embedding concluding that the random initialization and non-trainable embeddings are beneficial for CRNN-Max while the PubMed initialization and trainable embeddings are beneficial for CRNN-Att. The models show high performance without defining position embeddings which can boost their results.

SVM models have shown good results for the detection and classification in the DDIExtraction Task (Chowdhury and Lavelli, 2013*b*; Kim et al., 2015; Raihani and Laachfoubi, 2016, 2017). Currently, Deep Learning systems obtain the state-of-the-art performance for the DDI Corpus, but few of them perform a two-stage method where two classifiers for the detection and the classification as in (Zhao et al., 2016). The system proposed in (Huang et al., 2017) is the first work that uses a classical algorithm such as SVM for the detection of DDIs in sentences and a Deep Learning system such as RNN for the classification of the detected DDIs. Two-stage methods may outperform the multi-class classifiers in the one-stage method because the DDI types are very imbalanced, mostly for the negative class.

The set of features defines for the detection stage with the SVM includes the left and right three words of the two target drugs, three patterns that indicate the trigger of the relation and the unigrams and bigrams from the verb chunk, the syntactic structure of the DDI. Moreover, some auxiliary features are defined to indicate the real names or pronouns of the target drugs, if their names are the same and if they are in the same chunk.

A multi-class RNN with LSTM cells classifies the instances detected by the SVM. Firstly, the model generates the word embeddings, the position embeddings, stem embeddings, POS tag embeddings, chunk embedding and entity embeddings extracting these features with the model of (Jiang et al., 2015) for each word, and the dependency parse tree of the sentence using GDEP (Sagae and Tsujii, 2007), which is trained on the Genia Treebank (Kim et al., 2003). Concretely, a non-linear function transforms the relative distances to the two interacting drugs in each word into the position embeddings as $s(d) = tanh(d/A)$ where d is the distance to the interacting drug, and A is the average number of tokens in all the DDI sentences. All the embeddings are concatenated and are the input to the bidirectional RNN with LSTM cells.

Figure 3.25 shows the entire system for the classification of DDI instances, where the blocks 'A' represent three components: a maxout layer, a highway layer and a bidirectional LSTM layer. Before the block 'A', a dropout layer randomly drops some units of the input. After the block 'A', a mean-pooling layer is applied to all the outputs. Finally, a Softmax layer classifies this vector into one of the DDI categories. Besides, this work also explores a deeper system adding more than one layer with multiple 'A' blocks.

The authors conclude that the bidirectional LSTM is better than a single LSTM. Besides, the embeddings of stem, chunk and entity do not improve the results, while the embeddings of the word, position and POS tags are enough to have a good performance. Moreover, the two-stage method is better than the Random over-sampling to generate duplicates of the least representative class, i.e. the "Int" class. Furthermore, the architecture with the two layers has a higher performance than one layer or even three layers. Thus, the best performance of 60% in F-measure is obtained with the two-stage method of two bidirectional RNN layers with LSTM cells, which takes the embeddings of the word, relative distances to the two interacting drugs and the POS tags.

This work is the first one in exploring more than one layer and the combination of SVM with RNN. However, the system only reports F-measure with macro-average and cannot be compared

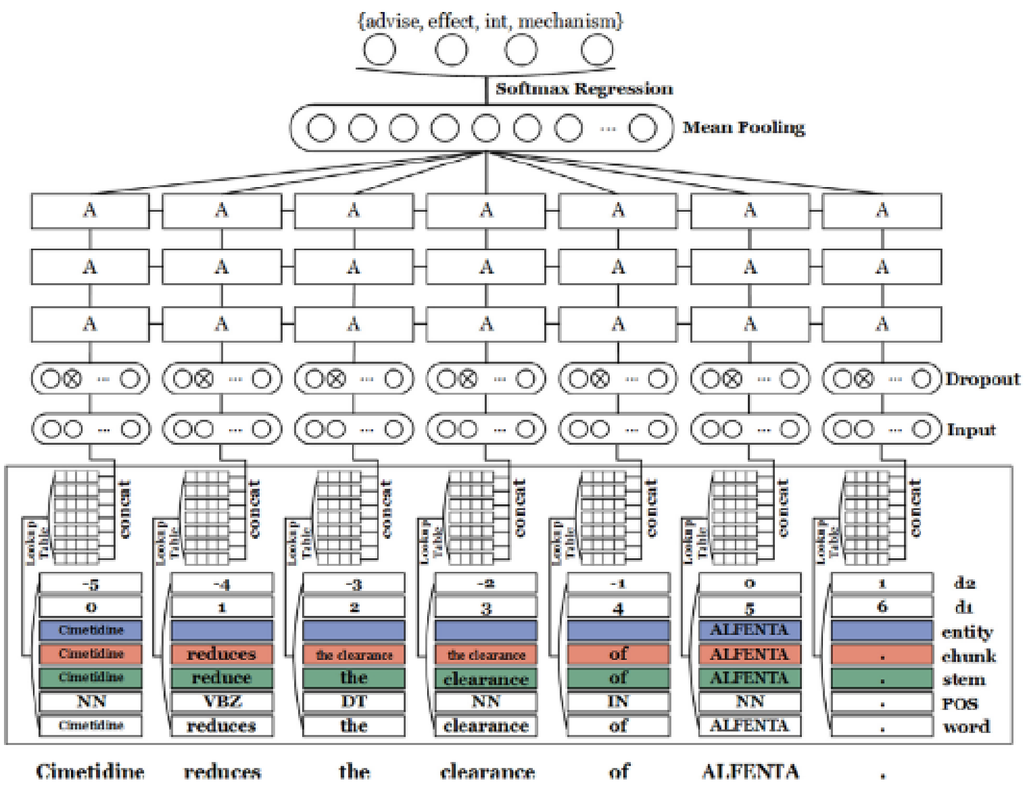


Figure 3.25: The model for classification of DDI sentences where 'A' represents a block with maxout, highway and bidirectional LSTM layers. Image extracted from (Huang et al., 2017).

with the state-of-the-art techniques because they applied the F-measure with micro-average, which is the official measure in the DDIExtraction Task 2013.

The system of (Zhang, Lin, Yang, Wang, Zhang, Sun and Yang, 2018) proposes a hybrid model that combines RNN and CNN with the word sequence and the SDP sequence of the DDI sentences as inputs. Contrary to (Raj et al., 2017) the model performs the combination of RNN and CNN in a parallel manner. Apart from the dependency graph of each word, the Stanford Parser gets the dependency relations of each edge of the graph which is not frequently used for the neural models. This information is added to the model to obtain a better understanding of the syntactic part of the sentence.

Figure 3.26 illustrates three parallel networks defined for building the architecture. Firstly, a bidirectional RNN with LSTM cells that takes the word embeddings and the position embeddings of the sentence as input. Secondly, a CNN with the word embeddings and positions embeddings of the SDP sequence of the sentence. Thirdly, a CNN whose input is the dependency relations of the SDP represented by a dependency relation embeddings. In both representations, a max-pooling layer reduces the matrices into the output vectors. Furthermore, the concatenation of the two directional RNN outputs and the two max-pooling vectors are applied to two different single NN. Thus, the final classifier layer computes the Softmax operation of the two concatenated NN outputs.

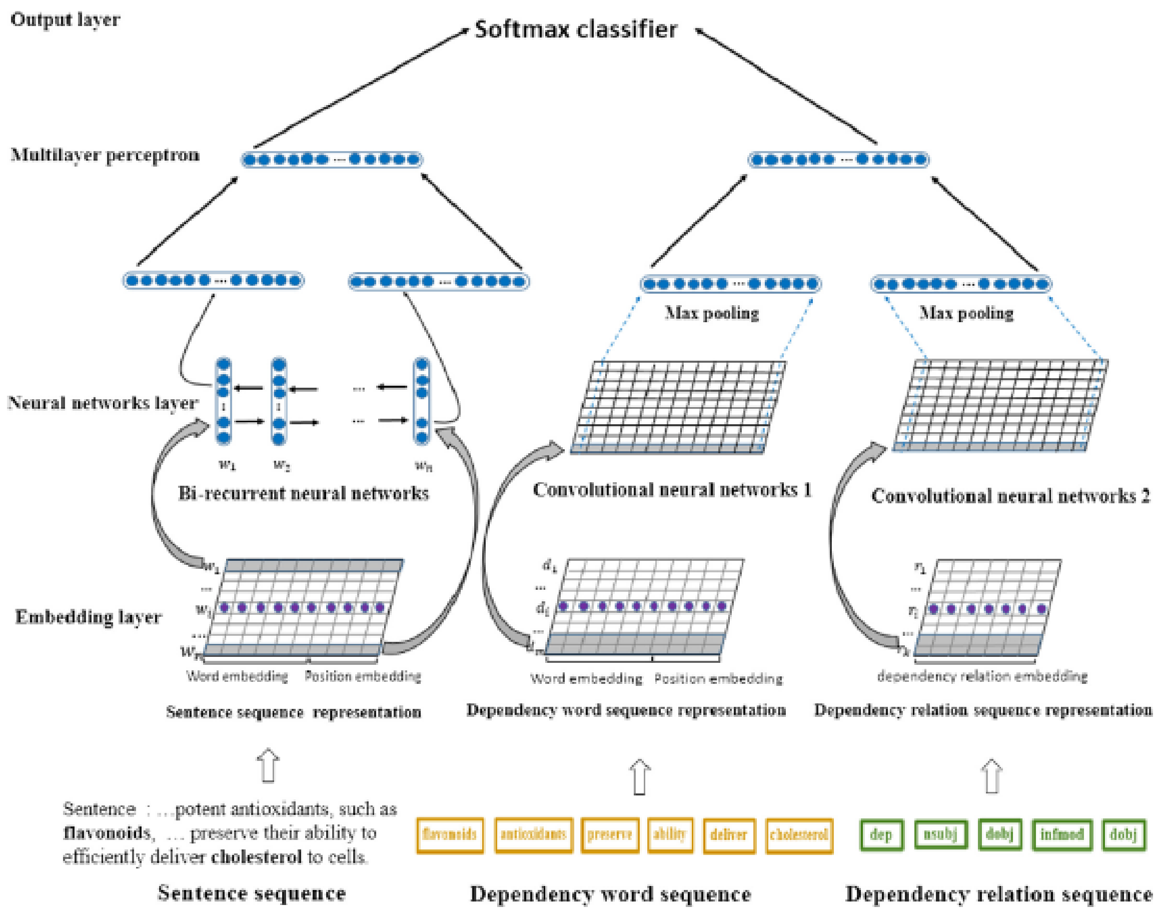
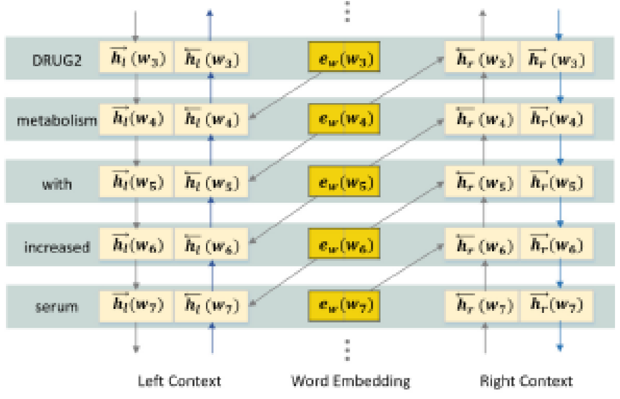

Output layer
Softmax classifier
Multilayer perceptron
Max pooling
Max pooling
Neural networks layer
Bi-recurrent neural networks
Convolutional neural networks 1
Convolutional neural networks 2
Embedding layer
Word embedding
Position embedding
Word embedding
Position embedding
dependency relation embedding
Sentence sequence representation
Dependency word sequence representation
Dependency relation sequence representation
Sentence : …potent antioxidants, such as flavonoids, … preserve their ability to efficiently deliver cholesterol to cells.
flavonoids
antioxidants
preserve
ability
deliver
cholesterol
dep
nsubj
dobj
infmod
dobj
Sentence sequence
Dependency word sequence
Dependency relation sequence

Figure 3.26: The overview of the system with the sentence, the dependency word and relation sequences from the SDP. Image extracted from (Zhang, Lin, Yang, Wang, Zhang, Sun and Yang, 2018).

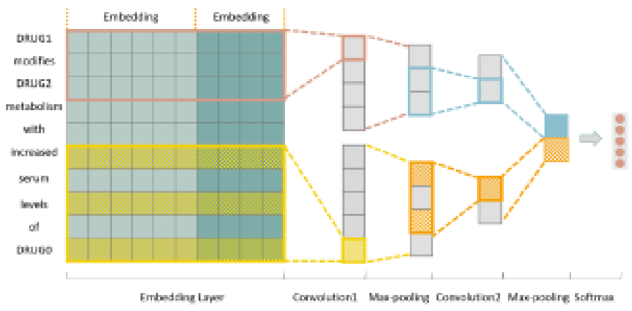
The hybrid model reaches 75.1% in F-measure suggesting that the SDP and their dependency relationships are a valuable source of information in order to classify the DDI instances.

The Recurrent Hybrid Convolutional Neural Network (RHCNN) proposed in (Sun et al., 2019) is a combination of an RNN, which captures the semantic embedding of each word from its context, and two CNN layers, which builds the sentence-level feature in order to perform the classification. This method is the first system that applies dilated convolutions for the DDI Corpus. Additionally, the cross-entropy loss function is replaced by the focal loss function that is more suitable for imbalanced datasets.

Firstly, the entities in the sentences are transformed using the drug blinding technique, and some negative instances are filtered out with basic manually-designed rules. Two bidirectional RNNs with LSTM cells capture the semantic information of the word in the dataset according to their left context and right context (see Figure 3.27a). To do this, the pre-trained word embeddings from (Pyysalo et al., 2013) are used in this architecture. The word embeddings and the embeddings extracted from both contexts are concatenated and fed into a single Neural Network. After that, a CNN applies the convolution and the dilated operations to the output vector of each word concatenated with its position embedding. In this model, max-pooling performs the dimensionality reduction of the resulting matrix from the convolutions filters. Then, another CNN with max-pooling creates a more abstractive sentence representation (see Figure 3.27a). Finally, a Softmax layer classifies each vector representation in one of the DDI types. Another contribution in this work is the use of the focal loss function (Lin et al., 2017) for dealing with the imbalanced DDI Corpus.



(a) Two bidirectional RNNs with LSTM cells for the left context and the right context of each word.



(b) CNN with convolution and dilated operations to generate the sentence representation.

Figure 3.27: The architecture of the system RHCNN. Image extracted from (Sun et al., 2019).

The RHCNN results for the DDIExtraction Task reaches to 75.48% in F-measure being the state-of-the-art technique for hybrid systems. The method includes a novel extraction of the word embedding with an RNN applied over the context of each word, and the use of a CNN combined with convolution and dilated operations. Furthermore, the work shows that the definition of a loss function for the imbalanced datasets improves the results on the DDI Corpus. These techniques boost the performance for the classifications of DDI sentences without using external tools and should be explored for future researches.

3.4 Conclusions

This chapter presents the state-of-the-art techniques for the tasks of DrugNER and DDIExtraction. Some conclusions can be extracted analysing the performance of the systems over time. All the participants of the DDIExtraction Shared Task applied supervised machine learning with manually generated features. The best results of the DrugNER Task participants was the WBI system (Rocktäschel et al., 2013) (Strict F-measure = 71.5%), which uses a CRF with features generated from the output of ChemSpot (Rocktäschel et al., 2012), an existing chemical named entity recognition tool, as well as a collection of domain-specific resources. The best results of the DDIExtraction Task participants was the FBK-irst system (Chowdhury and Lavelli, 2013*b*) (Micro-average F-measure = 65.1%), which designs a two-stage classification system using two SVMs based on a hybrid kernel composed by a feature-based kernel, shallow linguistic kernel and the Path-enclosed Tree kernel.

In general, the results on the DDI-DrugBank dataset are better than those obtained on the DDI-MedLine dataset for both subtasks. The main reason is that the MedLine dataset has a lower size and greater complexity in their texts than DrugBank dataset. Another reason is that DrugBank texts involve descriptions of drugs and their interactions, while the main topic of MedLine texts does not necessarily involve DDIs.

Currently, Deep Learning models have been successfully applied for the DrugNER and DDIExtraction tasks obtaining the best performance with 79.36% (Unanue et al., 2017) and 86.27% (Dewi et al., 2017) in F-measure, respectively. These methods not only reach the classical machine learning methods, but they outperform them. Besides, Deep Learning techniques automatically define the features to be used, and no expert knowledge in a specific domain is required. The architectures usually only take the word embeddings of the words as input to generate a prediction of the sentences. One of the main conclusions is that the pre-trained word embedding with specific domain collections performs better than the randomly initialized embeddings in most of the related works.

Table 3.2, Table 3.3, Table 3.4 and Table 3.5 show the compilation of all the models that were built for the DDIExtraction Task using Feature Engineering, Convolutional Neural Network, Recurrent Neural Network and Hybrid Neural Network, respectively. Entity blinding, negative instance filtering and position embeddings are commonly adopted for the best DDI systems because it has been tested that increase the results for Relation Extraction architectures. Moreover, the models that perform the detection and the classification in two-stages have better performance than the one-stage models. Generally, the two-stage systems in the first step filter the negative instances against all the positive instances being more balanced in the number of samples and reducing the number of FP and FN for the second step. On the contrary, most of Deep Learning models presented perform the classification of DDI sentences in one-stage because they must be able to create the right features for its detection and classification. It is not clear which kind of Deep Neural Network is better, but the best system on DDIExtraction Task (Dewi et al., 2017) shows that having deeper architectures of neural models increases their performance. Furthermore, the combination of hybrid neural models and the ensemble of systems usually outperform the basic configurations. Concretely, the models that aggregate different linguistic information as embeddings improve their performance.

Table 3.2: Feature Engineering systems results for the DDIExtraction task.

Systems	Negative instance filtering	Machine Learning technique	Kernel	Stages	Ensemble	External resources and tools	F-measure
Rich feature-based kernel (Raihani and Laachfoubi, 2017)	✓	SVM (*Int-vs-all* and *one-vs-one* in cascade)	RBF kernel with a rich feature set	2 stages	✗	LingPipe NLP, OpenNLP, Stanford Parser, LIBSVM	**71.79%**
EDDIDS (Raihani and Laachfoubi, 2016)	✓	SVM (*Int-vs-all* and *one-vs-one* in cascade)	RBF kernel	2 stages	✗	LingPipe NLP, OpenNLP, Stanford Parser, LIBSVM	**71.14%** (System_2)
		SVM (*one-vs-one* in cascade)					**70.61%** (System_1)
		SVM (*one-vs-all*)					**70.19%** (System_3)
Context Vector (Zheng et al., 2016)	✓	SVM (*one-vs-all*)	Graph kernel	2 stages	✗	Stanford parser	**68.4%**
Linear SVM with word and syntactic features (Kim et al., 2015)	✓	SVM (*one-vs-one*)	Linear	2 stages	✗	BioLemmatizer, McClosky's biomodel, Stanford parser	**67%**
Corpus-level features (Tu et al., 2017)	✗	SVM (*one-vs-all*)	RBF kernel	1 stage	Voting strategy of 5 models	RxNorm tool, SParseval, LIBSVM	**65.3%**
FBK-irst (Chowdhury and Lavelli, 2013*b*)	✓	SVM (*one-vs-all*)	Hybrid Kernel (Feature-based kernel, Path-enclosed Tree (PET) kernel, Shallow Linguistic Kernel)	2 stages	✗	Charniak-Johnson parser, McClosky's biomodel, Stanford parser, BioEnEx (NER for diseases), SVM-Light-TK, jSRE	**65.1%**

Table 3.3: Convolutional Neural Network systems results for the DDIExtraction task.

Systems	Negative instance filtering	Word embedding	Position embedding	Other embeddings	Stages	Ensemble	Attention	External resources and tools	F-measure
Multichannel + 10 CNN layers (Dewi et al., 2017)	✓	200-dimensional using Word2vec with MedLine and 4 word embeddings from (Pyysalo et al., 2013)	✗	✗	1 stage	10 CNN layers in sequence with Multichannel	✗	Keras	**86.27%**
DDNet with 16 CNN layers (Sun et al., 2018)	✗	300-dimensional using Glove	✗	✗	1 stage	16 CNN layers in sequence	✗	Keras	**84.5%**
CNN TEES (Björne and Salakoski, 2018)	✗	200-dimensional PubMed and Wikipedia from (Pyysalo et al., 2013)	8-dimensional	8-dimensional of POS tag, entity type, relative position, path, SDP and event argument embeddings	4 stages	Average of 5 models mixing the set split	✗	Charniak-Johnson parser, McClosky's biomodel, Stanford parser, TEES, Keras	**73.51%** (mixed 5 ensemble)
						Average of 5 models			**70.64%** (5 ensemble)
						Best of 5 models			**70.51%** (best system)
GCN methods (Asada et al., 2018)	✗	200-dimensional using Word2vec with MedLine/PubMed	20-dimensional	50-dimensional molecular structure with GGNN	1 stage	✗	✗	SMILES, RDKit, Genia tagger	**72.55%** (GGNN)
				50-dimensional molecular structure with NFP					**72.21%** (NFP)
CNN+DCNN Combined (Liu, Chen, Chen and Tang, 2016)	✓	300-dimensional using Order algorithm with MedLine	10-dimensional	✗	1 stage	CNN and DCNN	✗	NLTK, Charniak-Johnson parser, Stanford parser	**70.81%**
MCCNN (Quan et al., 2016)	✓	200-dimensional using Word2vec with MedLine and 4 word embeddings from (Pyysalo et al., 2013)	✗	✗	1 stage	Multichannel	✗	Keras	**70.21%** (with filtering)
	✗								**67.8%** (without filtering)
DCNN (Liu, Chen, Chen and Tang, 2016)	✓	300-dimensional using Order algorithm with MedLine	10-dimensional	✗	1 stage	✗	✗	NLTK, Charniak-Johnson parser, Stanford parser	**70.19%**
CNN (Liu, Tang, Chen and Wang, 2016)	✓	300-dimensional using Order algorithm with MedLine	10-dimensional	✗	1 stage	✗	✗	NLTK	**69.75%**
Attention CNN (Asada et al., 2017)	✗	200-dimensional using Word2vec with MedLine/PubMed	20-dimensional	✗	1 stage	✗	Input attention mechanism	Genia tagger	**69.12%**
SCNN (Zhao et al., 2016)	✓	50-dimensional using Word2vec with MedLine	10 bit binary vector	8-dimensional POS tag embedding	2 stages	✗	✗	Enju parser, MetaMap, biomedical tokenizer (Jiang and Zhai, 2007)	**68.6%** (SCNN^2)
					1 stages				**67%** (SCNN^1)

Table 3.4: Recurrent Neural Network systems results for the DDIExtraction task.

Systems	Negative instance filtering	Word embedding	Position embedding	Other embeddings	Stages	Ensemble	Attention	External resources and tools	F-measure
Attention LSTM (Zheng et al., 2017)	✓	200-dimensional using Word2vec with PubMed	10-dimensional	10-dimensional POS tag embeddings	1 stage	✗	Input attention mechanism	Stanford parser, Keras	77.3%
TreeLSTM (Lim et al., 2018)	✓	200-dimensional PubMed and PMC from (Pyysalo et al., 2013)	10 bit binary vector	10-dimensional subtree containment embeddings	1 stage	Sum the weigth results of 10 models	✗	BEST, Stanford parser, Tensorflow Fold	73.5% (One-stage (ensemble))
					2 stages	Sum the weigth results of 10 models			72.7% (Two-stage (ensemble))
					1 stage	✗			71.7% (One-stage (single))
					2 stages	✗			71.4% (Two-stage (single))
PM-BLST (Zhou et al., 2018)	✓	300-dimensional using Word2vec with PubMed	10-dimensional	✗	Multitask	✗	Position-aware attention	Keras	72.99% (with filtering)
	✗								71.61% (without filtering)
Hierarchical RNN (Zhang, Zheng, Lin, Wang, Yang and Dumontier, 2018)	✗	200-dimensional using Word2vec with PubMed	10-dimensional	10-dimensional POS tag embeddings using Word2vec with PubMed	1 stage	Two RNN in sequence	Input attention mechanism of the sentence sequence and the SDP sequence	Stanford parser, Keras	72.9% (with SDP)
							Input attention mechanism of the sentence sequence		71.7% (without SDP)
GRU + 2ATT (Yi et al., 2017)	✗	100-dimensional using Glove	10-dimensional	✗	1 stage	✗	Attentive pooling and sentence level attention	Tensorflow	72.2% (paper results) 67.58% (contingency matrix results)
WordRNN + CharRNN (Kavuluru et al., 2017)	✓	250-dimensional	32-dimensional	250-dimensional character embedding	1 stage	Bootstrapped model average of 10,000 character and word levels in sequence	✗	Theano, Tensorflow Fold	72.13% (with filtering)
	✗								70.81% (without filtering)
DLSTM (Wang et al., 2017)	✓	100-dimensional using Word2vec with MedLine	10 bit binary vector	✗	1 stage	Multichannel	✗	Stanford parser, NLTK, Tensorflow	72% (syntax embedding)
		100-dimensional							71.37% (random embedding)
Joint AB-LSTM (Sahu and Anand, 2018)	✓	100-dimensional using Glove with PubMed	10-dimensional	✗	1 stage	B-LSTM and AB-LSTM	Attentive pooling	Genia tagger, Tensorflow	71.48% (with filtering)
	✗								69.27% (without filtering)
Skeleton LSTM (Jiang et al., 2017)	✗	pre-trained with Word2vec	✓	✗	1 stage	✗	✗		71.4%
BR-LSTM (Xu et al., 2018)	✓	200-dimensional using Word2vec with PubMed, PMC and Wikipedia	10-dimensional	200-dimensional biomedical resource concept embeddings using Word2vec with OHSUMED and MedTrack	1 stage	✗	✗	Genia tagger, MetaMap, Keras	71.15%
Chain based RNN (Ebrahimi and Dou, 2015)	✗	50-dimensional from (Collobert and Weston, 2008)	✗	✗	1 stage	✗	✗	MaltParser, Supersense tagger	68.64%
BO-LSTM (Lamurias et al., 2018)	✓	200-dimensional PubMed and PMC from (Pyysalo et al., 2013)	✗	50-dimensional of ChEBI and WordNet embeddings	1 stage	✗	✗	Spacy, ChEBI ontology WordNet hypernym Keras	57.49%

Table 3.5: Hybrid Neural Network systems results for the DDIExtraction task.

Systems	Negative instance filtering	Word embedding	Position embedding	Other embeddings	Stages	Ensemble	Attention	External resources and tools	F-measure
RHCNN (Sun et al., 2019)	✓	200-dimensional PubMed and Wikipedia from (Pyysalo et al., 2013)	15-dimensional	200-dimensional of right-context and left-context embeddings	1 stage	Word and semantic levels in sequence	✗	Keras	75.48 %
Hybrid CNN+RNN (Zhang, Lin, Yang, Wang, Zhang, Sun and Yang, 2018)	✗	200-dimensional using Word2vec with PMC	50-dimensional	100-dimensional dependency relation embeddings	1 stage	RNN and two CNN	Input attention mechanism	Stanford parser	75.1 %
SVM+2BLSTM (Huang et al., 2017)	✗	pre-trained from (Jiang et al., 2015)	non-linear transformation	POS tag embeddings	2 stages	SVM and two RNN in sequence	✗	GDEP parser, Genia tagger	69 % (Macro-average)
CRNN (Raj et al., 2017)	✓	100-dimensional using Glove with PubMed	✗	✗	1 stage	RNN and CNN in sequence	✗	Genia tagger, Tensorflow	65.89 % (CRNN-Max)
							Attentive pooling		63.24 % (CRNN-Att)

Chapter 4

Biomedical Named Recognition using Word Embeddings

This chapter describes a machine learning-based approach that uses word embedding as features to recognize named entities from biomedical texts. As a starting point, a baseline system is developed based on CRF trained with standard features used in NER systems (Lafferty et al., 2001; Li and McCallum, 2003; Sha and Pereira, 2003). Then, the system was extended to incorporate new features, such as word vectors and word clusters generated by the Word2vec tool. Additionally, a lexicon feature from the DINTO ontology[1] (Herrero-Zazo et al., 2015) or from the DNorm tool (Leaman et al., 2013) is aggregated depending on the task. The Word2vec tool is trained over two different corpora: Wikipedia and MedLine. This section studies the effectiveness of using word embeddings as features to improve performance on the baseline system, as well as to analyse whether the described lexicon features could be a valuable complementary data source integrated into a machine learning NER system proposed in (Segura-Bedmar et al., 2015*a*; Segura-Bedmar et al., 2015*b*; Suárez-Paniagua et al., 2015). To evaluate this approach and compare it with previous works, a series of experiments are conducted on the DDI Corpus (Herrero-Zazo et al., 2013), the Chemical Entity Mention in Patents (CEMP) subtask of the BioCreative V Chemical Compound and Drug Name Recognition (CHEMDNER) patents task

[1]http://www.obofoundry.org/cgi-bin/detail.cgi?id=DINTO

(Krallinger, Leitner, Rabal, Vazquez, Oyarzabal and Valencia, 2015) and the Disease Named Entity Recognition and Normalization subtask of the BioCreative V Chemical-Disease Relation (CDR) task (Wei et al., 2015).

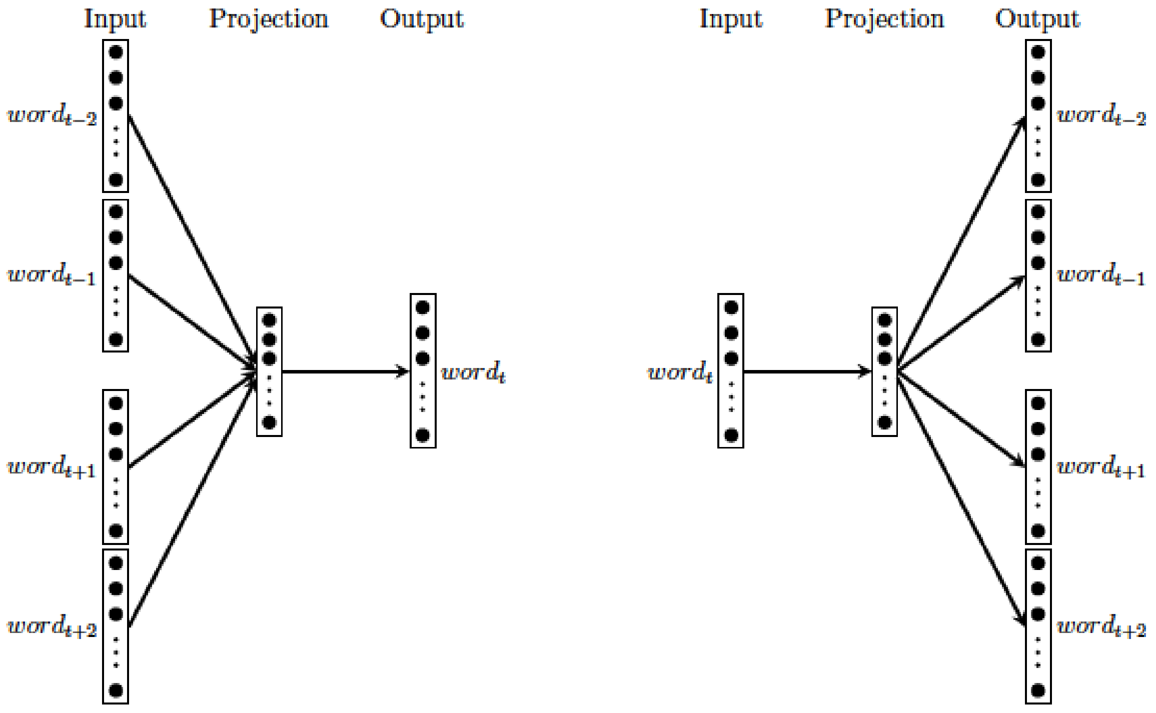
4.1 Introduction

Analogue audio and image data can be digitalized and processed by computer from pixel intensities and voltage changes, respectively. Contrary to these processing systems, natural language and more concretely words do not have high-dimensional representation encoded into a vector. Typically, the words are represented as an index in large vocabulary, but this representation lacks information about the relationships between words.

The basic representation of words are the one-hot-encoding vectors. These are vectors with a length of the vocabulary filled with zeros except in the index of the word that fires which is one. This discrete representation of words makes vector space with high dimensionality and very sparse suffering the curse of dimensionality (Bellman et al., 1957), i.e. statistical models need a vast amount of samples that represent the different values of all the dimensions in the dataset for training (Trunk, 1979). For this reason, it is desirable to have a dense vector representation with a low dimensionality size for each word that belongs to a concrete meaning in the semantic space (Bengio et al., 2003).

Word embedding is the representation of words into a real value vector with respect to a semantic space. These vector space models are built according to the context in where each word appears based on the idea of distributional hypothesis (Harris, 1954*a*) which states that lexical items with similar distributions have similar meanings.

Taking this idea, the Word2vec model (Mikolov, Chen, Corrado and Dean, 2013) was the first system that uses a neural network to encode the meaning of a word into a vector (auto-encoder) according to the context it appears. Two architectures were proposed to produce a distributed representation of words in Word2vec model. The continuous bag-of-words (CBOW) (Mikolov, Chen, Corrado and Dean, 2013) takes a window of the surrounding words and uses the word as the label to learn its projection in the space (see Figure 4.1a). The continuous Skip-gram (Mikolov, Sutskever, Chen, Corrado and Dean, 2013) takes the word and uses the window of the surrounding words as the label to learn its projection in the space (see Figure 4.1b). The main difference is that CBOW is faster than skip-gram, but skip-gram learns a better representation of infrequent words than CBOW.



(a) CBOW (b) Skip-gram

Figure 4.1: Word2vec architectures for word representation.

These distributed vectors capture the semantic meaning of the words avoiding to provide this information with an external resource or tool. A property of these embeddings is that they preserve the distance similarities between words with the same semantic relationships in the space. For instance, Figure 4.2 shows that subtracting to vector(*'King'*) the male meaning of vector(*'Man'*) and adding the female meaning of vector(*'Woman'*) results in a position in the space close to vector(*'Queen'*) (Mikolov, Yih and Zweig, 2013).

Named Entity Recognition (NER) is a crucial component for many Natural Language Processing (NLP) systems such as relation extraction, text classification or sentiment analysis systems, among many others. The automatic recognition of biomedical entities from scientific texts can markedly reduce the time that experts spend populating biomedical knowledge bases as well as

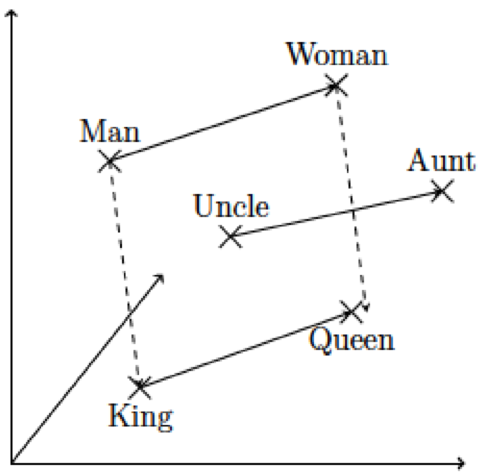


Figure 4.2: Word embeddings of masculine-feminine pairs.

annotating papers and patents. Conditional Random Fields (CRF) (Sutton et al., 2012) often show best results in the recognition of drugs and chemical names (Krallinger, Leitner, Rabal, Vazquez, Oyarzabal and Valencia, 2015; Segura-Bedmar et al., 2013). So far the most popular features for CRF-based NER systems concern syntactic and semantic properties of words (such as tokens, POS tags, lemmas, orthographic and lexicon features, among others). To this end, a system based on a CRF to recognize drug, chemical and disease mentions occurring in the biomedical texts is developed.

One of the goals of this chapter is to study whether the lexicon features from DINTO ontology and DNorm tool can provide a valuable source of information for these tasks. Moreover, this lexicon binary feature indicates whether the current token was found in a gazetteer of diseases provided by the DINTO ontology or DNorm tool. As far as we know, DINTO is the first ontology providing a comprehensive and accurate representation of drug-drug interactions knowledge. The DINTO ontology contains a total of 25,809 classes, in particular, 8,786 drugs and 11,555 DDIs. Several domain resources such as the CheBI ontology (Degtyarenko et al., 2008), the DrugBank database (Wishart et al., 2006) or the OAE ontology (He et al., 2014) have been reused to create DINTO. Furthermore, it was designed to be used by the computer science community working on the DDI domain. (Herrero Zazo, 2015) describes the DINTO ontology in detail.

As the main contribution, this chapter explores the effectiveness of new features for the biomedical NER tasks, concretely, word vectors and word clusters generated using the Word2vec tool (Mikolov, Chen, Corrado and Dean, 2013). The main hypothesis is that the word embedding features can represent the semantic information of the biomedical mentions into high-dimensional vectors in order to accurately detect them. Word embeddings have shown promising results in NLP tasks, such as named entity recognition, sentiment analysis or parsing (Socher, Bauer, Manning and Andrew Y., 2013; Socher, Perelygin, Wu, Chuang, Manning, Ng and Potts, 2013; Turian et al., 2010). The essential assumption of word embeddings is that semantically nearby words have similar vectors (Bengio et al., 2003).

In contrast to (Liu, Tang, Chen, Wang and Fan, 2015), the proposed model creates the word embedding features (word clusters and word vectors) using the latest Wikipedia dump[2], which contains more than 3 billion words, as well as the 2013 release of MedLine[3], which they used for generating their word representations. This release contains approximately one million words, being thus much smaller than the Wikipedia collection. While MedLine is a biomedical literature database, Wikipedia covers many different domains of knowledge. However, the hypothesis of this chapter is that the larger the dataset used for training the Word2vec models, the better word embeddings should be obtained. Thus, the effectiveness of word embeddings features trained on a specific domain corpus, such as MedLine, can be compared to those trained on a more extensive collection, such as Wikipedia. Another key difference with (Liu, Tang, Chen, Wang and Fan, 2015) is that while the authors only gave results for the whole DDI corpus, this chapter analyses and discuss the effect of the DINTO and Word2vec features on each one of the datasets: DDI-DrugBank and DDI-MedLine. This analysis is necessary in order to know what features are more efficient on each dataset. MedLine abstracts are very different from DrugBank texts. While abstracts from DDI-MedLine are mainly addressed to scientists in life sciences, texts from DDI-DrugBank are written in a language understandable to patients.

[2]http://dumps.wikimedia.org/
[3]http://www.nlm.nih.gov/databases/journal.html

4.2 Method

This section describes the proposed system and the datasets used for the evaluation.

4.2.1 Datasets

The DDI Corpus consists of two different datasets: DDI-DrugBank (792 texts selected from the DrugBank database) and DDI-MedLine (233 MedLine abstracts on the subject of DDIs). This corpus will allow us to compare the system to the participating systems in the DrugNER Task.

The CHEMDNER corpus containing 10,000 PubMed abstracts annotated with 84,355 chemistry and chemical entity mentions was generated with 19,805 unique chemical name strings (Krallinger, Rabal, Leitner, Vazquez, Salgado, Lu, Leaman, Lu, Ji, Lowe et al., 2015). The annotators focused on the types of mentions that represent the chemical structural information from patents to annotate the chemical entities. (Krallinger, Leitner, Rabal, Vazquez, Oyarzabal and Valencia, 2015) an overview of the task and the main relevant characteristics of participating systems.

The CDR task provided a new corpus for the evaluation that contains pieces of text with disease annotations selected from CTD-Pfizer set (Davis et al., 2013). The dataset is divided into a training set and a development set, which contains 500 articles each one. However, the testing phase uses raw PubMed abstracts for the extraction of the disease mentions and their normalized MeSH identifiers.

4.2.2 Proposed system

Most successful approaches for NER have used machine learning algorithms such as CRFs trained with linguistic features (tokens, lemmas or POS tags, among others) and semantic features from domain resources such as ontologies or dictionaries. Encouraged by the good results of the CRF-based methods, a system based on CRF and also explore word embedding features provided by the Word2vec tool is proposed. In particular, a python binding[4] to CRFsuite (Okazaki, 2007) is used as the core of this machine learning technique.

CRF performs the NER task as a classification task on each token, determining whether it is an entity or not. In order to represent the class of each token, the BIO tagging scheme is taken as a reference. According to this scheme, each token is tagged as either beginning entity token (B), inside entity token (I) or outside token (O). For the detection subtask (exact criterion), three types of entities are considered: B-ENTITY, I-ENTITY and O where ENTITY represents one of the different classes.

As a first stage, a baseline system is developed using a CRF algorithm in which each token is represented with the following features:

- The context window of three tokens to its right and its left in the sentence. The context window also includes the current token.
- POS tags and lemmas in the context window are also considered.
- An orthography feature which can take the following values: upperInitial (the token begins with an uppercase letter, and the rest are lowercase), allCaps (all its letters are uppercase), lowerCase (all its letters are lowercase) and mixedCaps (the token contains any mixture of upper and lowercase letters).
- A feature representing the type of token: word, number, symbol or punctuation.

The goals are to study the contribution of the lexicon information from DINTO ontology and DNorm tool for the NER task and building a binary feature that indicated whether the current token was found or not in them.

Figure 4.3 shows a pipeline with the General Architecture for Text Engineering (GATE) components used to process the texts and to obtain the feature set used to train the CRF model.

[4]http://python-crfsuite.readthedocs.org/en/latest/

There are five main processing modules: sentence splitter, tokenizer, POS tagger, morphological analyzer and the Gate onto root gazetteer, which links text to the DINTO ontology. The ontology is processed to produce a flexible gazetteer taking into account alternative morphological forms of the instances of the ontology.

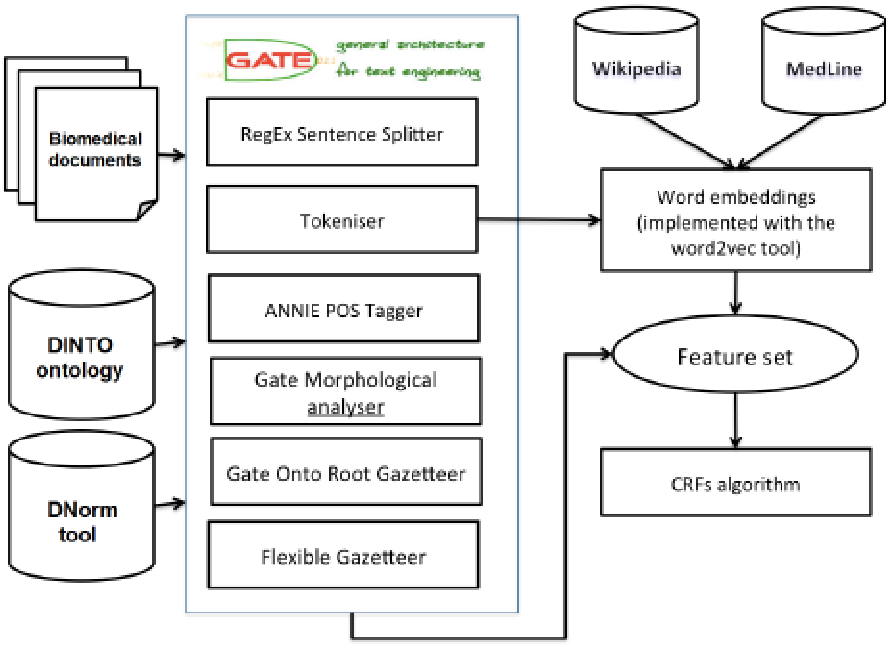


Figure 4.3: System architecture and the pipeline for NER of biomedical entities with a CRF classifier.

The main hypothesis of this chapter is that the incorporating of word embeddings as features into a CRF model could help to recognize unseen or sporadic mentions in the training set. For this reason, the word embeddings were trained using the Word2vec tool. Word2vec requires a large corpus of sentences as input dataset in order to generate word vectors by training an NN language model. The NN model can learn from the different contexts in which a word appears and then to compute its representation as a vector. This study explores the Word2vec tool trained on two different corpora. As the first option, the latest Wikipedia dump was taken, which contains more than 3 billion words. Then, the Word2vec model is trained on Wikipedia to obtain the word vectors for all tokens in the DDI corpus.

Based on the distributional hypothesis (Harris, 1954*b*), similar words will have similar vectors because they occur in similar contexts. The word vector for the current token was considered as a new feature into a CRF system with different dimensions of vectors (50, 100 and 200) (see

Table 4.2). These word representations are desirable inputs, not only for NER but also in many other NLP tasks (POS tagging, word name disambiguation, and lexical simplification, among others).

Moreover, the Word2vec tool contains a utility to compute word clusters using a k-means clustering algorithm. Thus, word clusters are included in the current token representation as a new feature for the CRF-based system. Word clusters represent words at a higher level of abstraction that may help to recognize even those mentions that are not observed in the training set. The experiments evaluate different values of k in the k-means in order to fine-tune the number of clusters.

4.3 Evaluation

4.3.1 DrugNER Task

Table 4.1 summarizes all the experiments with word vectors and word clusters trained with Wikipedia and MedLine where CRF is the baseline system and CRFD only uses the baseline with the DINTO feature.

Table 4.1: List of experiments for the DrugNER Task with different features configurations.

System	Feature set
CRF	standard feature set
CRFD	CRF + DINTO feature
CRFclusterK50Wiki	CRFD + word cluster trained with k=50 on Wikipedia
CRFclusterK50MedLine	CRFD + word cluster trained with k=50 on MedLine
CRFclusterK150Wiki	CRFD + word cluster trained with k=150 on Wikipedia
CRFclusterK150MedLine	CRFD + word cluster trained with k=150 on MedLine
CRFclusterK500Wiki	CRFD + word cluster trained with k=500 on Wikipedia
CRFclusterK50MedLine	CRFD + word cluster trained with k=500 on MedLine
CRFvec50Wiki	CRFD + word vectors of dimension 50 trained on Wikipedia
CRFvec50MedLine	CRFD + word vectors of dimension 50 trained on MedLine
CRFvec100Wiki	CRFD + word vectors of dimension 100 trained on Wikipedia
CRFvec100MedLine	CRFD + word vectors of dimension 100trained on MedLine
CRFvec200Wiki	CRFD + word vectors of dimension 200 trained on Wikipedia
CRFvec200MedLine	CRFD + word vectors of dimension 200 trained on MedLine

Table 4.2 shows the results for the different settings studied for the detection subtask (exact criterion) and the classification subtask (strict criterion). The scores correspond to the micro-average values, which were calculated with regarding all classes (B- and I-) of each corresponding subtask. The results of the detection and the classification are presented and discussed for each dataset: DDI-DrugBank and DDI-MedLine.

Table 4.2: Experimental results for the DrugNER Task using a CRF.

		Exact criterion			**Strict criterion**		
	Systems	**P**	**R**	**F1**	**P**	**R**	**F1**
DDI-DrugBank	WBI	0.90	0.89	0.90	0.88	0.87	0.87
	CRF	0.70	0.85	0.77	0.69	0.82	**0.75**
	CRFD	0.72	0.84	0.77	0.68	0.81	0.74
	CRFclusterK50Wiki	0.72	0.89	0.79	0.68	0.83	**0.75**
	CRFclusterK150Wiki	0.73	0.89	**0.80**	0.68	0.83	0.74
	CRFclusterK500Wiki	0.72	0.89	**0.80**	0.68	0.83	0.74
	CRFclusterK50MedLine	0.72	0.86	0.79	0.69	0.82	**0.75**
	CRFclusterK150MedLine	0.72	0.86	0.79	0.68	0.82	0.74
	CRFclusterK500MedLine	0.72	0.86	0.79	0.69	0.82	**0.75**
	CRFvec50Wiki	0.71	0.84	0.77	0.69	0.81	0.74
	CRFvec100Wiki	0.72	0.84	0.77	0.69	0.81	0.74
	CRFvec200Wiki	0.72	0.85	0.78	0.68	0.80	0.74
	CRFvec50MedLine	0.72	0.84	0.78	0.69	0.82	**0.75**
	CRFvec100MedLine	0.73	0.86	0.79	0.68	0.81	0.74
	CRFvec200MedLine	0.73	0.85	0.79	0.68	0.80	0.74
DDI-MedLine	WBI	0.81	0.74	0.77	0.61	0.56	0.58
	CRF	0.69	0.54	0.61	0.62	0.44	0.52
	CRFD	0.79	0.57	0.66	0.70	0.47	0.56
	CRFclusterK50Wiki	0.74	0.63	**0.68**	0.66	0.48	0.56
	CRFclusterK150Wiki	0.73	0.63	**0.68**	0.67	0.49	**0.57**
	CRFclusterK500Wiki	0.72	0.64	**0.68**	0.65	0.51	**0.57**
	CRFclusterK50MedLine	0.74	0.59	0.66	0.64	0.46	0.53
	CRFclusterK150MedLine	0.75	0.63	**0.68**	0.66	0.49	0.56
	CRFclusterK500MedLine	0.73	0.62	0.67	0.67	0.49	**0.57**
	CRFvec50Wiki	0.77	0.57	0.66	0.68	0.47	0.56
	CRFvec100Wiki	0.78	0.56	0.66	0.66	0.46	0.54
	CRFvec200Wiki	0.77	0.57	0.66	0.68	0.46	0.55
	CRFvec50MedLine	0.79	0.57	0.66	0.66	0.45	0.54
	CRFvec100MedLine	0.81	0.57	0.66	0.69	0.46	0.55
	CRFvec200MedLine	0.78	0.57	0.66	0.68	0.46	0.55

Results on DDI-DrugBank

For the detection task, the lexicon feature from DINTO achieved an increase in both Precision and Recall (and consequently, an improvement of 1% in F1 score). The results suggest that Word2vec features can potentially lead to improved detection performance. In general, the use of word clusters showed a significant increase in Recall values (from 84% to 89%) and hence a gain of 3% in F1. However, word clusters did not seem significant to alter the overall Precision values. As expected, the word cluster is a relevant feature to improve the coverage of the system.

The initial hypothesis was that Word2vec features trained on MedLine should provide better performance because these texts are focused on the biomedical domain. However, the results demonstrate that word clusters from Wikipedia, in general, had a better performance than those from MedLine. The main reason is that the size of the Wikipedia corpus is significantly larger than the release of Medline used in this work. Therefore, Wikipedia is the best option to train Word2vec models in the current settings, though Wikipedia covers a vast array of subjects, not necessarily related to the biomedical domain.

Word cluster features trained on MedLine always seem to provide the same scores, that is, there is no difference between to use a cluster which was calculated using k=50, k=150 or k=500. Word clusters trained on Wikipedia produced better results when the number of clusters is larger. More experiments with a different number of clusters are necessary to confirm or deny these results. In general, word clusters performed better than word vectors. To sum up, the results suggest that word clusters are the most important features for the detection subtask, achieving an improvement of 4% in Recall over the baseline system.

Regarding the results of the classification task on the DDI-DrugBank dataset, the use of Word2vec features did not necessarily give better results than the baseline system and might even be worse (see Table 4.2). The best F1 (75%) was obtained by five different strategies (see Table 4.2): baseline, word clusters (k=50) on Wikipedia, word clusters (k=50, k=500) on MedLine and word vectors (d=50) on MedLine. Similarly, DINTO did not overcome the baseline system yet. Therefore, while the experiments on the detection task show that the use of DINTO and Word2vec features could help to improve the performance, this positive effect does not seem to be present for the classification task.

Results on DDI-MedLine

In the detection task, the use of DINTO led to an increase in Precision, achieving 10% over the baseline system, and an increase of 3% in Recall. Thus, F1-score went up from 61% to 66%. Word cluster features generated from Wikipedia provided significant improvement of 6% in Recall, but with worse Precision than the combination of baseline with DINTO. As the case on DDI-DrugBank, the word clusters trained on MedLine obtains lower improvements. Moreover, word clusters seemed to perform better than word vectors. On the other hand, word vectors trained on MedLine showed Precision values very close to those obtained by the baseline system with DINTO.

Contrary to the evaluation of the DDI-DrugBank dataset, the use of DINTO increased the baseline Precision by 8% and the baseline Recall by 3% for the classification task. This improvement is because DINTO incorporates valuable information from several resources such as the ChEBI ontology, the DrugBank database and the ATC classification system[5] (a drug classification system developed by WHO). Word clusters (k=500) achieved the best performance by increasing the Recall (by 7%) and thus the F1 accordingly. However, word vectors do not seem to provide an improvement over the results achieved by DINTO.

Although this system does not provide better performance than the WBI system, the use of the DINTO feature shows a significant improvement by 9% in Precision over the WBI system, but with a sharp reduction in Recall.

[5]http://www.whocc.no/atc/structure_and_principles/

4.3.2 CEMP Task

Encouraged by the good results of the CRF-based methods in the previous edition of CHEMDNER (Krallinger, Leitner, Rabal, Vazquez, Oyarzabal and Valencia, 2015), the proposed CRF system is built for this task with the standard feature set, DINTO feature and the following features:

- A feature representing the long word shape of the current token. This feature is defined by mapping any uppercase letter, lowercase letter, digit, and other characters to *'X'*, *'x'*, *'0'*, and *'O'* respectively. For example, the long word shape of *'C1-6alkyl'* is *'X0O0xxxxx'*.
- A feature representing the brief word class of the current token. Consecutive uppercase letters, lowercase letters, digits, and other characters map to *'X'*, *'x'*, *'0'*, and *'O'* respectively. For example, the brief word shape of *'C1-6alkyl'* is *'X0O0x'*.

The combination of DINTO plus Word2vec clusters with k=100, it is the best model with an F1 of 83% in the development set. Using the number of clusters k=400, the Recall decreases a 1%. Both Word2vec clusters from MedLine and Wikipedia achieve very close results. Thus, based on the previous observations, the following settings were chosen for the runs:

- Run 1: DINTO-W2VMD-100K (the word clusters were trained on MedLine).
- Run 2: W2VMD-100K (DINTO is not used in this run).
- Run 3: W2VWIKI-200K (DINTO is not used in this run).
- Run 4: DINTO-W2VWIKI-200K (the word clusters were trained on Wikipedia).
- Run 5: CRF+DINTO (This run does not include the word clusters).

Table 4.3 shows the results obtained on the test dataset by all the runs. The best run (run 1) achieved a Recall of 82.15% and Precision of 86.3% (F1 of 84.17%). All the runs achieve results very close between them.

Table 4.3: CEMP results on the test dataset.

Systems	P	R	F1
Run 1	86.3%	**82.15%**	**84.17%**
Run 2	**86.32%**	81.95%	84.08%
Run 3	86.21%	81.99%	84.04%
Run 4	86.3%	81.85%	84.02%
Run 5	86.27%	82.02%	84.09%

As a result of the ranking, the five runs were placed in the positions 44 to 49 over a total of 93 submissions. The runs and the top system achieve very close performance in terms of Precision (only one point of difference). However, the Recall of the top system is almost 9% higher.

4.3.3 CDR Task

In the feature extraction phase, each token is represented with the basic feature set, DNorm tool feature and some special features are included:

- A binary feature that indicates whether the current token was found in a gazetteer of diseases, provided by the DNorm tool.
- A feature representing the long word shape of the current token. This feature is defined by mapping any uppercase letter, lowercase letter, digit, and other characters to *'X'*, *'x'*, *'0'*, and *'O'* respectively. For example, the long word shape of *'d-glycericacidemia'* is *'xOxxxxxxxxxxxxxxxx'*.
- A feature representing the brief word class of the current token. In this feature, consecutive uppercase letters, lowercase letters, digits, and other characters map to *'X'*, *'x'*, *'0'*, and *'O'* respectively. For example, the brief word shape of *'d-glycericacidemia'* is *'xOx'*.

The pipeline in GATE was also used to preprocess the sentences, and the Word2vec tool was trained using the Wikipedia and MedLine corpora. Different experiments on the development dataset are conducted in order to choose the best configuration for this system. The results of the validation set show that the use of DNorm seems to achieve a slight improvement in the baseline system. However, the combination of Word2vec features with DNorm does not seem to overcome the system using Word2vec features alone. That is, the system could ignore DNorm. Furthermore, the system that uses Word2vec features trained on MedLine without the DNorm gazetteer provides better results than those trained on Wikipedia. Moreover, if the system does not use the DNorm gazetteer, clusters provide better results than vectors. In particular, the best results are achieved using Word2vec clusters (k = 200 or k = 300) with MedLine. Apart from that, the DNorm only seems to help when DNorm is combined with Word2vec Wikipedia vectors.

Three different configurations were sent as the three runs for the DNER task, thereby the three best results on the development dataset were chosen. Thus, each run uses the MedLine data to train the Word2vec, and three different k values for the clustering (Run 1 uses 200, Run 2 uses 300 and Run 3 uses 100). None of them uses the DNorm gazetteer.

Table 4.4 shows the final results of the DNER task for the different runs of the proposed system. The best F-score is obtained by the run 1, which is also the best on the development dataset and improves 0.5% over the others runs. The Run 2 and Run 3 obtain very close results. The performance suggests that it is not suitable to use k values higher than 100. This system provides a significant improvement in Precision over the baseline system provided by the organizers (almost 49%), the difference in Recall is the opposite, since this model gets around 32% less. The primary source of this decrease is the high number of False Negatives (FN), mainly due to the lack of an accurate normalization system. For this reason, Table 4.5 shows the analysis of the results obtained in the disease mention recognition phase. As expected, the results in this evaluation achieve 77% in F-score in all runs which is very similar to the validation results.

Table 4.4: CDR results of normalization evaluate on the test set (500 documents).

Systems	TP	FP	FN	P	R	F1
Run 1	708	66	1280	**91.47%**	**35.61%**	**51.27%**
Run 2	703	69	1285	91.06%	35.36%	50.94%
Run 3	703	67	1285	91.30%	35.36%	50.98%

Table 4.5: CDR results of disease mention recognition evaluate on the test set (500 documents).

Systems	TP	FP	FN	P	R	F1
Run 1	3223	718	1201	81.78%	**72.85%**	**77.06%**
Run 2	3207	747	1217	81.11%	72.49%	76.56%
Run 3	3210	711	1214	**81.87%**	72.56%	76.93%

4.4 Conclusions

The main contribution of this chapter is the incorporation of word embedding features into a CRF-based NER system for drug entities. Besides, the DINTO ontology feature is a valuable resource for the drug and chemical compounds names recognition. However, the DNorm tool feature is not relevant for the detection and the classification of disease entities.

The results suggest that DINTO can lead to improving the performance over the detection subtask. As a conclusion from the experiments, the DINTO ontology is a useful resource for the drug name recognition task from scientific texts. For this reason, future work on how to better use DINTO in order to increase the performance of the task is taken into consideration. Moreover, the inclusion of additional semantic features from biomedical resources (such as DrugBank, CheBI, ChemIDPlus, the ATC classification system, Drugs@FDA[6] among others) is essential in order to improve performance for the classification subtask.

Contrary to the DINTO feature results, the use of the DNorm feature produces lower performance and could be discarded. Moreover, using MedLine instead of Wikipedia enhance the results and the k-means mode is better than the whole vector without DNorm. The final results show that word embedding clustering features achieve an improvement in Precision for the normalization task. Precisely, an increase of almost 49% over the baseline system provided by the organizers. However, it is slightly less in F-score because the presented approach only uses a list of diseases and their MeSH identifiers provided by the DNorm tool. On the contrary, in the recognition task, the system overcomes 28% due to the low number of False Positives.

In the initial hypothesis was confirmed and Word2vec features achieve a marked improvement in Recall for the detection task. Word cluster features trained on Wikipedia seem to provide the

[6]http://www.accessdata.fda.gov/scripts/cder/drugsatfda/

most satisfactory results. More experiments are necessary to determine the optimum number of clusters for the task. Although in general, these results are not better than those achieved by the top system in the DrugNER Task, the use of word embeddings for this task is worth further research.

The experiments conducted on the DDI corpus compare this approach with the participating systems of the DrugNER Task in the DDIExtraction Task. In general, the proposed system does not perform better than the top system (WBI) in this shared task. However, the results for the classification task on the DDI-MedLine dataset show that DINTO could be a valuable resource to improve Precision. The WBI system provided an F1 of 87.8% on DDI-DrugBank (which is very close to the IAA (0.91)), but performed worse on the DDI-MedLine dataset (showing an F1 of 58.1%). It stands to reason that this system could have already reached the maximum threshold results for the DDI-DrugBank dataset. On the other hand, there is much room for improvement in the DDI-MedLine dataset. The results reported in (Liu, Tang, Chen, Wang and Fan, 2015) are better than those provided by the WBI system. However, since the authors only provide results for the whole DDI corpus, the performance of their system on each dataset is unknown and whether their system overcomes the WBI system on the DDI-MedLine dataset.

As future work, the Word2vec tool should be trained using a large set of MedLine abstracts. It could provide better results than those obtained from the Word2vec model trained on Wikipedia. Since MedLine is a biomedical literature database, Medline abstracts should provide better word representations for drug entities than those obtained from Wikipedia articles. Besides, another tool such as the Global Vectors for Word Representation (GloVe) (Pennington et al., 2014) should be tested in future works to investigate their performance in the biomedical domain. Furthermore, an error analysis to determine the leading causes of wrong detection and classification should be carried out.

Chapter 5

Matrix-Vector Recursive Neural Networks for Drug-Drug Interaction Extraction

The purpose of this chapter is to explore in detail how a Recursive Neural Network (RecNN) can be applied to classify drug-drug interactions from biomedical texts (Suárez-Paniagua and Segura-Bedmar, 2016). The hypothesis is that Deep Learning architectures can outperform classical machine learning approaches without using a large set of human-defined features for the DDIExtraction Task. Precisely, the system is based on Matrix-Vector Recursive Neural Network (MV-RNN) (Socher et al., 2012) built from the Stanford constituency trees of the sentences. The experiments show a low performance that may be probably due to the Stanford parser (Klein and Manning, 2003) cannot capture the structural complexity of sentences because that DDIs are usually described by long sentences with complex structures (such as subordinate clauses, appositions, and coordinate structures, among others).

5.1 Introduction

Most of the systems developed for the DDIExtraction Task have been based on Support Vector Machines (SVM) using linear and nonlinear kernels. These models are the state-of-the-art performance obtaining 77.5% F-score for detection and 67% F-score for classification (Kim et al., 2015). All of them use large and rich sets of linguistic features proposed by text miners and domain experts.

Deep Learning methods can be an exciting alternative to classical methods since they can learn the best features to represent a problem. Furthermore, the prominent use of these techniques for Natural Language Processing tasks and its excellent performance on this field makes it a promising technique in Relation Extraction, such as Convolutional Neural Network (CNN) (Zeng et al., 2014) or Recurrent Neural Network (Xu et al., 2015). Recursive Neural Network (RecNN) is a Deep Learning architecture, which is created from the constituency parse tree, that captures the semantic meaning for phrases and sentences. Precisely, RecNN used in Matrix-Vector spaces (MV-RNN) was the first Deep Learning architecture that obtained improvements in Relation Extraction (Socher et al., 2012). This model introduces a RecNN that captures the compositional vector representation of long phrases or sentences. The compositionality is by definition the important quality of natural language that determines the meaning of its words and the rules used to combine them. To this end, the model assigns a vector and a matrix to every word, and it learns a compositional function that captures how this constituent changes the meaning of their neighbours through matrix-vector spaces.

From the review of the related work, it seems that any work has performed a full and detailed study of the DDI classification by using MV-RNN. Therefore, the aim is to explore this architecture in the biomedical domain and work in a multi-class classification setting for reporting a complete analysis on the DDI corpus, studying in depth its performance for each type of DDI. The main advantage of this approach over other Deep Learning architectures is that captures the semantic information in the whole sentence and each word through the matrix and the vector, respectively.

5.2 Method

The approach is based on the Recursive Neural Networks, which take the parse tree of the sentences as the structure of the network. In particular, a RecNN with Matrix-Vector spaces was the first Deep Learning architecture that obtained improvements in the classification of semantic relationships (Socher et al., 2012). This model can determine the meaning of each word and the rules used to combine them in long sentences. To this end, the model assigns a vector and a matrix to every word, and it learns a compositional function for computing these representations (Figure 5.1).

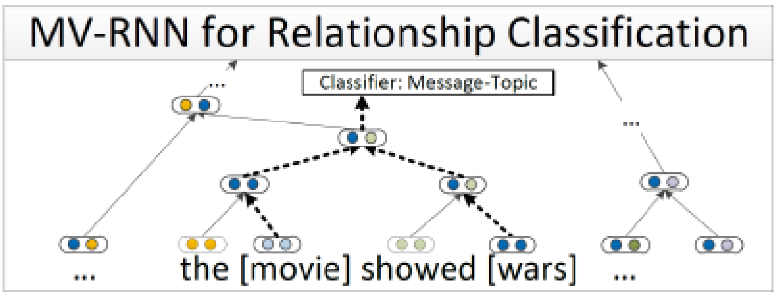

MV-RNN for Relationship Classification
Classifier: Message-Topic
...
...
...
the [movie] showed [wars]
...

Figure 5.1: An example of how MV-RNN architecture learns the vectors in the nodes of the path between the two entities (dotted line) to classify their relationship (Socher et al., 2012).

Firstly, MV-RNN uses as input a binary parse tree of phrases and sentences of arbitrary syntactic type and length from the Stanford Parser (Klein and Manning, 2003) as the RNN structure. Then, MV-RNN learns, in every node of the tree, a vector that represents the meaning of a constituent (a word or a sentence) and a matrix that captures how this constituent changes the meaning of their neighbours. Initially, it uses the pre-trained 50-dimensional word vectors from (Collobert and Weston, 2008) and the word matrices as an identity matrix with a small Gaussian noise. Afterwards, the MV-RNN architecture computes the parent vector p of each node as a single layer neural network:

$$p = g(W \begin{bmatrix} C_1 c_2 \\ C_2 c_1 \end{bmatrix} + b) \tag{5.1}$$

where c_1 and c_2 are the word vectors of their children in the binarized tree with dimensionality n, C_1 and $C_2 \in \mathbb{R}^{n \times n}$ are matrices for single words, $W \in \mathbb{R}^{n \times 2n}$ is a matrix that maps both words back into the same n-dimensional space, g is a nonlinearity function and b is the bias term. Moreover, another function computes the non-terminal phrase matrices:

$$P = W_M \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} \tag{5.2}$$

where $W_M \in \mathbb{R}^{n \times 2n}$ are the weight matrices. Finally, the MV-RNN uses the computed vector of the highest nodes in the path between the pairs of words as features for predicting a DDI type label using a simple Softmax classifier.

MV-RNN was adapted in (Socher et al., 2012) for the SemEval-10 task 8, whose goal was the classification of relationships between nominals. Thus, the initial step is to transform the DDI corpus to the format of the SemEval-2010 task 8. Since the implementation of MV-RNN does not deal with discontinuous entities, DDI candidates involving this kind of entities are removed. Besides, whether a sentence contains more than one interaction they are separated into independent sentences, i.e. one instance per interaction. Following this process, a total of 33351 sentences are created. However, teh sentences from the SemEval-2010 task 8 dataset are much simpler than the sentences in the DDI corpus. Drug-drug interactions are usually described by long sentences with complex structures (such as subordinate clauses, appositions, and coordinate structures, among others). Moreover, many drugs have very long and complex names, like chemical compounds (for example, *1,3-difluoro-2-propanol*). These drug names pose a significant challenge for the tokenization task of the biomedical texts. Concretely, the Stanford parser cannot provide accurate tokenization of the sentences in the DDI corpus. Wrong tokenization can cause a wrong syntactic tree parser, and thereby, a bad input for the MV-RNN. For this reason, chemical compound names were replaced by easier names of common drugs. For example, *1,3-difluoro-2-propanol* was substituted by *Rifampin*. Similarly, numerical expressions were also simplified by a common token as part of the pre-processing phase.

5.3 Evaluation

This section summarizes the evaluation results with the method MV-RNN with the DDI corpus and provides detailed analysis and discussion. Precision (P), Recall (R) and F-score (F1) are the measures of the results for all the categories in the classification.

Table 5.1 shows the performance of MV-RNN over the DDI corpus test dataset. The model achieves an F-score of 46% using syntactic information for building the RNN architecture. In general, Precision is higher than Recall due to a large number of False Negatives in each class caused by the misclassification. The class *advice* achieves the best performance (54% in F-measure) for the other classes because these recommendations follow specific patterns and are easy to learn.

Table 5.1: Results on DDI Corpus using MV-RNN without external features.

Classes	**TP**	**FP**	**FN**	**P**	**R**	**F1**
Effect	163	206	197	0.44	0.45	0.45
Mechanism	105	102	194	0.51	0.35	0.42
Advice	108	71	113	0.60	0.49	0.54
Int	36	18	60	0.67	0.38	0.48
Overall	412	397	564	0.51	0.42	0.46

Table 5.2 shows the results adding external features such as POS tags, the WordNet hypernyms and the name entity tags of the two words to the computed vector of the highest node in the relationship for the classification in the Softmax layer. These three features increase the performance F-measure (+4%) and the Recall for all the classes. Although the features raise the number of instances classified correctly, the False Positives are 38 instances bigger than without using external features. It may be due to an over-fitting in the Softmax layer because in all the cases the number of False Negatives decreases whereas that the False Positives increases for the Table 5.1 causing a trade-off problem.

The leading cause of the low performance is caused by the Stanford parser which cannot build the syntactic trees of sentences from the DDI corpus correctly. Mainly, the sentences that involve drug interactions usually have complex structures (such as subordinate clauses, appositions and coordinate structures, complex named entities among others). Thus, wrong syntactic

Table 5.2: Results on DDI Corpus using MV-RNN with external features.

Classes	**TP**	**FP**	**FN**	**P**	**R**	**F1**
Effect	193	232	167	0.45	0.54	0.49
Mechanism	121	110	178	0.52	0.40	0.46
Advice	119	76	102	0.61	0.54	0.57
Int	37	17	59	0.69	0.39	0.49
Overall	470	435	506	0.52	0.48	0.50

trees involve wrong RNN structures that cannot capture the compositionality of sentences.

5.4 Conclusions

In this work, a Recursive Neural Network used in Matrix-Vector spaces is explored for the extraction of interactions between drugs in the DDI corpus. From the review of the related work, Deep Learning architectures outperform the most common machine learning algorithms applied to relation extraction so far. MV-RNN can learn the meaning of a word and how that word modifies the context of the sentence through the combination of vectors and matrices. This recursive network contains the parsing information of each sentence regardless of length and grammatical structure.

However, MV-RNN does not seem to provide satisfactory results for the classification of DDIs. Thus, the experimental results are much lower than those reported using a CNN (Liu, Tang, Chen and Wang, 2016). This low performance is mainly because the Stanford parser cannot capture the structural complexity of sentences in the biomedical literature. The experiments show that blinding the entities is not enough to reduce the complexity of drug mentions. Thus, a biomedical parser capable can provide accurate tokenization and syntactic trees of the sentences. Furthermore, MV-RNN uses the WordNet dictionary in order to achieve the hypernyms for interacting drugs. However, most drugs are not included in WordNet since it is not a biomedical domain specific resource.

As future work, replacing the initial word vectors from (Collobert and Weston, 2008) by those from a new word vector model generated using a state-of-the-art word embedding system, like Word2vec (Mikolov, Sutskever, Chen, Corrado and Dean, 2013) trained on an extensive collection of biomedical texts (for example, the latest version of MedLine) could improve the semantic representation of each word because of it will also include biomedical technical terms and jargon, which are not generally represented in (Collobert and Weston, 2008). Moreover, the position embeddings and negative instance filtering techniques could improve the presented results. Alternatively, instead of the Stanford parser, it is also planned to use a biomedical parser capable of capturing the complex structures of the biomedical sentences in order to build the RNN structures. Instead of using WordNet, other biomedical terminological resources could include the hypernyms such as the UMLS Methatesaurus (Aronson, 2001) or the ATC drug classification system [1].

[1]http://www.whocc.no/atc_ddd_index/

Chapter 6

Convolutional Neural Networks for Biomedical Relation Extraction

The MV-RNN poorly captures the syntactic information of the DDI Corpus given that the Stanford parser is not prepared for biomedical texts. For this reason, the propagation of errors from the first step makes a low performance in the DDIExtraction Task using this architecture. Thus, the purpose of this chapter is to examine a Convolutional Neural Network (CNN) for classifying DDIs from biomedical texts, which only uses word embeddings as input features without the use of external resources. The proposed system is a CNN architecture with one convolutional layer for creating a more computationally efficient model. Therefore, individual experiments can determine the best configuration for this architecture (Suárez-Paniagua et al., 2017) in the DDIExtraction Task. One of the goals of this section is to set the best parameter of this basic CNN that should be considered for future researches.

Consequently, the proposed CNN architecture is tested on two different datasets from scientific publications. Concretely, this section describes two participations at SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications and at SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers. The subtask of extraction of relationships between two identified keyphrases of the SemEval 2017 Task 10 can be very helpful in improving search engines for scientific articles. The extraction and the classification of relationships between entities in scientific papers of the SemEval 2018 Task 7 are performed by the CNN classifier using POS tags and the distances to the target entities as part of the embedding for each word.

Most CNN architectures incorporate a pooling layer to reduce the dimensionality of the convolution layer output, preserving relevant features and removing irrelevant information. Concretely, all the previous CNNs for the DDIExtraction Task used max-pooling layers. Additionally, this chapter shows an evaluation of the performance for various pooling methods (max-pooling, average-pooling and attentive pooling) and their combinations for the DDIExtraction Task (Suárez-Paniagua and Segura-Bedmar, 2018).

6.1 Introduction

Information extraction (IE) from both structured and unstructured data sources may significantly assist the pharmaceutical industry by enabling the identification and extraction of relevant information as well as providing a novel way of reducing the time spent by health-care professionals to review the literature. Most of the previous research on the extraction of DDIs from biomedical texts implement supervised machine-learning algorithms with extensive feature sets, which are manually defined by text miners and domain experts. The prominent use of Deep Learning in NLP and its good performance in this field makes it a promising technique for the task of RE. Deep Learning methods are potential alternatives to classical supervised machine-learning algorithms because they can automatically learn the most appropriate features for a given task. Matrix-Vector Recursive Neural Network (MV-RNN) (Socher et al., 2012), Recurrent Neural Network (RNN) (Xu et al., 2015) and Convolutional Neural Network (CNN) (Zeng et al., 2014) have been successfully applied to RE tasks.

CNN is a robust Deep Learning architecture which has exhibited excellent performance in many NLP tasks such as sentence classification (Kim, 2014), semantic clustering (Wang, Xu, Xu, Tian, Liu and Hao, 2016) and sentiment analysis (dos Santos and Gatti, 2014). One of its main advantages is that it does not require the definition of hand-crafted features; instead, it can learn the most suitable features for the task automatically. This model combines the word embeddings of an instance (i.e. a sentence or a phrase containing a candidate relation between two entities) using filters in order to construct a vector which represents this instance. Finally, a Softmax layer assigns a class label to each vector. (Zeng et al., 2014) developed the first work that used CNN for RE using the SemEval-2010 Task 8 dataset (Hendrickx et al., 2009). This work concatenated the word embeddings with a new position embedding which represents the relative distances of each word to the two entities in the instance relation in an embedding vector. Moreover, they added a nonlinear layer after the CNN architecture to learn more complex features obtaining an F1-score of 69.7%. They obtained an improvement of 13% by adding external lexical features such as the word embeddings of the entities, their WordNet hypernym and the word embeddings of the context tokens.

In particular, the hypothesis is that CNN may be an effective method to learn the best feature set to classify DDIs without the need for manual and extensive feature engineering. Although previous works have already incorporated the use of CNNs, none of them reported a detailed study of the influence of the CNN hyper-parameters on the performance for DDIExtraction Task.

Concretely, different window sizes should be tried to adapt the filter size parameter for DDIs sentences because they are usually very long and their interacting drugs are often far from each other (the average distance between entities for all the instances in the train set is 14.6). Moreover, the performance of the system is provided for each DDI type on the whole test set and each dataset of the DDI corpus (DDI-DrugBank and DDI-MedLine).

Pooling layers are used to achieve more compact representations, and they must preserve relevant features while removing irrelevant details. There are different operations for appliying as the pooling layer, such as taking the average or the maximum, or a learned linear combination of its inputs. All the previous CNN based systems for the DDIExtraction Task used max-pooling layers. Therefore, one of the goals of this chapter is to evaluate the effect of several pooling operations (such as max-pooling, average-pooling and attentive pooling) on the results of the task separately, but also their combination. Summing up, the comparison of different pooling operations and their combination on the performance in a CNN architecture for DDI classification had not been evaluated before, and it is one of the significant contributions of the thesis.

Nowadays, a deluge of scientific articles is published every year, which demonstrates that we are living in an emerging knowledge era. An essential drawback of this situation is that the study of a given field or problem requires reviewing a considerable number of scientific publications, becoming such a very arduous task. Thus, experts cannot deal with the high increase in the publication of scientific articles and it is very hard to be up to date about the state-of-the-art techniques. The automatically classification of the concepts or keyphrases and their relationships from scientific articles with NLP techniques can reduce the vast of time for this process. Most search engines apply linguistic normalization (such as lemmatization or stemming), and some of them also exploit the semantic analysis of texts in order to detect concepts to improve their Recall.

The goal of the ScienceIE Task at SemEval 2017 (Augenstein et al., 2017) is the extraction of keyphrases (such as MATERIALS, PROCESSES and TASKS) and relationships between them from scientific articles. This competition provides a common evaluation framework to researches allowing a fair way to evaluate and compare their approaches. This participation focuses on the subtask of extracting relationships between keyphrases. In particular, these relationships are HYPONYM-OF (for example, *'femur'* is HYPONYM-OF *'bone'*), SYNONYM-OF (for example, *'ophthalmologist'* is SYNONYM-OF *'oculist'*) and NONE. The detection of these relationships between keyphrases can improve the performance of current researches. The Semantic Relation Extraction and Classification in Scientific Papers task at SemEval-2018 task 7 (Gábor et al., 2018) provides a framework for measuring the automatic annotation performance by models which are trained on scientific publications abstracts. The task defines six categories of relations between concepts and proposes two subtasks: the classification of the relations between two entities in the predefined categories, which includes two scenarios according to the data used: clean or noisy; and the extraction of the relations given the entities from the clean data, which also could involve their subsequent classification. This section describes the participation in both tasks using the CNN previously proposed.

The model uses as the input of each instance the transformation into real value vectors of the words of the sentence, the distances to the target entities of each word and the Part-of-Speech types. Furthermore, a sampling technique alleviates the imbalance issues of the dataset equalizing the number of instances for all the classes.

6.2 Method

This approach is based on the CNN model proposed in (Kim, 2014), which was the first work to exploit a CNN for the task of sentence classification. This model infers the class of each sentence and reaches to good results without the need for external information. To this end, the model applies convolutional filters to the input through several windows of different sizes and computes an output vector that describes the whole sentence. Finally, this vector is used in a classification layer to assign a class label. This section presents the model for the particular case of sentences that describe DDIs. Instead of using text classification CNN implemented by in (Kim, 2014)[1] with Theano (a python library for mathematical computation[2], the system is an adaptation of the implementation provided by Denny Britz[3] based on TensorFlow (an open-source library for machine learning[4]). TensorFlow has a graphic visualization of the model and generates summaries of the parameters to keep track of their values, thus, simplifying the study of the parameters.

Figure 6.1 shows the whole process of the CNN model from its input, which is a sentence with marked entities, until the output, which is the classification of the instance into one of the classes.

[1]https://github.com/yoonkim/CNN_sentence
[2]http://deeplearning.net/software/theano/
[3]https://github.com/dennybritz/cnn-text-classification-tf
[4]https://www.tensorflow.org/

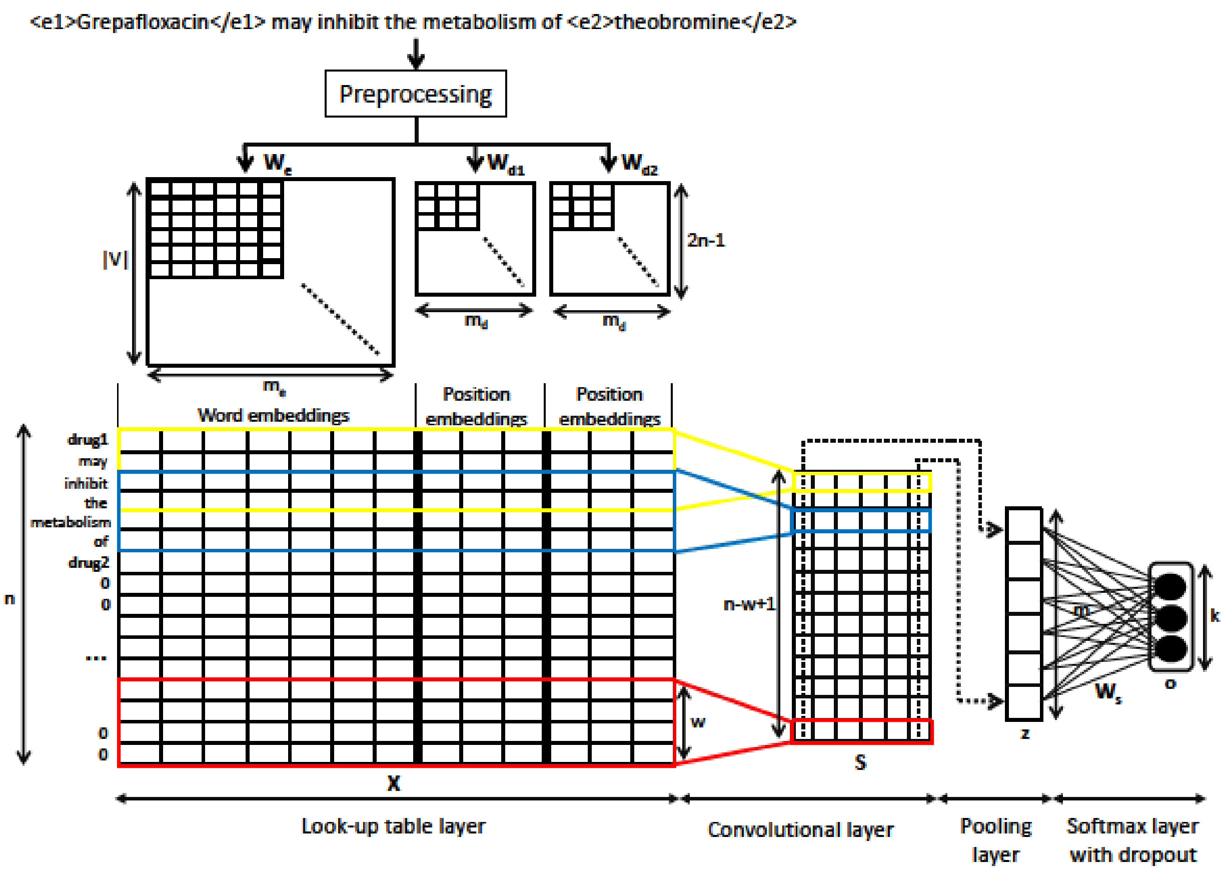


Figure 6.1: CNN model for classifying a DDI instance.

6.2.1 Datasets

This chapter explores the CNN performace for the DDI Corpus. However, other datasets from scientific publications are used with the same model in order to ensure the viability in a particular domain.

SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications

The valuable contribution of the ScienceIE challenge was to provide an annotated corpus for training and evaluating supervised algorithms to extract Keyphrases from Scientific Publications. The whole corpus contains 500 journal articles about Computer Science, Material Sciences and Physics from ScienceDirect[5]. The corpus is split into training, development and

[5]http://www.sciencedirect.com/

testing sets with 350, 50 and 100 documents, respectively. A detailed description of the method used to collect and process documents can be found in (Augenstein et al., 2017).

SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers

The SemEval-2018 Task 7 provides an annotated corpus for training and testing the participating systems. The dataset contains 350 and 150 abstract from scientific articles for training and testing set, respectively.

The relation instances are divided into the following classes: *USAGE*, *RESULT*, *MODEL*, *PART WHOLE*, *TOPIC* and *COMPARISON*. All of them are asymmetrical except *COMPARISON*, where both entities are involved in the same bidirectional relation. A detailed description and analysis of the corpus and its methodology used to collect and process the scientific abstracts can be found in (Gábor et al., 2018).

6.2.2 Pre-processing phase

Each pair of drugs in a sentence represents a possible relation instance. The CNN model classifies each of these instances.

The DDI corpus contains a tiny number of discontinuous drug mentions (only 47). The following noun phrase shows an example of discontinuous mention *'ganglionic or peripheral adrenergic blocking drugs'*, which contains two different drug mentions: *ganglionic adrenergic blocking drugs* and *peripheral adrenergic blocking drugs*, with the first one being a discontinuous entity. As this kind of mentions only produces a tiny percentage (1.26%) of the total number of instances, they are removed for the training set.

Firstly, following a similar approach as that described in (Kim, 2014), the sentences were tokenized and cleaned (converting all words to lower-case, replacing numbers by the label *NUM*, separating special characters with white spaces by regular expressions and performing the drug blinding). Then, the two mentions of each instance were replaced by the labels *'entity1'* and *'entity2'* for the two interacting entities, and by *'entity0'* for the remaining drug mentions. This method is known as entity blinding and verifies the generalization of the model. For instance, the sentence: *'Amprenavir significantly decreases clearance of rifabutin and 25-O-desacetylrifabutin'* should be transformed into the following relation instances:

1. *'drug1 significantly decreases clearance of drug2 and drug0'* for the relation (*Amprenavir, rifabutin*)

2. *'drug1 significantly decreases clearance of drug0 and drug1'* for the relation (*Amprenavir, 25-O-desacetylrifabutin*)

3. *'drug0 significantly decreases clearance of drug1 and drug2'* for the relation (*rifabutin, 25-O-desacetylrifabutin*)

However, the relation between (*rifabutin, 25-O-desacetylrifabutin*) in the sentence: *'Amprenavir significantly decreases clearance of rifabutin and 25-O-desacetylrifabutin'* cannot be a DDI because these drugs are conjuncts in the same coordinate structure. Therefore, all the instances that their drugs occur in coordination can be ruled out. Similarly, all the instances that their drugs occur in a hyponymous apposition (Meyer, 2007) can be discarded. Apposition is a noun phrase that follows another noun phrase and further describes or explains it. In a hyponymous apposition, the noun phrases are related by the relation of hyponymy. The following sentence shows an example of this kind of structure where first part of the sentence is an apposition: *'Anticoagulants, such as heparin and warfarin, are often given prophylactically to prevent DVT.'* The relation instances (*Anticoagulants-heparin*), (*Anticoagulants-warparin*) and (*heparin-warfarin*) can be directly removed from the set of instances. Following the beneficial results of using negative filtering preprocessing on DDI (Chowdhury and Lavelli, 2013*b*; Kim et al., 2015; Liu, Tang, Chen and Wang, 2016), a set of regular expressions are defined to capture the sentences that contain the structure of the most frequent coordination and hyponymous apposition in the DDI corpus. Thus, the imbalance problem of the DDI corpus is also alleviated (almost 85% of instances are negatives).

These regular expressions achieve to automatically identify and rule out around 35% of negative instances (8,409) from the training dataset and approximately 29% (1,670) from the test dataset, while mistakenly filter out 150 and 32 positive instances from training and test datasets, respectively. At the end of this process, 19,233 relation instances (positives and negatives) are extracted to train the network and 4018 to test its performance.

6.2.3 Word table layer

After the pre-processing phase, the input matrix is created for the CNN architecture. The input matrix should represent all training instances for the CNN model; therefore, they should have the same length. The maximum length of the sentence in all the instances is fixed (denoted by n) and then extended those sentences with lengths shorter than n by padding with an auxiliary token *'0'*.

Moreover, each word has to be represented by a vector. To do this, two different options are considered: (a) to randomly initialize a vector for each different word, or (b) to use a pre-trained word embedding model which replaces each word by its word embedding vector obtained from this model: $\mathbf{W}_e \in \mathbb{R}^{|V| \times m_e}$ where V is the vocabulary size and m_e is the word embedding dimension. Finally, a vector is obtained $\mathbf{x} = [x_1; x_2; ...; x_n]$ for each instance where each word of the sentence is represented by its corresponding word vector from the word embedding matrix. p_1 and p_2 are defined as the positions of the two interacting drugs mentioned in the sentence.

The following step involves calculating the relative position of each word to the two interacting drugs, $i - p_1$ and $i - p_2$, where i is the word position in the sentence. For example, the relative distances of the word *'inhibit'* in the sentence shown in Figure 6.1 to the two interacting drug mentions *'Grepafloxacin'* and *'theobromine'* are 2 and -4, respectively. In order to avoid negative values, the range $(-n+1, n-1)$ are transformed into the range $(1, 2n-1)$. Then, these distances are mapped into a real value vector using two position embedding $\mathbf{W}_{d1} \in \mathbb{R}^{(2n-1) \times m_d}$ and $\mathbf{W}_{d2} \in \mathbb{R}^{(2n-1) \times m_d}$. Finally, the input matrix $\mathbf{X} \in \mathbb{R}^{n \times (m_e + 2m_d)}$ is represented by the concatenation of the word embeddings and the two position embeddings for each word in the instance.

One of the objectives of this work was to study the effect of the pre-trained word embeddings on the performance of CNNs. Additionally, the CNN is trained with different pre-trained word embedding models and compared with the random initialization. First, the different word embedding models using the toolkit Word2vec (Mikolov, Sutskever, Chen, Corrado and Dean, 2013) are trained on the BioASQ 2016 dataset (Krithara et al., 2016), which contains more than 12 million MedLine abstracts. Skip-gram and continuous bag-of-words (CBOW) architectures of Word2vec are applied with the default parameters used in the C version of the Word2vec toolkit (i.e. minimum word frequency 5, the dimension of word embedding 300, sample threshold 10-5 and no hierarchical Softmax). In addition, different values for the parameters context window (5, 8 and 10) and negative sampling (10 and 25) are also used to pre-train the models. For a detailed description of these parameters, refer to (Mikolov, Sutskever, Chen, Corrado and Dean, 2013). A word embedding model on the XML text dump of the English 2016 version of Wikipedia[6] is also used with the default parameters of Word2vec.

6.2.4 Convolutional layer

Once the input matrix is defined, a filter matrix $\mathbf{f} = [f_1; f_2; ...; f_w] \in \mathbb{R}^{w\times(m_e+2m_d)}$ is applied to a context window of size w in the convolutional layer to create higher level features. For each filter, a score sequence $\mathbf{s} = [s_1; s_2; ...; s_{n-w+1}] \in \mathbb{R}^{(n-w+1)\times 1}$ is obtained for the whole sentence as

$$s_i = g(\sum_{j=1}^{w} f_j x_{i+j-1}^T + b) \tag{6.1}$$

where b is a bias term and g is a nonlinear function (such as tangent or sigmoid). Figure 6.1 represents the m total number of filters with the size w in the matrix $\mathbf{S} \in \mathbb{R}^{(n-w+1)\times m}$. However, the same process can be applied to filters with different sizes by creating additional matrices that would be concatenated in the following layer. The filter size is an important parameter in the CNN model, and may influence its performance because it directly defines the size of the vector, which represents each instance. Moreover, window contexts have been traditionally exploited by most relation-classification systems. In particular, a window with a size of 3 is

[6] http://mattmahoney.net/dc/text8.zip

widely adopted (Giuliano et al., 2006).

6.2.5 Pooling layer

The main goal of the pooling operation is to extract the most relevant features of each filter using an aggregating function. This layer is a critical part of the architecture because it compacts the filtered information into a vector representation. This vector may capture the salient parts of the text and can be directly used as input of the classifier layer. For this reason, the selection of a correct pooling layer improves the final classification of the model. The aim of this chapter is also to explore and select the best pooling operation for DDI and create a vector $\mathbf{z} = [z_1, z_2, ..., z_{m*k}]$, whose dimension is the total number of filters m by the number of different filter length k that represents the relation instance. In this study, three different pooling layers are applied to the output matrix of the convolutional layer.

Max-pooling

The max function is the most common choice for the pooling layer in CNN architectures. This operation produces a single value in each filter as

$$z_f = max\{\mathrm{s}\} = max\{s_1; s_2; ...; s_n\} \tag{6.2}$$

Average-pooling

The average pooling is commonly used in image classification tasks, but it is not very popular in NLP tasks. In this study, its performance was measure for the DDIExtraction Task. In this case, the operation computes the average of each filter values:

$$z_f = mean\{\mathrm{s}\} = mean\{s_1; s_2; ...; s_n\} \tag{6.3}$$

Attentive pooling

The attentive pooling is a neural attention mechanism which focuses on the relevant words, capturing the important semantic information without using lexical resources or NLP tools. For this work, the attentive pooling method is the same as the proposed by (Zhou et al., 2016) for the task of relation extraction with LSTM.

In the case of CNN, the operation uses a weight vector $w^T \in \mathbb{R}^{m \times 1}$ which is multiplied by a filter normalization $\mathbf{M} \in \mathbb{R}^{(n*k) \times m}$. The resulting vector $\alpha \in \mathbb{R}^{n \times 1}$ determinates the relevant values of each word in the sentence for the classification.

$$\mathbf{M} = tanh(\mathbf{S}) \tag{6.4}$$

$$\alpha = Softmax(\mathbf{M}w^{\alpha}) \tag{6.5}$$

Finally, the vector α is multiplied by the filter matrix $\mathbf{S}$ to reduce its dimensionality given by the relevancy of their words

$$\mathbf{z}^* = tanh(\alpha^T \mathbf{S}) \tag{6.6}$$

6.2.6 Softmax layer

Before performing the classification, a dropout prevents the over-fitting of the network. To do this, the elements of $\mathbf{z}$ are randomly set to zero with a probability p following a Bernoulli distribution in order to generate a reduced vector $\mathbf{z}_d$. After that, this vector is fed into a fully connected Softmax layer with weights $\mathbf{W}_s \in \mathbb{R}^{m \times k}$ to compute the output prediction values for the classification as

$$\mathbf{o} = \mathbf{z}_d \mathbf{W}_s + d \tag{6.7}$$

where d is a bias term; in the dataset, there are $k = 5$ classes (*advice*, *effect*, *int*, *mechanism* and non-DDI). At test time, the vector $\mathbf{z}$ of a new instance is directly classified by the Softmax layer without a dropout.

6.2.7 Learning

For the training phase, the CNN parameter set to be learnt is defined as $\theta = (\mathbf{W}_e$, $\mathbf{W}_{d1}$, $\mathbf{W}_{d2}$, $\mathbf{W}_s$, $\mathbf{F}_m)$, where $\mathbf{F}_m$ are all of the m filters $\mathbf{f}$. For this purpose, the conditional probability of a relation r is obtained by the Softmax operation as

$$p(r|\mathbf{x},\theta) = \frac{\exp(\mathbf{o}_r)}{\sum_{l=1}^{k}\exp(\mathbf{o}_l)} \tag{6.8}$$

to minimize the negative log-likelihood function for all instances $(\mathbf{x}_i, y_i)$ in the training set T as follows

$$J(\theta) = -\sum_{i=1}^{T}\log p(y_i|\mathbf{x}_i,\theta) \tag{6.9}$$

Besides, the objective function is minimized to learn the parameters with the stochastic gradient descent using the Adam update rule (Kingma and Ba, 2014) applied over shuffled mini-batches. Finally, l_2-regularization is added to the weights of the Softmax layer $\mathbf{W}_s$ to prevent over-fitting.

6.3 Evaluation

This section describes the evaluation results with the CNN model on the datasets and provides a detailed analysis and discussion. The results were measured using the Precision (P), Recall (R) and F-measure (F1) for all of the categories in the classification.

6.3.1 DDI Corpus

To investigate the effect of the different parameters, an evaluation process to choose the best model was followed by selecting the parameters separately in a validation set to obtain the best values. Since the DDI corpus is only split into training and test datasets, 2748 instances (candidate pairs) (10%) are selected from the training dataset at the sentence level, forming the validation set, which was used for all the experiments to fine-tune the hyper-parameters of the architecture.

In order to validate each setting, a statistical significance analysis between the models is performed. For this purpose, the significance is tested with the χ^2 and p-value statistics. Two models produce different levels of performance whether χ^2 is greater than 3.84 and p-value is lower than 0.05.

Firstly, the performance is shown in a learning curve to find the optimal number of epochs for which the system achieves the best results with the stopping criteria. Secondly, a basic CNN was computed using predefined parameters to create a baseline system, and its results are analysed. Thirdly, the effects of the filter size and the selection of different word embeddings and position embeddings were observed. Finally, a CNN model using the best parameters found in the above steps was created. For all the experiments, the values of the remainder parameters for the systems are defined as follows:

- Maximal length $n = 128$.
- Filters for each window size $m = 200$.
- Dropout rate $p = 50\%$.
- l_2-regularization $= 3$.
- Mini-batch size $= 50$.
- Rectified Linear Unit (ReLU) as the nonlinear function g.

The parameter n is the maximum length in the dataset after the pre-processing phase, m is the same as in (Liu, Tang, Chen and Wang, 2016), and the rest of the parameters are the same as in (Kim, 2014).

Learning curve

Figure 6.2 shows the learning curve for the CNN model from random initialization, i.e. instead of using pre-trained word embeddings as input features for the architecture, random vectors of 300 dimensions are generated using a uniform distribution in the range $(-1, +1)$. The curve

Table 6.1: Number of instances in each dataset of the DDI corpus after the pre-processing phase. The class Other represents the non-interaction between pairs of drug mentions.

Classes	DDI-DrugBank	DDI-MedLine	Total
Advice	1028	14	1042
Effect	1815	214	2029
Int	272	12	284
Mechanism	1535	83	1618
Other	26486	1892	28373
Total	31136	2215	33351
Train	25885	1778	27663
Test	5251	437	5688

shows the performance of each iteration of a learning step (epoch), and it measures the F1 performance for the classification of the Softmax layer. According to this learning curve, the best validation F1 is reached with 27 epochs (77.7%), which is the optimum number of epochs to train the network (see the green point in Figure 6.2). Moreover, the training F1 is still around 100%, and the validation F1 does not improve by using more epochs. There is not a large gap between the training and validation F1, and therefore, the model does not appear to produce over-fitting. Figure 6.2 also shows that the validation and test variation perform very similar, confirming that the choice of the parameters in the validation set is also valid for the test set. Finally, the network is trained with 25 epochs in the following experiments because after this point the model starts to decrease its performance. Moreover, it was the value chosen by (Kim, 2014).

Baseline performance

As previously mentioned, the baseline CNN model is trained with random initialization (i.e. without pre-trained word embeddings) of 300 dimensions, filter size (3, 4, 5) and no position embeddings. The performance of this model for each of the DDI types is shown in Table 6.2, Table 6.3 and Table 6.4. The model achieves an F1 of 61.98% on the DDI-DrugBank dataset, while its F1 on the DDI-MedLine dataset is lower (43.21%). The differences between the results are mainly because the DDI-MedLine dataset (with 327 positive instances) is much smaller than the DDI-DrugBank dataset (with 4701 positive instances).

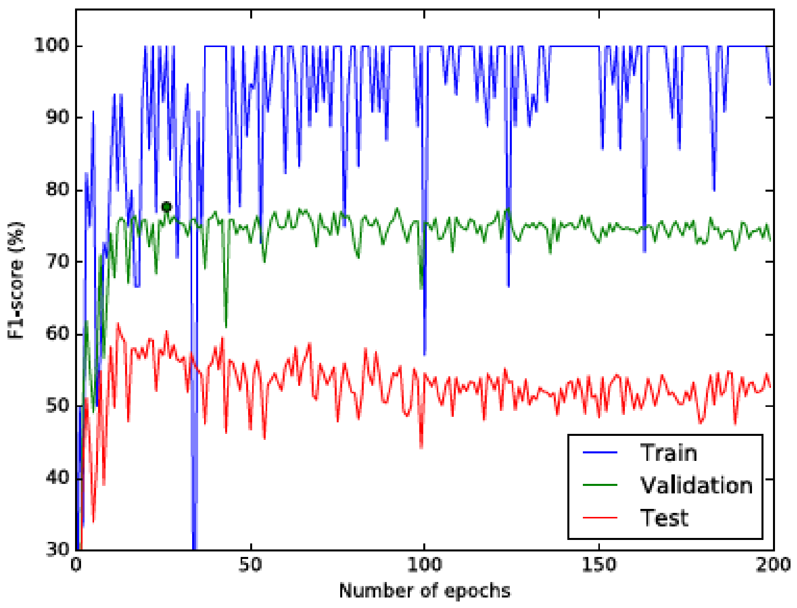


Figure 6.2: Learning curve of a CNN with random initialization. The blue line shows the training-curve variation along the number of epochs, the green represents the validation and the red one the testing curve.

Table 6.2: Results obtained for CNN from random initialization on the whole DDI corpus.

Classes	TP	FP	FN	Total	P	R	F1
Advice	131	43	90	221	75.29%	59.28%	66.33%
Effect	239	220	121	360	52.07%	66.39%	58.36%
Int	27	3	69	96	90%	28.12%	42.86%
Mechanism	176	84	122	298	67.69 %	59.06%	63.08%
Overall	573	350	402	975	62.08%	58.77%	60.38%

Focussing on the results obtained for each DDI type on the whole DDI corpus, the *advice* class is the type with the best F1. The main reason of this result is that these interactions are typically described by very similar patterns such as *'DRUG should not be used in combination with DRUG'* or *'Caution should be observed when DRUG is administered with DRUG'*. Thus, the model can easily learn these patterns because they are very common in the DDI corpus, like in the DDI-DrugBank dataset. The *mechanism* type is the second one with the best performance (F1 = 63%), even though its number of instances is lower than the *effect* type (see Table 6.1). While the systems which were involved in the DDIExtraction-2013 challenge agreed that the second easiest type was *effect* (Segura-Bedmar et al., 2014), this may have been because it was the second type with more examples in the DDI corpus; the model appears to obtain better performance for the *mechanism* type. As described in (Herrero-Zazo et al., 2013), one of the most common reasons for disagreement between the annotators of the DDI corpus is the DDIs that represent *mechanism* and *effect* because sometimes the selection of these types is not obvious. For example, the sentence *'Concomitant administration of TENTRAL and theophylline-containing drugs leads to increased theophylline levels and theophylline toxicity in some individuals'* describes a change in the *mechanism* of the DDI (*'increased theophylline levels'*), as well as a *effect* (*'theophylline toxicity'*). In order to solve these cases, the annotators defined the following priority rule: first *mechanism*, second *effect* and third *advice*. While the systems developed so far cannot learn this rule, the CNN model appears to have acquired it correctly. Moreover, the sentences that describe the type *mechanism* include the PK parameters such as area under the curve (AUC) of blood concentration-time, clearance, maximum blood concentration (Cmax) and minimum blood concentration (Cmin). These kinds of parameters, which in general are expressed by a small vocabulary of technical words from the pharmacological domain, may be easily captured by the CNN model because the word vectors are fine-tuned for the training.

Finally, the results show that the *int* class is the most difficult type to classify because the proportion of instances in the DDI corpus (5.6%) is much smaller than those of the remainder of the types (41.1% for *effect*, 32.3% for *mechanism* and 20.9% for *advice*).

Table 6.3 and Table 6.4 also show that the performance of each type is different depending of the dataset. Thus, while the above explanation can be extrapolated to the DDI-DrugBank dataset, the conclusions are entirely different for the DDI-MedLine dataset. For example, the CNN model obtains lower results for the advice type (F1 = 25%) compared to the *effect* and *mechanism* types (with an F1 around 43-45%). This difference could be caused because the *advice* type is very scarce in the DDI-MedLine dataset. Likewise, the CNN model cannot classify the *int* type, which is even scarcer than the *advice* type in this dataset.

Table 6.3: Results obtained for CNN from random initialization on the DDI-DrugBank dataset.

Classes	TP	FP	FN	Total	P	R	F1
Advice	130	43	84	214	75.14%	60.75%	67.18%
Effect	212	190	86	298	52.74%	71.14%	60.57%
Int	27	2	67	94	93.1%	28.72%	43.9%
Mechanism	169	79	109	278	68.15%	60.79%	64.26%
Overall	538	314	346	884	63.15%	60.86%	61.98%

Table 6.4: Results obtained for CNN from random initialization on the DDI-MedLine dataset.

Classes	TP	FP	FN	Total	P	R	F1
Advice	1	0	6	7	100%	14.29%	25%
Effect	27	30	35	62	47.37%	43.55%	45.38%
Int	0	1	2	2	0%	0%	0%
Mechanism	7	5	13	20	58.33%	35%	43.75%
Overall	35	36	56	91	49.3%	38.46%	43.21%

Filter-size selection

Figure 6.3 shows the distances between entities in the DDI corpus, which were obtained from more than 100 samples. Concretely, the most common distances are 2, 4 and 6, with 3205, 1858 and 1586 samples, respectively. Because biomedical sentences describing DDIs are usually very long and their interacting drugs are often far from each other (the average distance between entities is 14.6), different window sizes are used to adapt this parameter to biomedical sentences.

Table 6.5 shows the results of the CNN baseline trained with different filter sizes. With the excepting of some cases (e.g. filter size = 2), most of the filter sizes provide very close results. In the case of single filter size, 14 is the best one because it can capture long dependencies in a sentence with just one window. Although it seems logical to consider that larger filter sizes should give better performance, the experiments did not agree with this conclusion. Increasing the size appears to create incorrect filter weights, which cannot capture the most common cases. Concretely, the best filter size was (2, 4, 6), which may be because they are the most common distances between entities in the DDI corpus.

Table 6.6 shows the significance tests for the experiments assessing the effect of the filter size parameter. In general, most of the comparisons are statistically significant, and especially those with the filter-size (2,4,6) that achieves the best performance. Therefore, the best performance

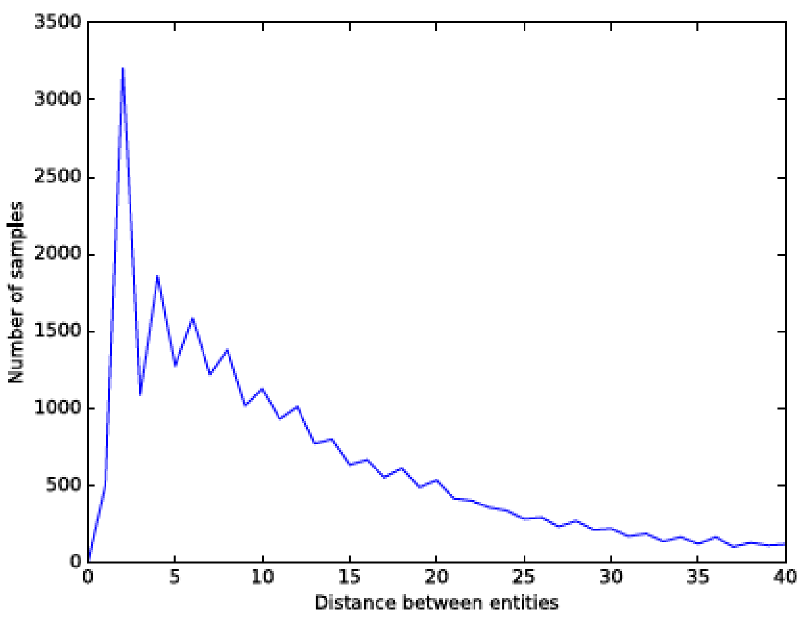


Figure 6.3: Distance between entities in sentences describing DDIs.

Table 6.5: Results for several filter sizes.

Filter Size	**P**	**R**	**F1**
2	56.89%	52.1%	54.39%
4	65.65%	52.92%	58.6%
6	**75.35%**	49.23%	59.55%
(2, 3, 4)	63.15%	57.13%	59.99%
(3, 4, 5)	62.08%	**58.77%**	60.38%
(2, 4, 6)	73.57%	52.82%	**61.49%**
(2, 3, 4, 5)	71.31%	52%	60.14%
14	71.23%	51.79%	59.98%
(13, 14, 15)	72.64%	49.03%	58.54%

is obtained using a filter-size of (2, 4, 6). Thus, the most common distances between entities are the best choice to be used as the filter size parameter.

Effects of the embeddings

Table 6.7 shows the results for the different word embeddings as well as for several dimensions (5, 10) of position embeddings with a filter size (3, 4, 5). As previously explained, position embeddings represent the position of the candidate entities (which are involved in the DDI) as a vector. Only the word embeddings are the input matrix when the position embeddings are

Table 6.6: χ^2 and p-value statistics between the different filter sizes. Asterisk denotes results statistically significant.

Filter Size	4	6	(2, 3, 4)	(3, 4, 5)	(2, 4, 6)	(2, 3, 4, 5)	14	(13, 14, 15)
2	13.22*	50.68*	155.88*	1.20	25.77*	119.71*	44.52*	5.28*
	2.77e-04*	1.09e-12*	8.99e-36*	2.73e-01	3.84e-07*	7.32e-28*	2.52e-11*	2.16e-02*
4		20.01*	118.53*	21.92*	3.87*	97.79*	17.79*	0.14
		7.71e-06*	1.33e-27*	2.84e-06*	4.91e-02*	4.66e-23*	2.46e-05*	7.12e-01
6			44.14*	73.39*	7.92*	26.78*	0.08	22.25*
			3.06e-11*	1.06e-17*	4.89e-03*	2.28e-07*	7.73e-01	2.39e-06*
(2, 3, 4)				177.61*	69.08*	4.82*	33.78*	81.04*
				1.61e-40*	9.48e-17*	2.81e-02*	6.18e-09*	2.21e-19*
(3, 4, 5)					33.25*	146.78*	67.70*	11.33*
					8.12e-09*	8.75e-34*	1.91e-16*	7.65e-04*
(2, 4, 6)						51.97*	7.16*	5.06*
						5.63e-13*	7.45e-03*	2.45e-02*
(2, 3, 4, 5)							20.44*	67.10*
							6.16e-06*	2.58e-16*
14								31.01*
								2.57e-08*

not included in the model.

In general, the implementation of position embeddings appears to realize a slight improvement in the results, providing the best scores when the dimension is 10. For example, the inclusion of position embeddings achieves a slight increase in F1 for random initialization (i.e. the word vectors are randomly initialized and fine-tuned for the training). In this case, the position embedding with a dimension of 5 achieves the best F1. On the contrary, the CNN model which was trained using a word embedding model on Wikipedia (with a default setting in the C version of Word2vec, which is represented as Wiki_bow_8w_25n in Table 6.7) appears to benefit from the implementation of position embeddings, achieving its best F1 (60.91%) with a dimension of 10 for the position embedding. Likewise, the CNN models, which were trained on the word embedding model from the BioASQ collection with skip-gram architecture (Bio_skip_8w_25n and Bio_skip_10w_10n), also provide better results when the dimension is 10. If the architectures are CBOW (Bio_bow_8w_25n and Bio_bow_5w_10n), the best F1 are obtained with dimension 5.

The results of training the CNN models on pre-trained word embeddings model from Wikipedia are slightly lower than the random initialization because the word embeddings learned from Wikipedia, which contains texts from different domains, may not be appropriate for the pharmacological domain. Neither of the word embedding models learned from the BioASQ collection (which focuses on the biomedical scientific domain) appears to provide better results than the CNN model initialized with random vectors. A possible reason for this may be that most texts in the DDI corpus are not scientific texts but also fragments from health documents for patients, such as drug package inserts (which contain information about a given medication).

Furthermore, the effects of the Word2vec parameters on the CNN performance are also explored. In Table 6.7, the two architectures (skip-gram and CBOW) provide very similar scores. However, the former has a very high computational complexity with a very long generation time compared to the latter, therefore, CBOW appears to be the best option to train the word embedding models instead of skip-gram. For the CBOW architecture, the best F1 is 60.91% (window size 8 and negative sampling 25, trained on Wikipedia). When the same model trained on BioASQ, the model obtains a very close F1 (60.78%).

Table 6.7: Performance with different word embedding and position embedding size. The prefix Wiki (Wikipedia corpus) or Bio (BioASQ dataset) refers to the corpus used to train the word embedding model. The label bow (CBOW) or skip (skip-gram) refers to the type of architecture used to build the model. The preceding number w and n indicates the size of the context window and the negative sampling, respectively.

Word Embedding	Position Embedding	P	R	F1
random	0	62.08%	58.77%	60.38%
	5	69.34%	55.9%	**61.9%**
	10	**70.76%**	54.36%	61.48%
Wiki_bow_8w_25n	0	60.89%	54.46%	57.5%
	5	59.2%	60.72%	59.95%
	10	70.64%	53.54%	60.91%
Bio_skip_8w_25n	0	62.39%	57.85%	60.03%
	5	67.8%	53.33%	59.7%
	10	66.92%	55.18%	60.48%
Bio_skip_10w_10n	0	70.66%	49.64%	58.31%
	5	61.84%	56.51%	59.06%
	10	68.77%	54.87%	61.04%
Bio_bow_8w_25n	0	64.09%	54.36%	58.82%
	5	69.43%	54.05%	60.78%
	10	67.27%	49.95%	57.33%
Bio_bow_5w_10n	0	58.25%	59.38%	58.81%
	5	60.18%	**61.23%**	60.7%
	10	65.21%	56.72%	60.67%

The significance tests for the different word embeddings and position embeddings indicate that

Table 6.8: χ^2 and p-value statistics between the different word embeddings and position embeddings. Asterisk denotes results statistically significant.

Word Embedding and Postion Embedding		random		Wiki_bow_8w_25n			Bio_skip_8w_25n			Bio_skip_10w_10n			Bio_bow_8w_25n			Bio_bow_5w_10n		
		5	10	0	5	10	0	5	10	0	5	10	0	5	10	0	5	10
random	0	78.41* 8.38e-19*	11.44* 7.19e-04*	0.74 3.89e-01	11.41* 7.31e-04*	0.44 5.08e-01	10.10* 1.48e-03*	0.14 7.06e-01	20.98* 4.64e-06*	133.47* 7.13e-31*	0.26 6.08e-01	0.02 8.83e-01	127.06* 1.80e-29*	71.74* 2.46e-17*	10.71* 1.06e-03*	5.89* 1.52e-02*	132.98* 9.12e-31*	8.32* 3.93e-03*
	5		41.42* 1.23e-10*	43.25* 4.81e-11*	29.50* 5.59e-08*	78.75* 7.05e-19*	23.52* 1.23e-06*	46.94* 7.31e-12*	13.26* 2.71e-04*	23.89* 1.02e-06*	49.09* 2.44e-12*	40.16* 2.34e-10*	20.05* 7.56e-06*	1.48 2.24e-01	20.31* 6.60e-06*	29.08* 6.94e-08*	23.86* 1.04e-06*	23.58* 1.20e-06*
	10			2.96 8.52e-02	0.21 6.45e-01	15.52* 8.16e-05*	0.26 6.08e-01	8.78* 3.05e-03*	4.07* 4.37e-02*	92.25* 7.64e-22*	4.61* 3.18e-02*	6.76* 9.32e-03*	85.78* 2.01e-20*	40.50* 1.97e-10*	0.39 5.33e-01	0.03 8.63e-01	92.07* 8.36e-22*	0.03 8.56e-01
Wiki_bow_8w_25n	0				6.20* 1.28e-02*	3.09 7.90e-02	7.15* 7.51e-03*	1.72 1.90e-01	15.94* 6.54e-05*	124.17* 7.74e-29*	0.10 7.54e-01	1.13 2.88e-01	91.84* 9.41e-22*	46.65* 8.47e-12*	3.90* 4.83e-02*	1.57 2.10e-01	100.12* 1.44e-23*	2.78 9.52e-02
	5					21.54* 3.46e-06*	0.00 1.00e+00	9.43* 2.13e-03*	2.23 1.36e-01	90.06* 2.31e-21*	7.56* 5.96e-03*	7.51* 6.13e-03*	66.94* 2.80e-16*	30.24* 3.81e-08*	0.01 9.41e-01	0.43 5.12e-01	76.09* 2.71e-18*	0.02 8.76e-01
	10						17.78* 2.48e-05*	0.01 9.34e-01	42.24* 8.07e-11*	144.40* 2.91e-33*	2.15 1.42e-01	0.10 7.55e-01	117.05* 2.79e-27*	76.26* 2.49e-18*	12.84* 3.39e-04*	8.05* 4.55e-03*	128.69* 7.93e-30*	12.34* 4.44e-04*
Bio_skip_8w_25n	0							14.78* 1.21e-04*	2.16 1.41e-01	111.21* 5.33e-26*	11.72* 6.18e-04*	11.36* 7.51e-04*	61.25* 5.02e-15*	27.80* 1.35e-07*	0.00 1.00e+00	0.49 4.82e-01	69.84* 6.43e-17*	0.05 8.18e-01
	5								24.49* 7.47e-07*	139.36* 3.67e-32*	1.25 2.63e-01	0.07 7.87e-01	101.05* 8.98e-24*	63.22* 1.85e-15*	9.74* 1.81e-03*	5.81* 1.59e-02*	102.92* 3.49e-24*	8.84* 2.95e-03*
	10									70.92* 3.72e-17*	26.30* 2.93e-07*	20.83* 5.03e-06*	52.75* 3.79e-13*	19.67* 9.19e-06*	1.06 3.03e-01	3.63 5.65e-02	55.44* 9.63e-14*	2.31 1.28e-01
Bio_skip_10w_10n	0										131.16* 2.28e-30*	130.70* 2.88e-30*	0.12 7.25e-01	9.66* 1.88e-03*	61.23* 5.09e-15*	90.72* 1.66e-21*	0.00 1.00e+00	71.64* 2.58e-17*
	5											0.69 4.07e-01	93.74* 3.60e-22*	53.76* 2.26e-13*	5.60* 1.80e-02*	2.69 1.01e-01	100.91* 9.61e-24*	4.50* 3.39e-02*
	10												95.72* 1.33e-22*	55.37* 9.96e-14*	7.98* 4.72e-03*	4.37* 3.66e-02*	95.76* 1.30e-22*	7.06* 7.90e-03*
Bio_bow_8w_25n	0													15.56* 8.01e-05*	89.27* 3.44e-21*	88.45* 5.22e-21*	0.11 7.43e-01	82.26* 1.19e-19*
	5														36.01* 1.97e-09*	45.52* 1.51e-11*	12.88* 3.33e-04*	47.91* 4.47e-12*
	10															0.71 4.00e-01	90.31* 2.04e-21*	0.15 6.99e-01
Bio_bow_5w_10n	0																96.01* 1.15e-22*	0.20 6.56e-01
	5																	95.07* 1.84e-22*

many of the comparisons are significant. In particular, the best model (whose word vectors were randomly initialized and the position embedding was set to 5) is statistically significant compared to the remainder models (see Table 6.8).

Optimal parameter performance

From the observation of the results over the validation set, the best model is obtained with randomly initialized, filter size (2, 4, 6) and dimension of position embedding 5. Table 6.9 shows the results of this model for each type. The type with the best F1 is *advice* (71.25%), followed by *mechanism* (58.65%) and *effect* (58.65%). The worst type appears to be *int*, which has an F1 of only 41.22%. The overall F1 is 62.23%. Figure 6.4 shows that this model does not achieve a new state-of-the-art F1 for DDI classification. However, this model is very promising, and its results are comparable to those of previous systems.

Finally, the statistical significance analysis between the baseline system and the model gives the optimal parameter values with the χ^2 and p-value statistics being 5.7 y 0.017, respectively. These results suggest that the two models produce different levels of performance.

Table 6.9: Results obtained for the best CNN model (random initialization, filter size (2, 4, 6) and position embedding of dimension 5) on the DDI corpus test set.

Classes	**TP**	**FP**	**FN**	**Total**	**P**	**R**	**F1**
Advice	145	41	76	221	77.96%	65.61%	71.25%
Effect	183	81	177	360	69.32%	50.83%	58.65%
Int	27	8	69	96	77.14%	28.12%	41.22%
Mechanism	192	106	106	298	64.43%	64.43%	64.43%
Overall	547	236	428	975	69.86%	56.1%	62.23%

Different pooling layers are explored with the provided parameters in order to find the best operation for the DDIExtraction Task. Besides, a combination of the two better pooling operations is performed to compare its performance.

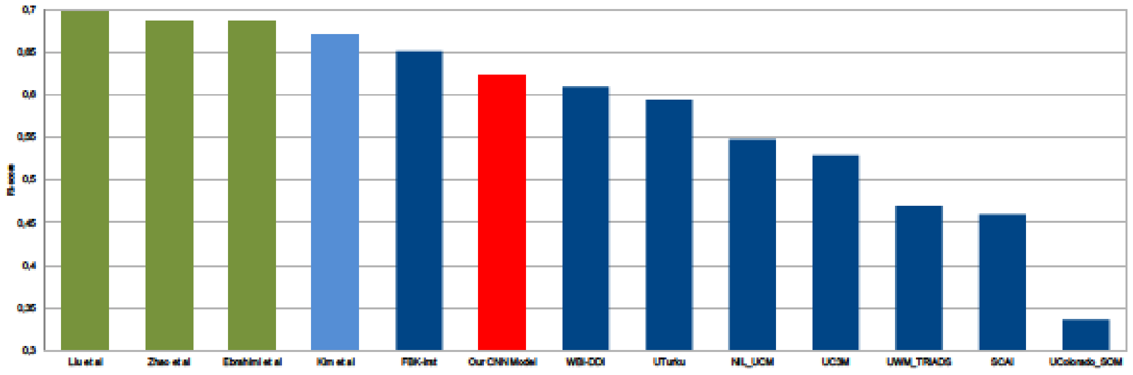


Figure 6.4: State-of-the-art F1-scores for DDI classification. The deep blue bars represent the participating system in DDIExtraction Task. The light blue is the work of (Kim et al., 2015), and the green bars represent recent systems based on the Deep Learning techniques for DDI classification, which were presented subsequently. The best model is represented by the red bar.

Max-pooling

Firstly, the effects of negative filtering are tested on the max-pooling operation architecture. The results using the max-pooling CNN without negative filtering are very similar to (Suárez-Paniagua and Segura-Bedmar, 2016) because the same configuration is used (see Table 6.10). However, a higher F1 (62.93%) is obtained using the entire training dataset.

Table 6.10: Results obtained for max-pooling CNN on the test dataset without negative filtering.

Classes	**P**	**R**	**F1**
Advice	79.33%	64.25%	71.00%
Effect	68.90%	54.17%	60.65%
Int	81.08%	31.25%	45.11%
Mechanism	58.29%	70.57%	63.84%
Overall	67.13%	59.22%	62.93%

Table 6.11 shows the results with the negative filtering, which increases the overall performance in F1 over the previous experiment (+1.63%). Negative filtering also improves the performance for three of the four DDI classes, except for the advice type, where F1 is slightly lower and not significant. Negative filtering is applied for the remaining experiments.

Table 6.11: Results obtained for max-pooling CNN on the test dataset with negative filtering.

Classes	P	R	F1
Advice	80.36%	61.09%	69.41%
Effect	62.06%	64.15%	63.09%
Int	62.32%	44.79%	52.12%
Mechanism	67.24%	66.11%	66.67%
Overall	67.19%	62.14%	64.56%

Average-pooling

As can be seen in Table 6.12, all performance measures present a drastic decrease, especially in Recall, when average pooling layer is used instead of the max-pooling operation. A possible reason may be the padding operation. In shorter sentences, the average-pooling may disrupt the representation caused by the appending of the special pad tokens.

Table 6.12: Results obtained for average-pooling CNN on the test dataset with negative filtering.

Classes	P	R	F1
Advice	66.99%	63.35%	65.12%
Effect	58.14%	63.03%	60.48%
Int	66.67%	31.25%	42.55%
Mechanism	61.90%	47.99%	54.06%
Overall	61.70%	55.35%	58.35%

Attentive pooling

Similarly to the average-pooling results, the attentive pooling layer (see Table 6.13) does not achieve better results than those obtained with max-pooling. Even so, the results are better than the average-pooling results. In this case, the adverse effect of padding on the results may be much lower than in the average-pooling because the weight of PAD tokens is possibly much smaller than the rest of the tokens.

Pooling combination

Similarly to (Sahu and Anand, 2018), two separately CNN models are trained using two different pooling operations (in particular, max-pooling and attentive) to increase the performance of the

Table 6.13: Results obtained for attentive pooling CNN on the test dataset with negative filtering.

Classes	P	R	F1
Advice	78.74%	61.99%	69.37%
Effect	58.29%	57.14%	57.71%
Int	79.07%	35.42%	48.92%
Mechanism	60.75%	54.03%	57.19%
Overall	64.42%	55.14%	59.42%

model. Then, the two pooling vectors are concatenated into a single vector, that is the input of the Softmax layer for the final classification of the instances. Table 6.14 shows the results of this combination. Neither the combination of the max and attentive pooling operations overcomes the use of a single max-pooling layer.

Table 6.14: Results obtained for the combination of max-pooling and attentive pooling CNN on the test dataset corpus with negative filtering.

Classes	P	R	F1
Advice	79.23%	65.61%	71.78%
Effect	65.28%	61.62%	63.40%
Int	80.49%	34.38%	48.18%
Mechanism	69.23%	60.40%	64.52%
Overall	70.40%	59.47%	64.47%

6.3.2 SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications

Each pair of keyphrases represents a possible relation instance. The CNN model classifies each of the instances in three classes HYPONYM-OF, SYNONYM-OF and NONE. The corpus is annotated in the paragraph level, that is why the sentence splitter of the NLTK[7] separates the relations in the sentence level because the relations are annotated within a sentence. Once each instance is obtained, following a similar approach previously described the sentences are cleaned and preprocessed using the entity blinding.

For the particular case of the HYPONYM-OF class, the directionality must be considered. For instance, if an entity A is HYPONYM-OF B, the relation of B with A is annotated as

[7]http://www.nltk.org

NONE. For the remainder classes, both directions are annotated with the same class label. The keyphrases corpus contains some annotated entities that are overlapped. As this kind of mentions produces wrong entity blinding, they are removed. The classification of keyphrases involving overlapped entities is a challenging task which will be tackled in future work. One possible solution will consider different instances for each overlapped entities.

Firtsly, a basic CNN system with predefined parameters classifies the sentences into the relationships categories. Secondly, the effects of the position embeddings are observed assigning a dimension $M_d = 10$. Besides, the parameters are the same as in (Kim, 2014): word embeddings dimension $M_e = 300$, filters $m = 200$ with a window size $w = (3, 4, 5)$. For the training phase, the hyper-parameters are set as follows: dropout rate $p = 50\%$, mini-batch size of size 50 and the Rectified Linear Unit (ReLU) as the nonlinear function g. The parameter $n = 95$ which is determined by the maximum length sentence in the dataset. Only the number of epochs was fine-tuned in the validation set using the stopping criteria.

Table 6.15 shows the results of the basic CNN configuration without position embeddings. The Recall performance in both classes is very low because the parameters were not explored in detail and the model is not fine-tuned for this problem correctly. In addition, removing the overlapped entities discards many examples, and improves the results.

Table 6.15: Results over the SemEval 2017 Task 10 dataset using a basic CNN.

Classes	**P**	**R**	**F1**
HYPONYM-OF	0.27	0.07	0.12
SYNONYM-OF	0.65	0.32	0.43
Overall	0.53	0.21	0.30

Table 6.16 shows the official results obtained by the CNN with position embeddings. The Precision increases considerably in the case of HYPONYM-OF and the Recall of SYNONYM-OF. This result proves that sentences are best represented using position embeddings (+8% in F1-score against to CNN without position embeddings).

For both cases, the class SYNONYM-OF is classified better than the class HYPONYM-OF because the examples in the former class are simpler, e.g. in the sentence *'trajectory surface hoping (TSH)'* the keyphrase *'trajectory surface hoping'* is a SYNONYM-OF *'TSH'*, and the double of instances are obtained due to the class is the same in both directions. That is the main reason why the class HYPONYM-OF obtained low Recall in both models. For this reason, some preprocessing are needed to correct these classification errors with a rule-based system.

Table 6.16: Results over the SemEval 2017 Task 10 dataset using a CNN with position embedding size of 10.

Classes	P	R	F1
HYPONYM-OF	0.54	0.07	0.13
SYNONYM-OF	0.61	0.46	0.52
Overall	0.60	0.28	0.38

6.3.3 SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers

The relations between scientific concepts are annotated pair by pair in the abstracts. All the annotations span within one sentence, but the corpus is given in the paragraph level. For this reason, the NLTK tool[8] splits the paragraphs of the abstracts into sentences to generate all the possible instances. Then, the instances are tokenized, cleaned and preprocessed with entity blinding.

For the *COMPARISON* class, which is a bidirectional relationship, both instances are annotated with the same class label. For example, the sentence: *'We suggest a method that mimics the behaviour of the oracle using a neural network or a decision tree.'* should be transformed into the relation instances showed in Table 6.17.

Table 6.18 shows the number of instances extracted in the training set per each class. The *None* class represents the number of pairs of entities that are not related (negative instances). The number of positive instances is very low compared to the negative ones, 1323 over 19210 (around 7%), mainly because most classes are unidirectional and we annotated the reverse instance as *None*.

A sampling technique described in (Wang et al., 2017) adjusts the same numbers of instances

[8]http://www.nltk.org

Table 6.17: Instances of a sentence in the corpus after applying the pre-processing phase with entity blinding.

(*entity1*, *entity2*)	**Instances after entity blinding**
(*oracle*, *neural network*)	*'We suggest a method that mimics the behaviour of the entity1 using a entity2 or a entity0.'*
(*neural network*, *oracle*)	*'We suggest a method that mimics the behaviour of the entity2 using a entity1 or a entity0.'*
(*oracle*, *decision tree*)	*'We suggest a method that mimics the behaviour of the entity1 using a entity0 or a entity2.'*
(*decision tree*, *oracle*)	*'We suggest a method that mimics the behaviour of the entity2 using a entity0 or a entity1.'*
(*neural network*, *decision tree*)	*'We suggest a method that mimics the behaviour of the entity0 using a entity1 or a entity2.'*
(*decision tree*, *neural network*)	*'We suggest a method that mimics the behaviour of the entity0 using a entity2 or a entity1.'*

per each class. Therefore, 60% of the negative instances are randomly discarded, and the instances in each class are duplicated until having the same number as the more representative class, 483 corresponding to *USAGE*. Thus, this technique solves possible issues associated with the imbalanced dataset.

Table 6.18: Number of instances in the SemEval 2018 Task 7 dataset.

Classes	**Instances**
COMPARE	190
MODEL-FEATURE	326
PART WHOLE	234
RESULT	72
TOPIC	18
USAGE	483
None	17887
Total	19210

The proposed CNN is tested for the classification of these sentences. Additionally, the POS tag features of each word (entities are marked as common nouns) are extracted in order to create a POS embedding matrix as (Zhao et al., 2016) $\mathbf{W}_{POS} \in \mathbb{R}^{|P| \times m_{POS}}$ where P is the POS types vocabulary size and m_{POS} is the POS embedding dimension. Finally, the input matrix $\mathbf{X} \in \mathbb{R}^{n \times (m_e + m_{POS} + 2m_d)}$ represents the concatenation of the word embeddings, the POS embeddings and the two position embeddings for each word in the instance.

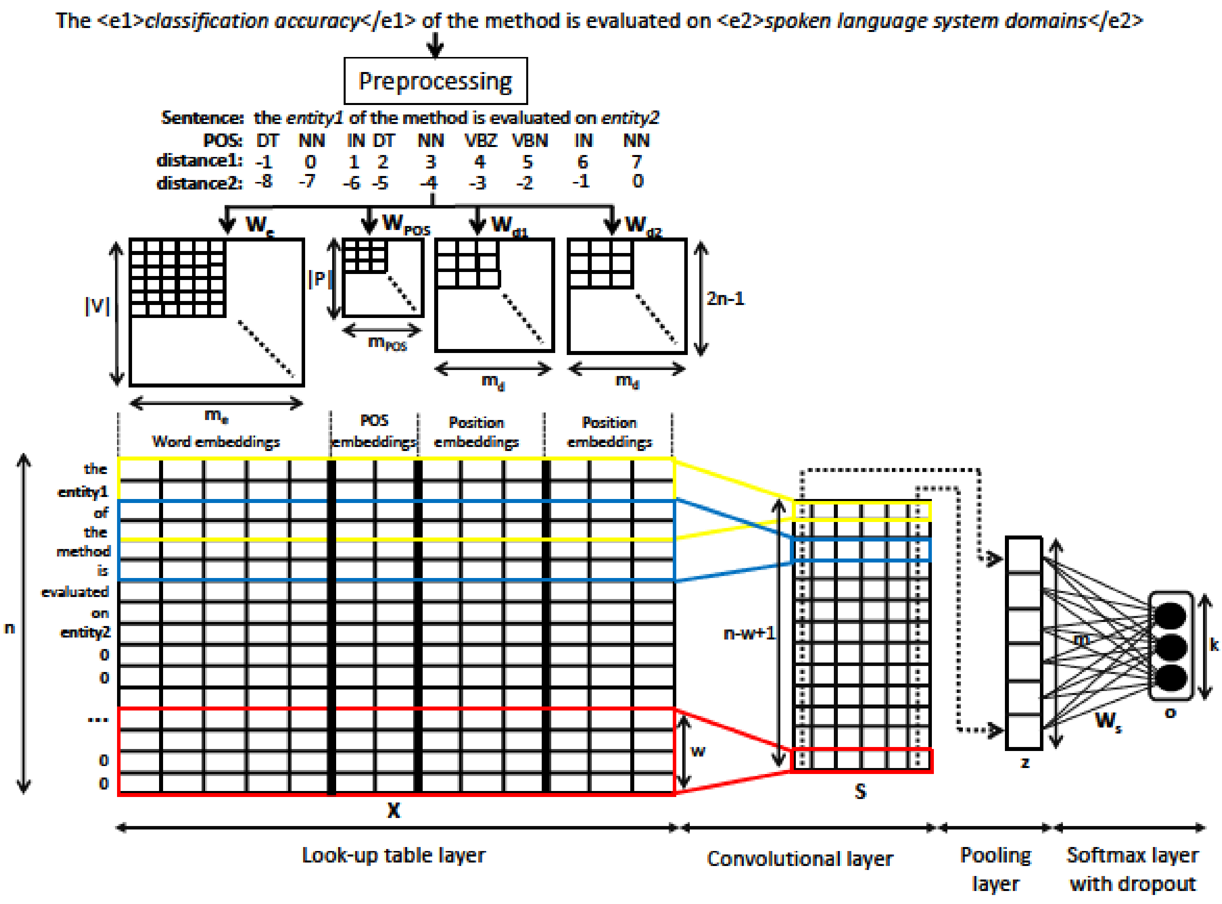

The <e1>classification accuracy</e1> of the method is evaluated on <e2>spoken language system domains</e2>
Preprocessing
Sentence: the entity1 of the method is evaluated on entity2
POS: DT NN IN DT NN VBZ VBN IN NN
distance1: -1 0 1 2 3 4 5 6 7
distance2: -8 -7 -6 -5 -4 -3 -2 -1 0
W_e
W_POS
W_d1
W_d2
|V|
|P|
2n-1
m_e
m_POS
m_d
Word embeddings
POS embeddings
Position embeddings
the
entity1
of
the
method
is
evaluated
on
entity2
0
n
w
X
n-w+1
S
z
m
W_s
o
k
Look-up table layer
Convolutional layer
Pooling layer
Softmax layer with dropout

Figure 6.5: CNN model for the semantic relation classification in scientific papers of SemEval-2018 Task 7.

Word table layer

For the experiments, the CNN uses $M_d = 5$ for the position embeddings and $M_POS = 10$ for the POS tag embeddings. Moreover, the parameters are the same as in (Kim, 2014): word embeddings dimension $M_e = 300$, filters $m = 200$ with a window size $w = (3, 4, 5)$. For the training phase the hyper-parameters are set as follows: dropout rate $p = 50\%$, mini-batch size of size 50 and the Rectified Linear Unit (ReLU) as the nonlinear function g. The parameter $n = 95$ which is determined by the maximum length sentence in the dataset. Only the number of epochs was fine-tuned in the validation set using the stopping criteria.

The CNN system obtains an F1-score of 35.4% for the relation extraction task in which only the detection of relation is considered. Table 6.19 shows the official results obtained for the relation classification task. This model reaches an F1-score in Macro-average of 18.5% with one step classifier, which means that the extraction and classification are considered at the same time. This performance was expected because the validation set, which was created from the training set, reaches similar results. Furthermore, 147 instances with correct directionality over 367 are correctly predicted (i.e. 40.05% in coverage).

The main problem is the high number of FP in the majority of classes, which are the *None* instances classified as a class. In some classes, such as *PART WHOLE* and *USAGE*, the number of FN compared is higher than the total number of instances. The main reason is that the representations of the two directions of each relation are very similar, only the position distances and the target entity names are inverted, and the CNN cannot distinguish between them.

Table 6.19: Results over the SemEval 2018 Task 7 dataset using a CNN model measured by True Positives, False Positives, False Negatives, Precision, Recall and F1-measure, respectively.

Classes	TP	FP	FN	P	R	F1
COMPARE	8	116	11	6.45%	42.11%	11.19%
MODEL-FEATURE	36	185	37	16.29%	49.32%	24.49%
PART WHOLE	22	66	60	25%	26.83%	25.88%
RESULT	2	21	14	8.7%	12.5%	10.26%
TOPIC	0	0	3	0%	0%	0%
USAGE	41	96	133	29.93%	23.56%	26.37%
Micro-averaged	-	-	-	18.38%	29.7%	22.71%
Macro-averaged	-	-	-	14.39%	25.72%	18.46%

6.4 Conclusions

State-of-the-art methods for the DDIExtraction Task use classical supervised machine-learning algorithms (such as SVM) and intensive feature-engineering. CNN model can be used to classify DDIs and automatically learns the best feature representation. The main contributions of this section are: to make a detailed comparison of previous work for the DDIExtraction Task; to provide an in-depth study of the influence of the CNN hyper-parameters on the results; and to evaluate the performance of a CNN model for different types of texts such as scientific articles and drug package leaflets as well as for the different type of DDIs.

Unlike some previous works based on Deep Learning (Ebrahimi and Dou, 2015; Zhao et al., 2016), this CNN model does not employ any external features in the classification layer. Their systems used external features such as the distance between the entities, the depth of the tree of the entities, the type of syntactic dependencies that links the entities or the contexts around the entities, among others. There is an extensive literature showing that these features can positively contribute to solving the relation extraction task. Consequently, the real contribution of a Deep Learning model as a feature learning model is not clear if the architecture uses external features. Despite the low performance, the system achieves very promising results without any feature-engineering. The classification of DDIs remains an unsolved challenge in scientific texts, such as MedLine abstracts, and this is primarily because the size of the training dataset is not enough to learn the features, which are more appropriate for the extraction of DDIs from MedLine abstracts. Thus, it is crucial to increase the size of the DDI-MedLine dataset. The same problem occurs with the classification of the *advice* and *int* DDI types, which have a very low frequency in the DDI corpus, and therefore, their results were worse than those obtained for the *mechanism* and *effect* types.

Comparing with previous works that did not use Deep Learning methods, an automatic feature-learning method is proposed and obtains 62.23% in F1 that is a suitable alternative for the classification task without any external information. These systems have a higher classification rate (Chowdhury and Lavelli, 2013*b*; Kim et al., 2015) using an ensemble of kernel methods with an extensive feature set built from a demanding feature-engineering task. Unlike previous works, an exhaustive and detailed study of possible settings is performed (in particular, the filter size, word and position embeddings) of the CNN architecture, and also an in-depth analysis of the results for each type of DDI and over each dataset of the DDI corpus. It is planned to study the effect of adding additional layers to this architecture and use the two-step classification (detection and classification of each DDI) as (Zhao et al., 2016). Furthermore, other Deep Learning architectures for DDI classification, e.g. recurrent neural network, and exploring its parameters without external features could be implemented as in the present work.

The experiment results showed that the random initialization of the input word vectors have a better performance than the pre-trained word embedding models because these models are learned from text collections such as Wikipedia or MedLine, which do not contain texts similar to those on drug package inserts. Most texts of the DDI-DrugBank dataset were obtained from this kind of documents. Thus, the word embedding model can be trained on an extensive collection of drug package inserts in order to study their effects on the performance for the proposed system. The perfromance of Word2vec with different parameters suggests that both architectures (i.e. skip-gram and CBOW) achieved very similar results. However, it is recommended using CBOW because it has a significantly less computational complexity compared to skip-gram. For the other Word2vec parameters, the default setting used in the C version of Word2vec appears to give the best performance. The filter size is another parameter that significantly affects the model performance. Although the early assumptions were that a large filter size would provide better results because biomedical sentences are usually very long, the experiments confirmed that the best filter was (2, 4, 6). Furthermore, the aggregation of position embeddings generally improves the results, being 10 dimensions slightly better than 5.

Additionally, three different pooling operations are compared for the task of the DDIExtraction Task using a CNN architecture. The experiments show that max-pooling exhibits a higher performance (F1=64.56%) than attentive pooling (F1=59.92%) and than average-pooling (F1=58.35%). Attentive and average pooling operations provide worse results possibly caused by the negative effect of special pad tokens that are appending to the shorter sentences. Concretely, the combination of max-pooling and attentive pooling does not improve the performance as compared with the single max-pooling technique.

The official results for the ScienceIE keyphrases Relation Classification task of SemEval 2017 show that this architecture obtained an F1-score of 38% for Keyphrases Relation Classification. This performance is promising, but the results are lower in comparison with other participant results. However, this model is a basic configuration with a generic preprocessing phase and without external features. The results suggest that the use of the position embedding improves the performance in both classes. The top system for the relation extraction subtask was a CNN with max-pooling that used the word, position, type of entity and POS tags embeddings of the words between the target entities in the sentence in order to generate the class prediction (Lee et al., 2017) and obtaining 63.8% in F-measure.

The Relation Classification task of SemEval 2018 dataset is balanced using sampling techniques, blinded the entities in the sentence and aggregated position embedding and POS embedding to the word embedding of each word to have more representation of each instance with the proposed CNN. This architecture obtains an F1-score of 35.4% for the relation extraction task and 18.5% for the relation classification task with a basic configuration of the one step CNN. The architecture of (Rotsztejn et al., 2018) ranked first in the classification subtasks using an ensemble of CNN and LSTM with the word, POS and relative position embeddings of the words in the sentence which achieves 49.3% in F1.

In future work, it is planned to avoid the pad tokens in the implementation of attentive and average pooling operations. Besides, using a multi-channel word embedding by integrating several word embeddings models can improve the performance of CNN. Moreover, a two steps model that overcomes the extraction of the relationships between two concepts and subsequently classify them in the different semantic classes is proposed because it seems to perform better than one step models for relation classification (Kim et al., 2015; Wang et al., 2017; Zhao et al., 2016). For both SemEval tasks, it is planned to rule out the reverse instances of each class as *None* in order to avoid having very similar representation with different labels. The directionality problem could be solved with some post-processing rules after the classification or defining different labels for the classes depending on the direction.

Chapter 7

End-to-end Information Extraction System in scientific and biomedical texts

This chapter presents a multilingual end-to-end system for Named Entity Recognition and Relation Extraction from biomedical and scientific texts. This system involves the entire process of classifying relations from raw data. Concretely, the NER and RE tasks applied as end-to-end is a crucial step to create a real scenario application in the clinical domain.

7.1 Introduction

The proposed system is based on two Deep Learning methods for solving the NER and RE tasks. Firstly, NER is performed combining a bidirectional RNN with LSTM cells (Bi-LSTM) and a Conditional Random Field (CRF). Secondly, RE applies a Convolutional Neural Network with the position information of the entities. The approach is domain and language independent, and easily expandable to support other languages and clinical applications that require the exploitation of semantic information from texts. This is one of the most relevant contributions of this thesis because this is the first end-to-end system for the DDIExtraction Task. Moreover, this architecture obtains state-of-the-art results on the eHealth-KD challenge, which was part of the Workshop on Semantic Analysis at SEPLN (TASS-2018) (Martínez-Cámara et al., 2018).

During the last decade, tremendous advances have been made in NLP and IE for biomedical and clinical domains due to the numerous shared tasks organized. Starting from the pioneer BioCreative (Hirschman et al., 2005) until the most recent NLP clinical Challenge (n2c2)[1]. Recently, the International Workshop on Semantic Evaluation (SemEval) organized some evaluations of computational semantic analysis systems for Relation Extraction. Concretely, the goal of SemEval 2017 Task 10: ScienceIE is the automatic extraction of keyphrases and their relationships from scientific publications (Augenstein et al., 2017). In addition, SemEval-2018 Task 7 (Gábor et al., 2018) is focused on the extraction of semantic relationships in scientific papers and defined two subtasks for detection and classification. Moreover, TAC 2017 ADR (Roberts et al., 2017) proposed the evaluation of systems for extracting adverse drug reactions from drug labels. Furthermore, the second Social Media Mining for Health Research and Applications Workshop dealt with the detection of adverse drug reactions from tweets (Sarker and Gonzalez, 2017). All these competitions have been focused on English, while eHealth-KD challenge (Martínez-Cámara et al., 2018) has been the first initiative to promote the development of information extraction techniques to automatically extract knowledge from eHealth documents written in the Spanish language. The shared task proposes the identification and classification of keyphrases, as well as the detection of all relevant semantic relationships between the recognized entities.

7.2 Method

Following the state-of-the-art techniques, the proposed system for NER and RE are based on Deep Learning. Concretely, the combinination of a bidirectional RNN with LSTM cells (Bi-LSTM) and a Conditional Random Field (CRF) and a CNN performs the extraction of the entities and the classification of the relationships, respectively. Comprehensive experimentation of the approach is provided for the different datasets, such as the DDI corpus (Herrero-Zazo

[1]https://n2c2.dbmi.hms.harvard.edu/

et al., 2013) and the dataset used in the eHealth-KD challenge. To the best of our knowledge, this is the first end-to-end system for the DDIExtraction Task. Moreover, the proposed system obtains state-of-the-art results on the eHealth-KD challenge. This approach is domain and language independent, and thereby easily expandable to support other languages and clinical applications that require the exploitation of semantic information from texts. This section describes the datasets and the proposed architecture for each module of the information extraction system independently.

7.2.1 Datasets

The experiments are calculated taken two biomedical datasets with different languages, English and Spanish. One of them is the DDI corpus which has been previously described. The other corpus is taken from the eHealth-KD challenge that involves electronic health documents.

The eHealth-KD challenge, which is part of the Workshop on Semantic Analysis at SEPLN (TASS-2018), provided to participating teams an annotated collection of MedlinePlus documents. MedlinePlus[2] is an informative website directed to patients, which offers information about health topics such as medicines and diseases. The documents were annotated with keyphrases as the entities and their semantic relationships. Two different entity types are proposed to classify the keyphrases: *Concept* and *Action*. Likewise, six types of relationships are defined: *is-a*, *part-of*, *property-of* and *same-as*, which are relationships between concepts, and *subject* and *target*, which can represent relationships between actions and concepts or between actions themselves. Table 7.1 and Table 7.2 show the statistics for the entity and relation types in the eHealth-KD dataset, respectively. The corpus was split into three subsets: a training set (559 sentences), a validation set (285 sentences) and a test set (300 sentences). The test set is divided into three different sets for each scenario.

[2]https://medlineplus.gov/spanish/

Table 7.1: Entity types in the eHealth-KD challenge datasets.

Classes	**Train**	**Validation**	**Scenario 1 Test**	**Scenario 2 Test**	**Scenario 3 Test**
Concept	2,427	849	432	439	434
Action	1,525	434	163	154	183
Total	3,952	1,283	595	593	617

Table 7.2: Relation types in the eHealth-KD challenge datasets.

Classes	**Train**	**Validation**	**Scenario 1 Test**	**Scenario 2 Test**	**Scenario 3 Test**
is-a	434	370	74	92	69
part-of	149	145	31	33	32
property-of	399	244	58	58	62
same-as	30	13	2	1	5
subject	693	339	147	117	137
target	991	504	180	195	212
Total	2,696	1,615	492	496	517

7.2.2 System overview

Figure 7.1 presents the two modules that compose the end-to-end IE system. The approach deals with three different subtasks: A) entity detection, B) entity classification and C) relation extraction. Therefore, there are three possible evaluation scenarios:

- Scenario 1: Only plain text is given (subtasks A, B and C).
- Scenario 2: Plain text with manually annotated entity boundaries are given (subtasks B and C).
- Scenario 3: Plain text with manually annotated entities and their types are given (subtask C).

7.2.3 Named Entity Recognition System

The approach for the module of NER is based on a Deep Neural Network with a bidirectional RNN with LSTM cells and a classification layer with a CRF model (see the Named Entity

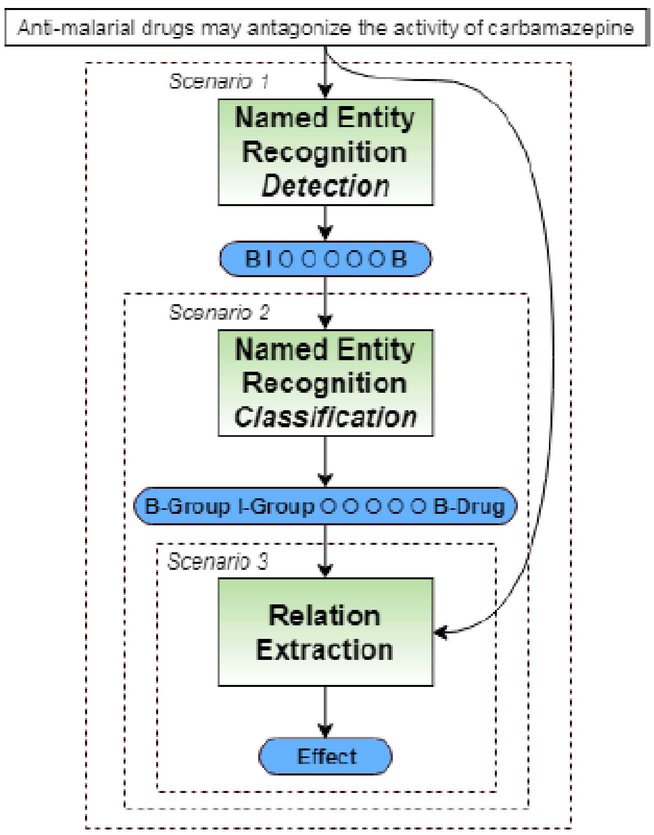

Anti-malarial drugs may antagonize the activity of carbamazepine
Scenario 1
Named Entity Recognition Detection
B I O O O O O B
Scenario 2
Named Entity Recognition Classification
B-Group I-Group O O O O O B-Drug
Scenario 3
Relation Extraction
Effect

Figure 7.1: Pipeline of the proposed end-to-end system.

Recognition module of Figure 7.2). The input for the first Bi-LSTM layer is the character embeddings of each word of the sentence. In the second layer, the output of the first layer is concatenated with word embeddings and sense embeddings, which takes into consideration the word and its POS tag. Finally, the last layer uses a CRF to obtain the most suitable labels for each token.

Pre-processing phase

Firstly, texts are preprocessed in order to create the input for this network. The sentences are split and tokenized by Spacy (Explosion AI, 2017), an open source library for advanced natural language processing with support for 26 languages. After that, each extracted token in the sentence are labelled by BILOU-V extended tag encoding with the BRAT format annotations. The B tag indicates the beginning token of an entity, the I tag indicates the inside token of an entity, the L tag indicates the last token of an entity, the U tag indicates a unit-length entity token, and the O tag represents other tokens that do not belong to any entity. BILOU tag

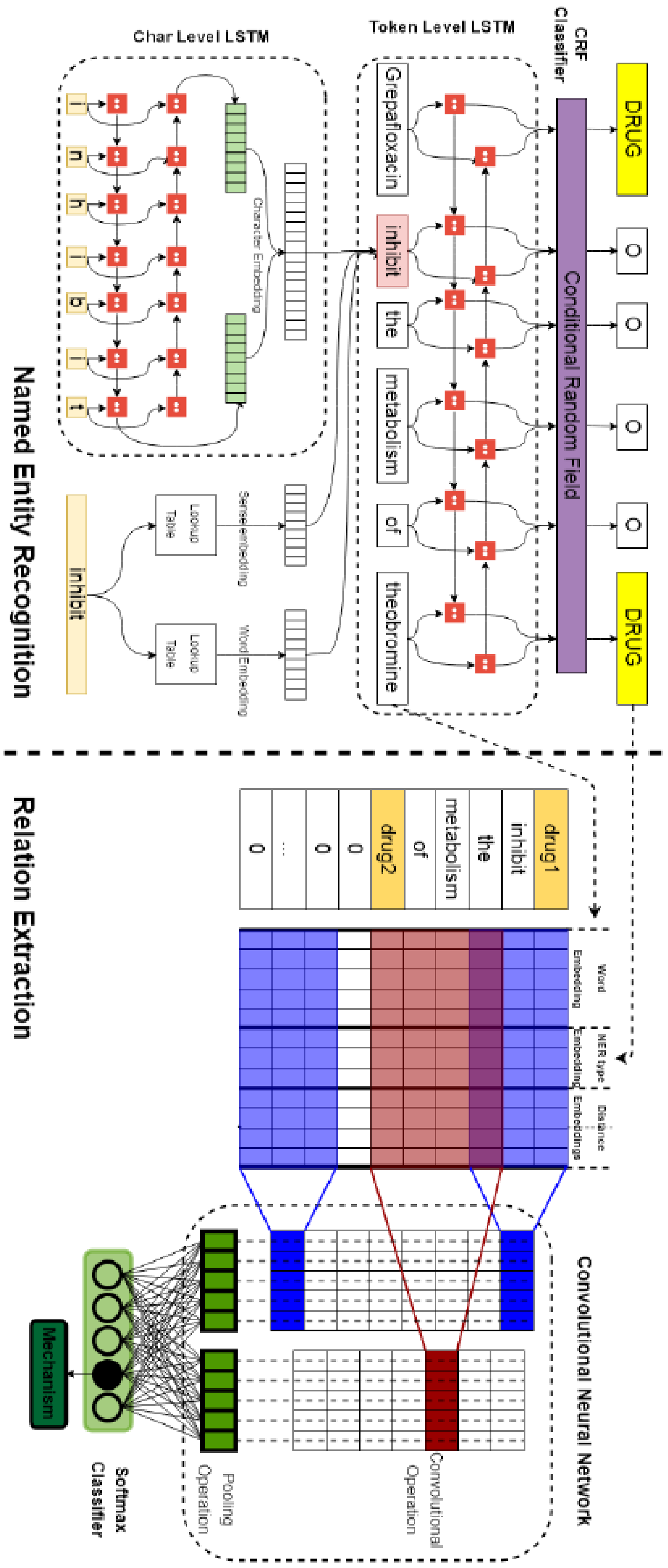


Figure 7.2: Overview of the whole end-to-end architecture.

scheme is reported to be better than BIO-encoding (Ratinov and Roth, 2009). Furthermore, it also allowed the representation of discontinuous entities, overlapping or nested entities with the V tag. Some post-processing rules will be applied to annotate token of the V tag for their surrounding token tags. Thus, the total number of classes to be predicted for the NER are the same as the tags BILU-V for each category and another class for the class O.

Bi-LSTM layer with character embeddings

Although word embedding models capture syntactic and semantic information, other linguistic information such as morphological information, orthographic transcription and POS tags are not exploited. For this reason, the character embedding representation is used in the proposed system to obtain morphological and orthographic information of the tokens.

The tokens in the sentences are divided into their characters. These characters are represented by a randomly initialized vector of 25 dimensions, which will be updated in the training step. A Bi-LSTM takes the character representation of a token and concatenates the outputs of the two directions of the network to create a word representation.

Bi-LSTM layer with using the word and sense embeddings

The output of the previous layer is concatenated together with the word embeddings and the sense embeddings of each token. The concatenation of embeddings is the input for another Bi-LSTM layer that takes all the token representations of an input sentence.

The most popular methods to create word embeddings are Word2vec (Mikolov, Chen, Corrado and Dean, 2013), the global aggregate model of word-word co-occurrence statistics (Pennington et al., 2014) and the morphological representation of fastText (Bojanowski et al., 2017). In addition, there are many pre-trained word embedding models freely available for the NLP community to use, which were trained on large collections of texts. The embedding matrix of the system uses the Spanish Billion Words (Cardellino, 2016), which is a model word embeddings model pre-trained on different text corpora written in Spanish (such Ancora Corpus

(Taulé et al., 2008), OPUS Corpus (Tiedemann and Nygaard, 2004) and Wikipedia among others). Additionally, the system uses a word embedding from GloVe model pre-trained on 2014 Wikipedia and Gigaword 5 edition corpus written in English (Parker et al., 2011). One of the most critical limitations of word embeddings models is that a single word vector represents all its possible meanings, known as polysemy. For this reason, Sense2Vec (Trask et al., 2015) is applied in order to provide multiple word vectors for each meaning of the word. The meaning of the word in this model is defined by the POS tag of a word based on its definition, its context and its relation to adjacent and related words within a phrase, sentence or paragraph. In the Sense2Vec model, each word is considered as a pair composed of the target word and its PoS tag with the context of the word. This method analyses the context of a word and then assigns its more adequate vector. In this work, the sense-disambiguation word representation vectors are generated with the sense2vec tool and trained on a collection of comments published on Reddit (corresponding to the year 2015). Table 7.3 shows the details of all three pre-trained models.

Table 7.3: Pre-trained models used in the proposed model.

Parameters	**Spanish Billion Words**	**Glove.6B**	**Reddit**
Corpus size	1.5 billion	6 trillion	2 billion
Vocabulary size	1,000,653	2 million	1 million
Array size	300	100	128
Algorithm	Skip-gram BOW	GloVe	Sense2Vec

All the embeddings are concatenated together with the character embedding output in each token of the sentence. Then, a Bi-LSTM layer computes all these embeddings in order to obtain a representation of each token in the sentence.

CRF classifier

A Conditional Random Field model takes the output vector of each token in the second Bi-LSTM layer as the input. This classification layer obtains a prediction of the tag sequence for each sentence. Finally, once tokens have been annotated with their corresponding labels in the BILOU-V encoding format, some rules are described in order to discard the wrong annotations of the tags. For instance, an I tag cannot be found before a B tag or after an L tag. Additionally, V tags, which identify nested or overlapping entities, are generated as new annotations within the scope of other mentions.

7.2.4 Relation Extraction System

The Relation Extraction module of Figure 7.2 shows the system overview used for this subtask. The main process involves a pre-processing phase for cleaning the sentences and a concatenation of the word embeddings, the position embeddings and the entity type embeddings of each token given by the previous module. Then, a CNN classifier similar to (Suárez-Paniagua and Segura-Bedmar, 2016) performs the classification of the relations.

Pre-processing phase

The first step of this module generates all the possible relation instances. A relation instance is composed by a pair of annotated entities that are in the same sentence. Given that eHealth-KD dataset contains relations in both directions, pairs are unordered if the relationship is symmetrical, while pairs are ordered if the relationship is asymmetrical. Figure 7.3 shows a sentence taken from the eHealth-KD dataset with several entities and their relationships annotated for both directions.

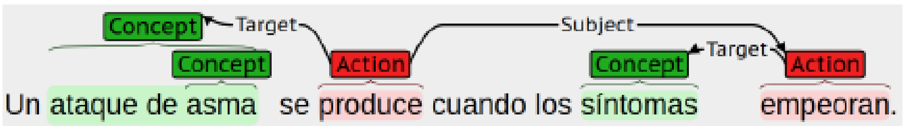


Figure 7.3: Example of sentence taken from the eHealth-KD dataset. English translation: *'An asthma attack occurs when symptoms get worse.'*.

In this example, the relations, *Target* and *Subject*, are asymmetrical. Table 7.4 shows all possible relationships instances and their relation types generated from this sentence. A *None* type is also considered for representing the non-relationship between the entities.

Once the instances with their relations are generated, sentences are tokenized and cleaned with regular expressions converting the numbers to a common name *'NUM'*, words to lower cases

Table 7.4: Possible relationships for the sentence shown in Figure 7.3.

Relation instances	**Relation type**
(ataque de asma → produce)	*None*
(ataque de asma ← produce)	*target*
(ataque de asma → síntomas)	*None*
(ataque de asma ← síntomas)	*None*
(ataque de asma → empeoran)	*None*
(ataque de asma ← empeoran)	*None*
(asma → produce)	*None*
(asma ← produce)	*None*
(asma → síntomas)	*None*
(asma ← síntomas)	*None*
(asma → empeoran)	*None*
(asma ← empeoran)	*None*
(produce → síntomas)	*None*
(produce ← síntomas)	*None*
(produce → empeoran)	*subject*
(produce ← empeoran)	*None*
(síntomas → empeoran)	*None*
(síntomas ← empeoran)	*target*

and replacing special Spanish accents to Unicode, e.g. *ñ* to *n*. In addition, the application of the entity blinding process ensures the generalization of the architecture.

Nested entities are frequent in biomedical texts. For this reason, a deoverlapping method takes each entity mention that contains a nested entity and creates a new instance without nested or overlapped entities. Moreover, all possible relation instances that involve a relationship between entities in the same nested entity are removed. To this end, each sentence is considered as a graph where the vertices are the entities and the edges are the non-overlapped entities with itself. This graph recursively obtains all the possible paths without overlapping.

Some entities can contain gaps in their mentions. For example, the noun phrase *'ganglionic or peripheral adrenergic blocking drugs'* contains two different drug entities: *ganglionic adrenergic blocking drugs* and *peripheral adrenergic blocking drugs*. The first one is a discontinuous entity because it contains a gap in its mention. In these cases, the overlapping part of the entities is removed in the sentence. In the previous example, the words *adrenergic blocking drugs* are discarded and the *ganglionic* and *peripheral* are kept as the interacting drugs.

CNN classifier

Similarly to (Suárez-Paniagua et al., 2017), randomly initialized word embeddings represents each word of the sentences together with the two position embeddings. Besides, the types of the two interacting entities from the NER module are converted to a real value vector with an entity type embedding matrix and concatenated with the word and position embeddings. Then, a convolutional layer computes this input matrix in order to create a higher level representation of the sentence. After that, a pooling layer extracts the most relevant features of each filter taking their maximum argument, and the resulting features for different filter sizes are concatenated into a vector. A dropout is performed before the classification to prevent overfitting, which will be disabled in test time. Finally, a Softmax layer classifies the reduced vector from the pooling.

7.2.5 Learning

During the training, the parameters of the networks are updated according to the negative log-likelihood given by the predictions for the NER and RE tasks. To learn the parameters, the minimization of the objective function is performed over shuffled mini-batches by the stochastic gradient descent for NER and Adam update rule (Kingma and Ba, 2014) for RE. Mainly, training and decoding the CRF model can be solved efficiently by adopting the Viterbi algorithm.

In the testing phase, the sequence with the highest probability given by the maximum argument of the conditional likelihood will be the prediction in the NER system. After that, these labels are taken by the RE system to predict the class of the relationship according to the highest probability in the Softmax layer of this module.

7.3 Evaluation

This section discusses the results given by each module of the information extraction system independently. Finally, the end-to-end system is developed to give the results of the entire process, which each module takes the output of the previous subsystems as the input. Precision

(P), Recall (R) and F1 are the standard evaluation metrics used for the entity and relation extraction tasks.

7.3.1 NER results

The NER model is evaluated in two different tasks: i) the detection and classification of pharmacological substances in the DDI corpus, and ii) the detection and classification of keyphrases in the eHealth-KD challenge dataset. Table 7.5 summarizes the parameters of the sets and the hyper-parameters for the Bi-LSTM CRF model:

Table 7.5: Parameters for Bi-LSTM CRF model.

Parameters	**DDI**	**eHealth-KD**
Sense embedding dimension	100	128
Word embeddings dimension	100	300
Character embedding dimension	50	50
Hidden layers dimension (for each LSTM)	100	100
Learning method	SGD	SGD
Dropout rate	0.5	0.5
Learning rate	0.005	0.005
Epochs	100	100

Table 7.6 shows the performance of the model for the entity detection task. Table 7.7 and Table 7.8 shows the results of the classification for each entity type in both tasks.

Table 7.6: Results for the entity detection task.

Datasets	**P**	**R**	**F1**
DDI	87.24%	87.15%	87.19%
eHealth-KD	86.2%	88.2%	87.2%

Comparing to previous works in drug name detection on the DDI corpus, this system also achieves good results, only 0.6% worse than the top system of the DDIExtraction Task (Rocktäschel et al., 2013), which includes a rich linguistic and semantic feature set to train a CRF classifier. An essential advantage of this approach over previous works is that the proposed system does not exploit any domain-specific features, and thereby, it could be easily adapted to any entity type. Furthermore, this approach achieves an F1 of 87.2% for the entity detection task on the eHealth-KD dataset, which is the best result for entity detection in the eHealth-KD challenge.

Moreover, the NER module provides better results on the eHealth-KD dataset than on the DDI corpus. A possible reason may be that the DDI corpus contains several discontinuous named entities, while entities in the eHealth-KD dataset do not contain any gap in their mentions, which are hard to disambiguate with the V tag.

Regarding the results for the entity classification task on the DDI corpus, the *Group* type obtains the best performance, followed by the *Drug* and *Brand* types. F1 for the *Group* type (86.06%) is almost nine points higher than the top system in DDIExtraction Task. Group of drugs are usually named by multi-word expressions, such as *'Nondepolarizing Neuromuscular Blocker'* or *'HMG CoA Reductase Inhibitor'*. Therefore, a deep network based on character, word and sense embeddings seems to obtain better results in the classification of multi-word expressions than the rich linguistic and semantic feature set used by the CRF model in (Rocktäschel et al., 2013).

Table 7.7: Entity classification results on the DDI corpus.

Classes	P	R	F1
Drug	80.73%	89.22%	84.76%
Brand	75.00%	90.00%	81.82%
Group	84.06%	94.31%	88.89%
Non-human drug	58.59%	35.21%	43.99%
Overall	76.11%	78.10%	76.21%

The proposed model obtained the best performance for the entity classification task in the eHealth-KD challenge, with an F-measure of 85%. Table 7.8 shows the results for each entity type. The *Concept* type is easier to classify than *Action* type because the instances of the *Action* type almost half of the *Concept* type instances. Furthermore, there is ambiguity in *Action* type entities that can sometimes represent *Concept*. For instance, the entity *'cuidado'* (*'dangerous'*) in the sentence *'Los empleados dedicados al cuidado de la salud están expuestos a muchos riesgos laborales.'* (*'Employees dedicated to health care are exposed to many occupational hazards.'*) is an *Action* type, but in the sentence *'Practicar deportes puede ser divertido, pero si no se tiene cuidado también puede ser peligroso.'* (*'Playing sports can be fun but if you are not careful, it can also be dangerous.'*) is a *Concept* type.

Table 7.8: Results for the entity classification on the eHealth-KD dataset.

Classes	**P**	**R**	**F1**
Concept	85.24%	86.77%	86.00%
Action	80.00%	83.22%	81.58%
Overall	84%	86%	85%

7.3.2 RE results

Table 7.9 summarizes the hyper-parameters for the CNN model where M_e, M_d, M_t, w and m were fine-tuning with a grid search on each dataset.

Table 7.9: Parameters for CNN model.

Parameters	**DDI**	**eHealth-KD**
Maximal length in the dataset, n	128	40
Word embeddings dimension, M_e	300	300
Position embeddings dimension, M_d	5	10
Type embeddings dimension, M_t	10	10
Filter window sizes, w	2, 4, 6	3, 4, 5
Filters for each window size, m	200	200
Dropout rate, p	0.5	0.5
nonlinear function, g	ReLU	ReLU
l2-regularization	3	0.1
Mini-batch size	50	50
Learning rate	0.001	0.001

In the eHealth-KD dataset, some sentences describe relationships between nested entities, that is, between an entity and its overlapped entity. It is not possible to blind all possible entity mentions forming a nested entity. For this reason, these relation instances are not considered for the training of the model. Moreover, there are relationships with more than one type in which only the first type is taken because the system cannot cope with a multi-labelling problem.

Table 7.10 and Table 7.11 show the results on Scenario 3 or RE subtask, when the annotated entities and their corresponding types are provided as input for the proposed system. These tables suggest that the relation extraction task is more complex than the name entity recognition task.

Focusing on the DDI corpus, the RE module obtains the best performance for the *advice* type because most of these interactions are typically described by similar patterns such as *'DRUG* *should not be used in combination with DRUG'* or *'Caution should be observed when DRUG is administered with DRUG'*. The model easily learns these patterns because they are very common in the DDI corpus. The *mechanism* type is the second one with the best performance (68.68%), even though its number of instances is lower than the *effect* type (see Table 7.10). Finally, the *int* type is the most difficult type to classify because the training instances for this type are much more scarce (5.6%) than those of the remainder of the types (41.1% for effect, 32.3% for *mechanism* and 20.9% for *advice*).

Table 7.10: Relation classification results on the DDI corpus.

Classes	P	R	F1
mechanism	74.23%	63.91%	68.68%
effect	65.57%	66.67%	66.12%
advice	75.12%	68.33%	71.56%
int	86.11%	32.29%	46.97%
Overall	71.26%	62.82%	66.78%

Regarding the results on the eHealth-KD dataset, the system overcomes the top system in the RE subtask improving the results from 44.8% to 54.23%. Besides, they seem to be directly correlated to the number of training instances for each relation types. In this way, the system obtains the best F1 for the target relation, followed by the subject type which is the most representative class in this dataset.

Table 7.11: Results for the relation classification on the eHealth-KD dataset.

Classes	P	R	F1
is-a	44%	15.94%	23.4%
part-of	37.5%	9.38%	15%
property-of	57.45%	43.55%	49.54%
same-as	50%	20%	28.57%
subject	57.69%	43.8%	49.79%
target	67.58%	69.81%	68.68%
Overall	61.73%	48.36%	54.23%

7.3.3 End-To-End results

The same evaluation metrics defined in eHealth-KD Task is performed to obtain a comparative of each scenario independently for the end-to-end performance. The results for the Scenario 1 are calculated with the aggregated metrics of the subtask A, B and C as follows:

$$P = \frac{correct(A) + \frac{1}{2}partial(A) + correct(B) + correct(C)}{correct(A) + partial(A) + spurious(A) + correct(B) + incorrect(B) + correct(C) + spurious(C)} \quad (7.1)$$

$$R = \frac{correct(A) + \frac{1}{2}partial(A) + correct(B) + correct(C)}{correct(A) + partial(A) + missing(A) + correct(B) + incorrect(B) + correct(C) + missing(C)} \quad (7.2)$$

where *correct* are the labels that matched to the test set and the prediction (true positives), *missing* are the labels that are in the test set but not in the prediction (false negatives), *spurious* are the labels that are in the prediction but not in the test set (false positives), *partial* are the detected entities whose boundaries do not exactly match, and *incorrect* are the entities wrongly classified. In order to calculate the same metrics for Scenarios 2 and 3, the instances for the previous tasks are cancelled in the equations. The F-measure is calculated from the Precision and Recall of each scenario of the task. Furthermore, the final score in this competition is the average of all three scenarios. Differently to eHealth-KD which has a test set for each scenario, in DDI corpus the same test is tested for all the scenarios.

Table 7.12 shows the results for each Scenario using the eHealth-KD dataset. The final averaged score for all the Scenarios is 67.62%, which is 21.2% higher than the top system in this task. The main reason for this improvement is that the winner system did not use a RE system and has lower statistics for Task C in the Scenarios. The end-to-end model obtains a state-of-the-art technique which shows a high Precision in the different Scenarios. Furthermore, it outperforms all the previous F1 measure in the eHealth-KD challenge.

Table 7.13 presents all Scenarios taking the DDI corpus test set given an average of 61.92% F1. Despite Scenario 3 in DDI corpus obtains better results than on the eHealth-KD dataset, the final score is lower because it is directly affected by the performance of the NER module which affects the remaining scenarios. Concretely, the number of spurious and incorrect are very high for task A and B which causing a low Precision in Scenario 1 and 2.

Table 7.12: eHealth-KD dataset results for the different scenarios.

Evaluation metrics	Scenario 1	Scenario 2	Scenario 3
Correct A	505	-	-
Partial A	40	-	-
Missing A	50	-	-
Spurious A	64	-	-
Correct B	511	553	-
Incorrect B	34	40	-
Correct C	168	205	250
Missing C	324	291	267
Spurious C	240	183	155
Recall	73.78%	69.61%	48.36%
Precision	77.08%	77.27%	61.73%
F-measure	75.39%	73.24%	54.23%

Table 7.13: DDI corpus results for the different scenarios.

Evaluation metrics	Scenario 1	Scenario 2	Scenario 3
Correct A	2236	-	-
Partial A	67	-	-
Missing A	149	-	-
Spurious A	1488	-	-
Correct B	2149	2149	-
Incorrect B	1503	1503	-
Correct C	559	559	615
Missing C	253	253	364
Spurious C	951	951	248
Recall	71.97%	60.66%	62.82%
Precision	55.6%	52.46%	71.26%
F-measure	62.73%	56.26%	66.78%

7.4 Conclusions

The end-to-end IE system for biomedical and clinical texts deals with three different tasks: A) entity detection, B) entity classification and C) relation extraction. The architecture is composed of two different modules based on Deep Learning architectures: one for detecting and classifying entities, and a second one for extracting relationships between them. The NER module is based on a Bi-LSTM network, exploiting character, word and sense embeddings models as input, and a final CRF-layer. The RE module is a CNN model using the word, the entity type and the position embeddings as inputs, and a Softmax classifier in its last layer. In this section, a detailed experimentation on two different datasets is presented for: the DDI corpus composed of scientific texts in English and annotated with pharmacological substances as well as their possible drug interactions; and the eHealth-KD dataset composed by articles about health for Spanish speaking patients and annotated with concepts, actions and general semantic relations such as part-of, property-of, among others (see Table 7.2). First, each module of the system is evaluated separately. Thus, the NER module does not require any previous annotation, but the RE module takes as input the texts annotated with the entities from the previous module. Finally, to assess the end-to-end system in a real scenario, the pain texts are taking as input.

The presented system achieves the state-of-the-art results for all subtasks (NER and RE), when it is evaluated in the eHealth-KD dataset. Additionally, the end-to-end pipeline is the state-of-the-art system for the eHealth-KD challenge. Focusing on the DDI corpus, the NER module outperforms the system for drug name recognition task with the DDI corpus (Rocktäschel et al., 2013), but this module is far from the state-of-the-art system for the DDIExtraction Task because is a basic CNN compared to the system with ten CNN layers from (Dewi et al., 2017).

This chapter does not only achieve the state-of-the-art results for the eHealth-KD challenge, to the best of our knowledge, the system is also the first attempt to develop an end-to-end system for extracting DDI from texts. Another significant contribution is that the approach is a task and language independent method because it deals with different languages (such as Spanish and English) as well as different entity and relation types.

There is much room for improvement of the end-to-end system. As future work, it is planned to study different combinations of Deep Learning architectures for NER and RE modules. Besides, exploring deeper layers systems and trying different word embeddings pre-trained in the clinical domain could improve the results for both tasks in DDI corpus. Furthermore, the architecture could include more syntactic information of the sentence, such as POS tags, Chunk labels, dependency types, through the embeddings. Mainly, the augmentation of the class instances with distant supervision techniques or Generative Adversarial Neural Networks could deal with the problem of imbalanced datasets for the IE tasks.

Chapter 8

Conclusion

This chapter presents the conclusions of the work realized. This document shows different models to tackle Information Extraction in the Biomedical Domain. These approaches are based on Deep Learning architectures created for Named Entity Recognition and Relation Extraction.

Firstly, a CRF-based NER system that incorporates word embeddings as features and DINTO ontology information performs the drug mention discovery, DrugNER. The experiments show that both features improve the detection performance for the extraction of drug names. Concretely, the clusters of the word embeddings increase the recall, and the use of DINTO increase the precision of the model. Additionally, this system is evaluated for other biomedical challenge tasks such as CHEMDNER-patents (CEMP) and Chemical Disease Relation (CDR) obtaining good results in its performance. As the main conclusion, word cluster features trained on Wikipedia corpus seem to provide the most satisfactory results. Moreover, the inclusion of additional embeddings from biomedical resources should improve the classification performance.

Secondly, the Matrix-Vector Recursive Neural Network, which was the first Deep Learning architecture applied to Relation Extraction, is implemented for the classification of drug interaction in the DDI Corpus. The Stanford parse tree of each sentence defines the structure of the network. This parser is not able to capture the structural complexity of the biomedical sentences, and the architecture produces low performance in classification. Thus, a biomedical parser should be explored for the extraction of the tokens and the syntactic trees for the DDI sentences. Despite this lack of representation, the system outperforms other systems of the participant in the DDIExtraction Task that employ manually designed features.

Thirdly, the Convolutional Neural Network, which is a state-of-the-art Deep Learning architecture, is tested for the DDIExtraction Shared Task. To this end, an exhaustive and detailed study of the CNN hyper-parameters evaluates the performance of the architecture and obtains the best configuration for the classification of the DDI sentences. This CNN model only uses the word embeddings and the relative position of each word for the interacting entities transformed into a real value vector as inputs. The experiments present promising results without adding any external information or resource to perform the classification. Moreover, the CNN performance with a negative instance filtering is compared for different pooling operations in the extraction of DDIs: max-pooling, attentive pooling and average-pooling. The experiments exhibit that the maximum operation performs higher than the remaining pooling layers, mainly, because of the padding tokens appended to the sentences. Furthermore, the basic configuration of a CNN has been validated satisfactorily in other Relation Extraction Shared Tasks such as the SemEval 2017 Task 10: ScienceIE and SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers. The use of position embeddings increases the results for the former task. Aggregating to the CNN a sampling techniques to unbalance the dataset, the entity type embeddings and POS embeddings improve the performance for the latter task.

Finally, an end-to-end IE system is built for a real scenario where the inputs of the system are the raw texts. Thus, the model detects and classifies the entities and their relationships in biomedical and clinical documents. The system consists of two modules based on deep learning. Firstly, a bidirectional RNN with LSTM cells, which takes the character, word and sense embeddings, performs the recognition of named mentions using a CRF classification layer. Secondly, a CNN with a Softmax layer, which takes the word, relative position and entity type embeddings, classifies the relationships of the entities given in the previous step. This system obtains the state-of-the-art in some tasks for different language datasets, English and Spanish. Besides, the architecture is the first attempt to develop an end-to-end system for the DDI Corpus that involves the DrugNER and the DDIExtraction.

The main objective of this thesis is to evaluate whether Deep Learning architectures are a real alternative to classical machine learning models for Information Extraction in the particular domain of the biomedical texts. The starting hypothesis is that previous supervised models for IE in the biomedical domain require much expert knowledge in order to manually design a good feature representation, contrary to this fact, Deep Learning architectures learn the feature representation automatically. Thus, the time-consuming task and the definition of complex representations used for feature engineering models can be avoided using feature learning techniques without any external resource.

The main contribution of this thesis is to give a complete description of the current systems for Information Extraction in biomedical documents so far. Concretely, the review presents Deep Learning models applied to texts that contain drug interactions. Thus, Chapter 3 concludes with tables that summarize the state-of-the-art systems for the DDIExtraction Task. Besides, Chapter 4 investigates the use of the word embeddings and their clusters as features for the recognition of drugs, chemical compounds, and diseases with a CRF classifier in biomedical texts. Another contribution is the exploration of the Matrix-Vector spaces for the extraction of DDIs using the MV-RNN, which is the first Deep Learning system for RE. Notably, Chapter 5 describes the performance of this architecture for each DDI type and the two datasets, DDI-DrugBank and DDI-MedLine, in order to check the specific misclassification error rates. Moreover, this document presents a detailed study of the hyperparameters used in the CNN fine-tuning them until reaching the best configuration for the DDI Corpus. Chapter 6 also proposes and test the first CNN with attention pooling for this task. Additionally, the CNN architecture is applied to deal similar tasks such as SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications and at SemEval 2018 Task 7: Semantic Relation Extraction and Classification in Scientific Papers in order to check their efficiency in other domains. Furthermore, Chapter 7 defines the first end-to-end model for classification of entities and their relationships from raw data in sentences that involve drug interactions. The eHealth Knowledge Discovery dataset, which contains electronic health documents written in Spanish, is also used for this architecture in order to test its language and domain independence with satisfactory results.

After testing the proposed architectures and reviewing the state-of-the-art techniques, the conclusion is that Deep Learning outperforms machine learning based on feature engineering. Furthermore, the inputs of Deep Learning architectures are the mapping of each word to a real value vector called embeddings. For this reason, the proposed systems are domain and language independent without the use of external features. Additionally, these systems could add the available information extracted with external resources as input embeddings for improving the feature representation.

Deep Learning is a growing field that provides top performance in the majority of machine learning applications. The research on new architectures, even the combination of existing models, progresses very fast in a constant manner. For this reason, there is much room for improvement in Biomedical Information Extraction. Thus, the exploration of state-of-the-art techniques in IE is a vital process to validate the systems in this specific domain. Therefore, combinations of Deep Learning models should be implemented for further improve, even adding deeper layers to generate different abstraction levels.

Given that the inputs of the architectures are word embeddings pre-trained on an extensive collection of biomedical texts should provide better performance than randomly initialized word embeddings improving the semantic representation for each word. Currently, there is a trend to discover a universal word embeddings that can be applied in multiple tasks. For this reason, using them in the extraction of biomedical information could validate their performance on this specific task additionally to the general purpose task because this domain includes biomedical technical terms and jargon. The related work summarizes that multi-channel word embedding provides better results because the several channels or embedding models increases the semantic space knowledge of each word. Furthermore, the aggregation of character embeddings provide morphological and orthographic information of the sub-word units and increase the result for the presented models. Additionally, the architectures could include more information about the sentence with the aggregation of lemmas, stems, POS tags, Chunk labels, dependency and constituent types, among others as the input embeddings.

It is planned to create two-step models for the IE tasks (detection and classification) which perform better according to the related works than the presented one-stage models. In order to obtain better structures for the MV-RNN, a biomedical parser should be used instead of the Stanford parser, which is the leading cause of misclassification. IE systems based on Deep Learning methods should incorporate attention mechanisms, but they should avoid the padding of sentences for the pooling layer.

Most of IE tasks are very unbalanced which generates different performance for each class using supervised classification models. For this reason, the incorporation of sampling techniques alleviates this problem, such as the random undersampling or the random oversampling. Particularly, preparing new examples for the less representative categories to lead to an increase in the variability and the performance of the models for these classes. Thus, the augmentation of the class instances with distant supervision techniques or Generative Adversarial Neural Networks could deal with the problem of imbalanced datasets.

Finally, it is a hard task for Deep Learning systems creating an exhaustive evaluation for the classification of the sentences due to the high abstractive information generated by the networks. However, performing a comprehensive error analysis of the misclassified instances by an expert would help to understand why the architecture is failing.

Bibliography

Aizerman, M. A., Braverman, E. M. and Rozonoér, L. I. (1964), 'Theoretical foundations of the potential function method in pattern recognition learning', *Automation and Remote Control* **25**, 821–837.

Aronson, A. R. (2001), Effective mapping of biomedical text to the umls metathesaurus: the metamap program., *in* 'Proceedings of the AMIA Symposium', American Medical Informatics Association, p. 17.

Aronson, J. (2004), 'Drug interactions-information, education, and the British National Formulary.', *British Journal of Clinical Pharmacology* **57**(4), 473–86.

Aronson, J. (2007), 'Communicating information about drug interactions', *British Journal of Clinical Pharmacology* **63**, 637–639.

Asada, M., Miwa, M. and Sasaki, Y. (2017), Extracting drug-drug interactions with attention cnns, *in* 'BioNLP 2017', Association for Computational Linguistics, pp. 9–18.
URL: *http://aclweb.org/anthology/W17-2302*

Asada, M., Miwa, M. and Sasaki, Y. (2018), 'Enhancing drug-drug interaction extraction from texts by molecular structure information', pp. 680–685.
URL: *http://aclweb.org/anthology/P18-2108*

Augenstein, I., Das, M., Riedel, S., Vikraman, L. and McCallum, A. (2017), Semeval 2017 task 10: Scienceie - extracting keyphrases and relations from scientific publications, *in* 'Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)', Associa-

tion for Computational Linguistics, pp. 546–555.
URL: *http://aclweb.org/anthology/S17-2091*

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970), 'A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains', *The Annals of Mathematical Statistics* **41**(1), 164–171.
URL: *http://www.jstor.org/stable/2239727*

Bellman, R., Bellman, R. and Corporation, R. (1957), *Dynamic Programming*, Rand Corporation research study, Princeton University Press.
URL: *https://books.google.es/books?id=rZW4ugAACAAJ*

Bengio, Y. (2009), *Learning Deep Architectures for AI*, Vol. 2, Now Publishers Inc., Hanover, MA, USA.
URL: *http://dx.doi.org/10.1561/2200000006*

Bengio, Y., Ducharme, R., Vincent, P. and Janvin, C. (2003), 'A neural probabilistic language model', *J. Mach. Learn. Res.* **3**, 1137–1155.
URL: *http://dl.acm.org/citation.cfm?id=944919.944966*

Bird, S. (2006), Nltk: The natural language toolkit, *in* 'Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions', Association for Computational Linguistics, pp. 69–72.
URL: *http://aclweb.org/anthology/P06-4018*

Björne, J. and Salakoski, T. (2018), Biomedical event extraction using convolutional neural networks and dependency parsing, *in* 'Proceedings of the BioNLP 2018 workshop', Association for Computational Linguistics, pp. 98–108.
URL: *http://aclweb.org/anthology/W18-2311*

Björne, J. et al. (2014), 'Biomedical event extraction with machine learning'.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2017), 'Enriching word vectors with subword information', *Transactions of the Association for Computational Linguistics* **5**, 135–146.
URL: *http://aclweb.org/anthology/Q17-1010*

Bond, C. and Raehl, C. L. (2006), 'Adverse drug reactions in united states hospitals', *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy* **26**(5), 601–608.

Bryson, A. E., Denham, W. F. and Dreyfus, S. E. (1963), 'Optimal programming problems with inequality constraints', *AIAA journal* **1**(11), 2544–2550.

Bui, Q.-C., Sloot, P. M., Van Mulligen, E. M. and Kors, J. A. (2014), 'A novel feature-based approach to extract drug–drug interactions from biomedical text', *Bioinformatics* **30**(23), 3365–3371.

Cardellino, C. (2016), 'Spanish Billion Words Corpus and Embeddings'.
URL: *https://crscardellino.github.io/SBWCE/*

Chalapathy, R., Zare Borzeshi, E. and Piccardi, M. (2016), 'An investigation of recurrent neural architectures for drug name recognition', pp. 1–5.
URL: *http://aclweb.org/anthology/W16-6101*

Charniak, E. (2000), A maximum-entropy-inspired parser, *in* '1st Meeting of the North American Chapter of the Association for Computational Linguistics'.
URL: *http://aclweb.org/anthology/A00-2018*

Charniak, E. and Johnson, M. (2005), Coarse-to-fine n-best parsing and maxent discriminative reranking, *in* 'Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)', Association for Computational Linguistics, pp. 173–180.
URL: *http://aclweb.org/anthology/P05-1022*

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014), 'Learning phrase representations using rnn encoder–decoder for statistical machine translation', pp. 1724–1734.
URL: *http://aclweb.org/anthology/D14-1179*

Chowdhury, M. F. M. and Lavelli, A. (2013*a*), Exploiting the scope of negations and heterogeneous features for relation extraction: A case study for drug-drug interaction extraction, *in* 'Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies', pp. 765–771.

Chowdhury, M. F. M. and Lavelli, A. (2013*b*), 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)', Association for Computational Linguistics, pp. 351–355.
URL: *http://aclweb.org/anthology/S13-2057*

Chung, J., Gülçehre, Ç., Cho, K. and Bengio, Y. (2014), 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *arXiv e-prints* **abs/1412.3555**. Presented at the Deep Learning workshop at NIPS2014.
URL: *https://arxiv.org/abs/1412.3555*

Ciaramita, M. and Altun, Y. (2006), Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger, *in* 'Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics, pp. 594–602.
URL: *http://aclweb.org/anthology/W06-1670*

Collobert, R. and Weston, J. (2008), A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning, *in* 'Proceedings of the 25th International Conference on Machine Learning', ICML '08, ACM, New York, NY, USA, pp. 160–167.

Cortes, C. and Vapnik, V. (1995), 'Support-vector networks', *Machine learning* **20**(3), 273–297.

Coulet, A., Garten, Y., Dumontier, M., Altman, R. B., Musen, M. A. and Shah, N. H. (2011), 'Integration and publication of heterogeneous text-mined relationships on the Semantic Web', *J Biomed Semantics* **2 Suppl 2**, S10.

Council, N. R. and of Medicine, I. (2009), *Preventing Mental, Emotional, and Behavioral Disorders Among Young People: Progress and Possibilities*, The National Academies Press, Washington, DC.
URL: *https://www.nap.edu/catalog/12480/preventing-mental-emotional-and-behavioral-disorders-among-young-people-progress*

Cybenko, G. (1989), 'Approximation by superpositions of a sigmoidal function', *Mathematics of Control, Signals, and Systems (MCSS)* **2**(4), 303–314.
URL: *http://dx.doi.org/10.1007/BF02551274*

Davis, A. P., Wiegers, T. C., Roberts, P. M., King, B. L., Lay, J. M., Lennon-Hopkins, K., Sciaky, D., Johnson, R., Keating, H., Greene, N., Hernandez, R., McConnell, K. J., Enayetallah, A. E. and Mattingly, C. J. (2013), 'A CTD-Pfizer collaboration: manual curation of 88,000 scientific articles text mined for drug-disease and drug-phenotype interactions', *Database (Oxford)* .

de Matos, P., Alcantara, R., Dekker, A., Ennis, M., Hastings, J., Haug, K., Spiteri, I., Turner, S. and Steinbeck, C. (2010), 'Chemical Entities of Biological Interest: an update', *Nucleic Acids Res.* **38**(Database issue), D249–254.

De Vine, L., Zuccon, G., Koopman, B., Sitbon, L. and Bruza, P. (2014), Medical semantic similarity with a neural language model, *in* 'Proceedings of the 23rd ACM international conference on conference on information and knowledge management', ACM, pp. 1819–1822.

Degtyarenko, K., De Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., Alcántara, R., Darsow, M., Guedj, M. and Ashburner, M. (2008), 'Chebi: a database and ontology for chemical entities of biological interest', *Nucleic acids research* **36**(suppl 1), D344–D350.

Dewi, I. N., Dong, S. and Hu, J. (2017), Drug-drug interaction relation extraction with deep convolutional neural networks, *in* '2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)', pp. 1795–1802.

dos Santos, C. and Gatti, M. (2014), Deep convolutional neural networks for sentiment analysis of short texts, *in* 'Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers', Dublin City University and Association for Computational Linguistics, pp. 69–78.
URL: *http://aclweb.org/anthology/C14-1008*

dos Santos, C. N., Xiang, B. and Zhou, B. (2015), 'Classifying relations by ranking with convolutional neural networks', *ACL 2015* p. 9.

Duchi, J., Hazan, E. and Singer, Y. (2011), 'Adaptive subgradient methods for online learning and stochastic optimization', *Journal of Machine Learning Research* **12**(Jul), 2121–2159.

Duda, S., Aliferis, C., Miller, R., Statnikov, A. and Johnson, K. (2005), Extracting drug-drug interaction articles from MEDLINE to improve the content of drug databases, *in* 'AMIA Annual Symposium Proceedings', Vol. 2005, p. 216.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A. and Adams, R. P. (2015), Convolutional networks on graphs for learning molecular fingerprints, *in* 'Advances in neural information processing systems', pp. 2224–2232.

Ebrahimi, J. and Dou, D. (2015), Chain based rnn for relation classification, *in* 'Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies', Association for Computational Linguistics, pp. 1244–1249.
URL: *http://aclweb.org/anthology/N15-1133*

Elman, J. L. (1990), 'Finding structure in time', *Cognitive Science* **14**(2), 179 – 211.
URL: *http://www.sciencedirect.com/science/article/pii/036402139090002E*

Explosion AI (2017), 'spaCy - Industrial-strength Natural Language Processing in Python'.
URL: *https://spacy.io/*

Fellbaum, C., ed. (1998), *WordNet: an electronic lexical database*, MIT Press.

Ferner, R. and Aronson, J. (2006), 'Communicating drug safety.', *JBM* **333**, 1435.

Gábor, K., Buscaldi, D., Schumann, A.-K., QasemiZadeh, B., Zargayouna, H. and Charnois, T. (2018), Semeval-2018 task 7: Semantic relation extraction and classification in scientific papers, *in* 'Proceedings of The 12th International Workshop on Semantic Evaluation', Association for Computational Linguistics, pp. 679–688.
URL: *http://aclweb.org/anthology/S18-1111*

Giuliano, C., Lavelli, A. and Romano, L. (2006), Exploiting shallow linguistic information for relation extraction from biomedical literature, *in* '11th Conference of the European Chapter

of the Association for Computational Linguistics'.
URL: *http://aclweb.org/anthology/E06-1051*

Glorot, X. and Bengio, Y. (2010), Understanding the difficulty of training deep feedforward neural networks, *in* Y. W. Teh and M. Titterington, eds, 'Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics', Vol. 9 of *Proceedings of Machine Learning Research*, PMLR, Chia Laguna Resort, Sardinia, Italy, pp. 249–256.
URL: *http://proceedings.mlr.press/v9/glorot10a.html*

Goodfellow, I., Bengio, Y. and Courville, A. (2016), *Deep Learning*, MIT Press. `http://www.deeplearningbook.org`.

Hansten, P. (2003), 'Drug interaction management', *Pharmacy World & Science* **25**(3), 94–97.

Harris, Z. S. (1954*a*), 'Distributional structure', *¡i¿WORD¡/i¿* **10**(2-3), 146–162.
URL: *https://doi.org/10.1080/00437956.1954.11659520*

Harris, Z. S. (1954*b*), 'Distributional structure', *Word* **10**(2-3), 146–162.

He, K., Zhang, X., Ren, S. and Sun, J. (2015), 'Delving deep into rectifiers: Surpassing human-level performance on imagenet classification', pp. 1026–1034.
URL: *http://dx.doi.org/10.1109/ICCV.2015.123*

He, Y., Sarntivijai, S., Lin, Y., Xiang, Z., Guo, A., Zhang, S., Jagannathan, D., Toldo, L., Tao, C. and Smith, B. (2014), 'Oae: the ontology of adverse events', *Journal of biomedical semantics* **5**(1), 29.

Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Ó Séaghdha, D., Padó, S., Pennacchiotti, M., Romano, L. and Szpakowicz, S. (2009), Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals, *in* 'Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions', DEW '09, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 94–99.

Herrero Zazo, M. (2015), Semantic Resources in Pharmacovigilance: A Corpus and an Ontology for Drug-Drug Interactions, PhD thesis, Carlos III University of Madrid.

Herrero-Zazo, M., Segura-Bedmar, I., Hastings, J. and Martínez, P. (2015), 'Dinto: using owl ontologies and swrl rules to infer drug–drug interactions and their mechanisms', *J. Chem. Inf. Model.* **55**(8), 1698–1707.

Herrero-Zazo, M., Segura-Bedmar, I., Martínez, P. and Declerck, T. (2013), 'The DDIcorpus: An annotated corpus with pharmacological substances and drug-drug interactions', *Journal of Biomedical Informatics* **46**(5), 914 – 920.

Hersh, W., Buckley, C., Leone, T. and Hickam, D. (1994), Ohsumed: an interactive retrieval evaluation and new large test collection for research, *in* 'SIGIR'94', Springer, pp. 192–201.

Hettne, K. M., Stierum, R. H., Schuemie, M. J., Hendriksen, P. J., Schijvenaars, B. J., Mulligen, E. M. v., Kleinjans, J. and Kors, J. A. (2009), 'A dictionary to identify small molecules and drugs in free text', *Bioinformatics* **25**(22), 2983–2991.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2012), 'Improving neural networks by preventing co-adaptation of feature detectors', *CoRR* **abs/1207.0580**.
URL: *http://arxiv.org/abs/1207.0580*

Hirschman, L., Yeh, A., Blaschke, C. and Valencia, A. (2005), 'Overview of biocreative: critical assessment of information extraction for biology', *BMC Bioinformatics* **6**(1), S1.

Hochreiter, S., Bengio, Y., Frasconi, P. and Schmidhuber, J. (2001), 'Gradient flow in recurrent nets: the difficulty of learning long-term dependencies'.

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

Hornik, K. (1991), 'Approximation capabilities of multilayer feedforward networks', *Neural Netw.* **4**(2), 251–257.
URL: *http://dx.doi.org/10.1016/0893-6080(91)90009-T*

Huang, D., Jiang, Z., Zou, L. and Li, L. (2017), 'Drug–drug interaction extraction from biomedical literature using support vector machine and long short term memory networks', *Infor-*

mation Sciences **415-416**, 100 – 109.
URL: *http://www.sciencedirect.com/science/article/pii/S002002551730110X*

Jiang, J. and Zhai, C. (2007), 'An empirical study of tokenization strategies for biomedical information retrieval', *Information Retrieval* **10**(4-5), 341–363.

Jiang, Z., Gu, L. and Jiang, Q. (2017), Drug drug interaction extraction from literature using a skeleton long short term memory neural network, *in* '2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)', pp. 552–555.

Jiang, Z., Li, L., Huang, D. and Jin, L. (2015), Training word embeddings for deep learning in biomedical text mining tasks, *in* 'Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on', IEEE, pp. 625–628.

Joachims, T. (1999), Making large-scale SVM learning practical, *in* B. Schölkopf, C. Burges and A. Smola, eds, 'Advances in Kernel Methods - Support Vector Learning', MIT Press, Cambridge, MA, chapter 11, pp. 169–184.

Johnson, A. E., Pollard, T. J., Shen, L., Li-wei, H. L., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Celi, L. A. and Mark, R. G. (2016), 'Mimic-iii, a freely accessible critical care database', *Scientific data* **3**, 160035.

Jordan, M. I. (1997), 'Chapter 25 - serial order: A parallel distributed processing approach', **121**, 471 – 495.
URL: *http://www.sciencedirect.com/science/article/pii/S0166411597801112*

Kavuluru, R., Rios, A. and Tran, T. (2017), Extracting drug-drug interactions with word and character-level recurrent neural networks, *in* '2017 IEEE International Conference on Healthcare Informatics (ICHI)', IEEE, Park City, UT, USA, USA, pp. 5 – 12.

Kim, J.-D., Ohta, T., Tateisi, Y. and Tsujii, J. (2003), 'Genia corpus—a semantically annotated corpus for bio-textmining', *Bioinformatics (Oxford, England)* **19 Suppl 1**, i180–2.

Kim, S., Liu, H., Yeganova, L. and 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.
URL: *http://www.sciencedirect.com/science/article/pii/S1532046415000441*

Kim, Y. (2014), Convolutional neural networks for sentence classification, *in* 'Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, pp. 1746–1751.
URL: *http://aclweb.org/anthology/D14-1181*

Kingma, D. P. and Ba, J. (2014), 'Adam: A method for stochastic optimization', *CoRR* **abs/1412.6980**.

Klein, D. and Manning, C. D. (2003), Accurate unlexicalized parsing, *in* 'Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1', ACL '03, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 423–430.
URL: *https://doi.org/10.3115/1075096.1075150*

Klinger, R., Kolářik, C., Fluck, J., Hofmann-Apitius, M. and Friedrich, C. M. (2008), 'Detection of iupac and iupac-like chemical names', *Bioinformatics* **24**(13), i268–i276.

Krallinger, M., Leitner, F., Rabal, O., Vazquez, M., Oyarzabal, J. and Valencia, A. (2015), 'Chemdner: The drugs and chemical names extraction challenge.', *J. Cheminformatics* **7**(S-1), S1.

Krallinger, M., Rabal, O., Leitner, F., Vazquez, M., Salgado, D., Lu, Z., Leaman, R., Lu, Y., Ji, D., Lowe, D. M. et al. (2015), 'The chemdner corpus of chemicals and drugs and its annotation principles', *Journal of cheminformatics* **7**(Suppl 1), S2.

Krithara, A., Nentidis, A., Paliouras, G. and Kakadiaris, I. (2016), Results of the 4th edition of bioasq challenge, *in* 'Proceedings of the Fourth BioASQ workshop', Association for Computational Linguistics, Berlin, Germany, pp. 1–7.

Lafferty, J. D., McCallum, A. and Pereira, F. C. N. (2001), 'Conditional random fields: Probabilistic models for segmenting and labeling sequence data', pp. 282–289.
URL: *http://dl.acm.org/citation.cfm?id=645530.655813*

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C. (2016), 'Neural architectures for named entity recognition', pp. 260–270.
URL: *http://aclweb.org/anthology/N16-1030*

Lamurias, A., Clarke, L. A. and Couto, F. M. (2018), 'Bo-lstm: Classifying relations via long short-term memory networks along biomedical ontologies', *bioRxiv* .
URL: *https://www.biorxiv.org/content/early/2018/06/01/336719*

Landrum, G. (2013), 'Rdkit: Open-source cheminformatics'.
URL: *https://www.rdkit.org/docs/*

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

Leaman, R. and Gonzalez, G. (2008), Banner: An executable survey of advances in biomedical named entity recognition, *in* 'Pacific Symposium on Biocomputing 2008, PSB 2008', pp. 652–663.

Leaman, R., Islamaj Doğan, R. and Lu, Z. (2013), 'Dnorm: disease name normalization with pairwise learning to rank', *Bioinformatics* **29**(22), 2909–2917.

LeCun, Y., Bengio, Y. and Hinton, G. (2015), 'Deep learning', *Nature* **521**(7553), 436–444.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998), 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE* **86**(11), 2278–2324.

Lee, J. Y., Dernoncourt, F. and Szolovits, P. (2017), 'Mit at semeval-2017 task 10: Relation extraction with convolutional neural networks', pp. 978–984.
URL: *http://aclweb.org/anthology/S17-2171*

Lee, S., Kim, D., Lee, K., Choi, J., Kim, S., Jeon, M., Lim, S., Choi, D., Kim, S., Tan, A.-C. and Kang, J. (2016), 'Best: Next-generation biomedical entity search tool for knowledge discovery from biomedical literature', *PLOS ONE* **11**(10), 1–16.
URL: *https://doi.org/10.1371/journal.pone.0164680*

Li, W. and McCallum, A. (2003), 'Rapid development of hindi named entity recognition using conditional random fields and feature induction', **2**(3), 290–294.
URL: *http://doi.acm.org/10.1145/979872.979879*

Li, Y., Tarlow, D., Brockschmidt, M. and Zemel, R. S. (2015), 'Gated graph sequence neural networks', *CoRR* **abs/1511.05493**.
URL: *http://arxiv.org/abs/1511.05493*

Lim, S., Lee, K. and Kang, J. (2018), 'Drug drug interaction extraction from the literature using a recursive neural network', *PLOS ONE* **13**(1), 1–17.
URL: *https://doi.org/10.1371/journal.pone.0190926*

Lin, D. and Wu, X. (2009), Phrase clustering for discriminative learning, *in* 'Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP', Association for Computational Linguistics, pp. 1030–1038.
URL: *http://aclweb.org/anthology/P09-1116*

Lin, T., Goyal, P., Girshick, R., He, K. and Dollár, P. (2017), Focal loss for dense object detection, *in* '2017 IEEE International Conference on Computer Vision (ICCV)', pp. 2999–3007.

Lin, Y., Shen, S., Liu, Z., Luan, H. and Sun, M. (2016), Neural relation extraction with selective attention over instances, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Association for Computational Linguistics, pp. 2124–2133.
URL: *http://www.aclweb.org/anthology/P16-1200*

Liu, S., Chen, K., Chen, Q. and Tang, B. (2016), Dependency-based convolutional neural network for drug-drug interaction extraction, *in* '2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)', pp. 1074–1080.

Liu, S., Tang, B., Chen, Q. and Wang, X. (2015), 'Effects of semantic features on machine learning-based drug name recognition systems: word embeddings vs. manually constructed dictionaries', *Information* **6**(4), 848–865.

Liu, S., Tang, B., Chen, Q. and Wang, X. (2016), 'Drug-drug interaction extraction via convolutional neural networks', *Computational and Mathematical Methods in Medicine* **Vol. 2016**, 8 pages.

Liu, S., Tang, B., Chen, Q., Wang, X. and Fan, X. (2015), 'Feature engineering for drug name recognition in biomedical texts: Feature conjunction and feature selection', *Computational and mathematical methods in medicine* **2015**.

Makary, M. A. and Daniel, M. (2016), 'Medical error—the third leading cause of death in the us', *Bmj* **353**, i2139.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. and McClosky, D. (2014), The stanford corenlp natural language processing toolkit, *in* 'Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations', Association for Computational Linguistics, pp. 55–60.
URL: *http://aclweb.org/anthology/P14-5010*

Martínez-Cámara, E., Almeida-Cruz, Y., Díaz-Galiano, M. C., Estévez-Velarde, S., García-Cumbreras, M. A., García-Vega, M., Gutiérrez, Y., Montejo Ráez, A., Montoyo, A., Muñoz, R., Piad-Morffis, A. and Julio, V.-R. (2018), Overview of TASS 2018: Opinions, health and emotions, *in* E. Martínez-Cámara, Y. Almeida Cruz, M. C. Díaz-Galiano, S. Estévez Velarde, M. A. García-Cumbreras, M. García-Vega, Y. Gutiérrez Vázquez, A. Montejo Ráez, A. Montoyo Guijarro, R. Muñoz Guillena, A. Piad Morffis and J. Villena-Román, eds, 'Proceedings of TASS 2018: Workshop on Semantic Analysis at SEPLN (TASS 2018)', Vol. 2172 of *CEUR Workshop Proceedings*, CEUR-WS, Sevilla, Spain.

McClosky, D., Charniak, E. and Johnson, M. (2010), Automatic domain adaptation for parsing, *in* 'Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics', Association for Computational

Linguistics, pp. 28–36.
URL: *http://aclweb.org/anthology/N10-1004*

McCulloch, W. S. and Pitts, W. (1943), 'A logical calculus of the ideas immanent in nervous activity', *The bulletin of mathematical biophysics* **5**(4), 115–133.

Medline (2007), 'National library of medicine: Fact sheet medline'.
URL: *http://www.nlm.nih.gov/pubs/factsheets/medline.html*

Meyer, C. F. (2007), *Apposition in contemporary English*, Cambridge University Press.

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013), 'Efficient estimation of word representations in vector space', *CoRR* **abs/1301.3781**.
URL: *http://arxiv.org/abs/1301.3781*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J. (2013), 'Distributed representations of words and phrases and their compositionality', pp. 3111–3119.
URL: *http://dl.acm.org/citation.cfm?id=2999792.2999959*

Mikolov, T., Yih, W.-t. and Zweig, G. (2013), Linguistic regularities in continuous space word representations, *in* 'Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies', Association for Computational Linguistics, pp. 746–751.
URL: *http://aclweb.org/anthology/N13-1090*

Miyao, Y. and Tsujii, J. (2008), 'Feature forest models for probabilistic hpsg parsing', *Computational linguistics* **34**(1), 35–80.

Moschitti, A. (2004), A study on convolution kernels for shallow semantic parsing, *in* 'Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics', ACL '04, Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: *https://doi.org/10.3115/1218955.1218998*

Nesterov, Y. (1983), A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2), *in* 'Doklady AN USSR', Vol. 269, pp. 543–547.

Nielsen, M. A. (2015), *Neural Neetworks and Deep Learning*, Determination Press. `http://neuralnetworksanddeeplearning.com`.

Okazaki, N. (2007), 'Crfsuite: a fast implementation of conditional random fields (crfs)'.
URL: *http://www.chokkan.org/software/crfsuite/*

Parker, R., Graff, D., Kong, J., Chen, K. and Maeda, K. (2011), English gigaword fifth edition ldc2011t07., *in* 'Philadelphia: Linguistic Data Consortium'.

Pascanu, R., Mikolov, T. and Bengio, Y. (2013), On the difficulty of training recurrent neural networks, *in* 'Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28', ICML'13, JMLR.org, pp. III–1310–III–1318.
URL: *http://dl.acm.org/citation.cfm?id=3042817.3043083*

Pennington, J., Socher, R. and Manning, C. (2014), Glove: Global vectors for word representation, *in* 'Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)', Association for Computational Linguistics, pp. 1532–1543.
URL: *http://aclweb.org/anthology/D14-1162*

Pyysalo, S., Airola, A., Heimonen, J., Bjorne, J., Ginter, F. and Salakoski, T. (2008), 'Comparative analysis of five protein-protein interaction corpora', *BMC bioinformatics* **9**(Suppl 3), S6.

Pyysalo, S., Ginter, F., Moen, H., Salakoski, T. and Ananiadou, S. (2013), Distributional semantics resources for biomedical text processing, *in* 'Proceedings of LBM 2013', pp. 39–44.
URL: *http://lbm2013.biopathway.org/lbm2013proceedings.pdf*

Qian, N. (1999), 'On the momentum term in gradient descent learning algorithms', *Neural Networks* **12**(1), 145 – 151.
URL: *http://www.sciencedirect.com/science/article/pii/S0893608098001166*

Quan, C., Hua, L., Sun, X. and Bai, W. (2016), 'Multichannel convolutional neural network for biological relation extraction', *Biomed Res Int* **2016**, 1850404.
URL: *http://www.ncbi.nlm.nih.gov/pmc/articles/PMC5174749/*

Raihani, A. and Laachfoubi, N. (2016), 'Extracting drug-drug interactions from biomedical text using a feature-based kernel approach.', *Journal of Theoretical & Applied Information Technology* **92**(1), 109 – 120.

Raihani, A. and Laachfoubi, N. (2017), 'A rich feature-based kernel approach for drug-drug interaction extraction', *International Journal of Advanced Computer Science and Applications* **8**(4).
URL: *http://dx.doi.org/10.14569/IJACSA.2017.080445*

Raj, D., SAHU, S. and Anand, A. (2017), Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text, *in* 'Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)', Association for Computational Linguistics, pp. 311–321.
URL: *http://aclweb.org/anthology/K17-1032*

Ratinov, L. and Roth, D. (2009), Design challenges and misconceptions in named entity recognition, *in* 'Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)', Association for Computational Linguistics, pp. 147–155.
URL: *http://aclweb.org/anthology/W09-1119*

Řehůřek, R. and Sojka, P. (2010), Software Framework for Topic Modelling with Large Corpora, *in* 'Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks', ELRA, Valletta, Malta, pp. 45–50. `http://is.muni.cz/publication/884893/en`.

Roberts, K., Demner-Fushman, D. and Tonning, J. M. (2017), Overview of the TAC 2017 adverse reaction extraction from drug labels track, *in* 'Proceedings of the 2017 Text Analysis Conference, TAC 2017, Gaithersburg, Maryland, USA, November 13-14, 2017'.

Rocktäschel, T., Huber, T., Weidlich, M. and Leser, U. (2013), Wbi-ner: The impact of domain-specific features on the performance of identifying and classifying mentions of drugs, *in* 'Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)',

Association for Computational Linguistics, pp. 356–363.
URL: *http://aclweb.org/anthology/S13-2058*

Rocktäschel, T., Weidlich, M. and Leser, U. (2012), 'Chemspot: a hybrid system for chemical named entity recognition', *Bioinformatics* **28**(12), 1633–1640.

Rodríguez-Terol, A., Caraballo, M., Palma, D., Santos-Ramos, B., Molina, T., Desongles, T. and Aguilar, A. (2009), 'Calidad estructural de las bases de datos de interacciones', *Farmacia Hospitalaria* **33**(3), 134–146.

Rosenblatt, F. (1958), 'The perceptron: A probabilistic model for information storage and organization in the brain', *Psychological Review* **65**(6), 386–408.

Rotsztejn, J., Hollenstein, N. and Zhang, C. (2018), 'Eth-ds3lab at semeval-2018 task 7: Effectively combining recurrent and convolutional neural networks for relation classification and extraction', pp. 689–696.
URL: *http://aclweb.org/anthology/S18-1112*

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986), Learning representations by back-propagating errors, Vol. 323, Nature Publishing Group, p. 533.

Sadikin, M., Fanany, M. I. and Basaruddin, T. (2016), 'A new data representation based on training data characteristics to extract drug name entity in medical text', *Computational intelligence and neuroscience* **2016**, 6.

Sagae, K. and Tsujii, J. (2007), Dependency parsing and domain adaptation with lr models and parser ensembles, *in* 'Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)'.
URL: *http://aclweb.org/anthology/D07-1111*

Sahu, S. K. and Anand, A. (2018), 'Drug-drug interaction extraction from biomedical texts using long short-term memory network', *Journal of Biomedical Informatics* **86**, 15 – 24.
URL: *http://www.sciencedirect.com/science/article/pii/S1532046418301606*

Samuel, A. L. (1959), 'Some studies in machine learning using the game of checkers', *IBM Journal of Research and Development* **3**(3), 210–229.

Sarker, A. and Gonzalez, G., eds (2017), *Proceedings of the 2nd Social Media Mining for Health Research and Applications Workshop co-located with the American Medical Informatics Association Annual Symposium (AMIA 2017), Washington D.C., United States, November 4, 2017*, Vol. 1996 of *CEUR Workshop Proceedings*, CEUR-WS.org.

Segura-Bedmar, I., Martínez, P. and Herrero Zazo, M. (2013), Semeval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013), *in* 'Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)', Association for Computational Linguistics, pp. 341–350.
URL: *http://aclweb.org/anthology/S13-2056*

Segura-Bedmar, I., Martínez, P. and Herrero-Zazo, M. (2014), 'Lessons learnt from the DDIExtraction-2013 Shared Task', *Journal of Biomedical Informatics* **51**, 152 – 164.

Segura-Bedmar, I., Martínez, P. and Sanchez-Cisneros, D. (2011), The 1st DDIExtraction-2011 Challenge Task: Extraction of Drug-Drug Interactions from Biomedical Texts, *in* 'Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction 2011', pp. 1–9.

Segura-Bedmar, I., Suárez-Paniagua, V. and Martínez, P. (2015*a*), Combining conditional random fields and word embeddings for the chemdner-patents task, *in* 'Proceedings of the Fifth BioCreative Challenge Evaluation Workshop (BIOCREATIVE V)'.

Segura-Bedmar, I., Suárez-Paniagua, V. and Martínez, P. (2015*b*), Exploring word embedding for drug name recognition, *in* C. Grouin, T. Hamon, A. Névéol and P. Zweigenbaum, eds, 'Proceedings of the Sixth International Workshop on Health Text Mining and Information Analysis, Louhi@EMNLP 2015, Lisbon, Portugal, September 17, 2015', Association for Computational Linguistics, pp. 64–72.
URL: *https://doi.org/10.18653/v1/W15-2608*

Settles, B. (2005), 'Abner: an open source tool for automatically tagging genes, proteins and other entity names in text', *Bioinformatics* **21**(14), 3191–3192.
URL: *http://dx.doi.org/10.1093/bioinformatics/bti475*

Sha, F. and Pereira, F. (2003), Shallow parsing with conditional random fields, *in* 'Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1', Association for Computational Linguistics, pp. 134–141.

Shapiro, S. C. (1992), *Encyclopedia of Artificial Intelligence*, 2nd edn, John Wiley & Sons, Inc., New York, NY, USA.

Socher, R., Bauer, J., Manning, C. D. and Andrew Y., N. (2013), Parsing with compositional vector grammars, *in* 'Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Association for Computational Linguistics, pp. 455–465.
URL: *http://aclweb.org/anthology/P13-1045*

Socher, R., Huval, B., Manning, C. D. and Ng, A. Y. (2012), 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', Association for Computational Linguistics, pp. 1201–1211.
URL: *http://aclweb.org/anthology/D12-1110*

Socher, R., Lin, C. C.-Y., Ng, A. Y. and Manning, C. D. (2011), Parsing natural scenes and natural language with recursive neural networks, *in* 'Proceedings of the 28th International Conference on International Conference on Machine Learning', ICML'11, Omnipress, USA, pp. 129–136.
URL: *http://dl.acm.org/citation.cfm?id=3104482.3104499*

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y. and Manning, C. D. (2011), Semi-supervised recursive autoencoders for predicting sentiment distributions, *in* 'Proceedings of the Conference on Empirical Methods in Natural Language Processing', EMNLP '11, Asso-

ciation for Computational Linguistics, Stroudsburg, PA, USA, pp. 151–161.
URL: *http://dl.acm.org/citation.cfm?id=2145432.2145450*

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. and Potts, C. (2013), Recursive deep models for semantic compositionality over a sentiment treebank, *in* ‘Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing’, Association for Computational Linguistics, pp. 1631–1642.
URL: *http://aclweb.org/anthology/D13-1170*

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S. and Tsujii, J. (2012), Brat: a web-based tool for nlp-assisted text annotation, *in* ‘Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics’, Association for Computational Linguistics, pp. 102–107.

Stricker, B. and Psaty, B. (2004), ‘Detection, verification, and quantification of adverse drug reactions’, *British Medical Journal* **329**(7456), 44–47.

Suárez-Paniagua, V. and Segura-Bedmar, I. (2016), Using recursive neural networks to detect and classify drug-drug interactions from biomedical texts, *in* ‘ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)’, pp. 1666–1667.
URL: *https://doi.org/10.3233/978-1-61499-672-9-1666*

Suárez-Paniagua, V. and Segura-Bedmar, I. (2018), ‘Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction’, *BMC Bioinformatics* **19-S**(8), 39–47.
URL: *https://doi.org/10.1186/s12859-018-2195-1*

Suárez-Paniagua, V., Segura-Bedmar, I. and Martínez, P. (2015), Word embedding clustering for disease named entity recognition, *in* ‘Proceedings of the Fifth BioCreative Challenge Evaluation Workshop (BIOCREATIVE V)’.

Suárez-Paniagua, V., Segura-Bedmar, I. and Martínez, P. (2017), 'Exploring convolutional neural networks for drug-drug interaction extraction', *Database* **2017**, bax019.
URL: *https://doi.org/10.1093/database/bax019*

Sun, X., Dong, K., Ma, L., Sutcliffe, R., He, F., Chen, S. and Feng, J. (2019), 'Drug-drug interaction extraction via recurrent hybrid convolutional neural networks with an improved focal loss', *Entropy* **21**(1).
URL: *http://www.mdpi.com/1099-4300/21/1/37*

Sun, X., Ma, L., Du, X., Feng, J. and Dong, K. (2018), Deep convolution neural networks for drug-drug interaction extraction, *in* '2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)', pp. 1662–1668.

Sutton, C., McCallum, A. et al. (2012), 'An introduction to conditional random fields', *Foundations and Trends® in Machine Learning* **4**(4), 267–373.

Tai, K. S., Socher, R. and Manning, C. D. (2015), 'Improved semantic representations from tree-structured long short-term memory networks', pp. 1556–1566.
URL: *http://aclweb.org/anthology/P15-1150*

Taulé, M., Martí, M. A. and Recasens, M. (2008), Ancora: Multilevel annotated corpora for catalan and spanish, *in* 'LREC 2008'.
URL: *http://www.lrec-conf.org/proceedings/lrec2008/pdf/35_paper.pdf*

TH, M., Sahu, S. and Anand, A. (2015), 'Evaluating distributed word representations for capturing semantics of biomedical concepts', pp. 158–163.
URL: *http://aclweb.org/anthology/W15-3820*

Tiedemann, J. and Nygaard, L. (2004), The opus corpus - parallel & free, *in* 'Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)', Lisbon, Portugal, pp. 765–771.

Trask, A., Michalak, P. and Liu, J. (2015), 'sense2vec - A fast and accurate method for word sense disambiguation in neural word embeddings', *CoRR* **abs/1511.06388**.
URL: *http://arxiv.org/abs/1511.06388*

Trunk, G. V. (1979), 'A problem of dimensionality: A simple example', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-1**(3), 306–307.

Tsuruoka, Y., Tateishi, Y., Kim, J.-D., Ohta, T., McNaught, J., Ananiadou, S. and Tsujii, J. (2005), Developing a robust part-of-speech tagger for biomedical text, *in* 'Panhellenic Conference on Informatics', Springer, pp. 382–392.

Tu, J. C., Lai, P.-T. and Tsai, R. T.-H. (2017), Enhancing drug-drug interaction classification with corpus-level feature and classifier ensemble, *in* 'Proceedings of the International Workshop on Digital Disease Detection using Social Media 2017 (DDDSM-2017)', Association for Computational Linguistics, pp. 52–56.
URL: *http://aclweb.org/anthology/W17-5808*

Turian, J., Ratinov, L. and Bengio, Y. (2010), Word representations: A simple and general method for semi-supervised learning, *in* 'Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics', ACL '10, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 384–394.
URL: *http://dl.acm.org/citation.cfm?id=1858681.1858721*

Unanue, I. J., Borzeshi, E. Z. and Piccardi, M. (2017), 'Recurrent neural networks with specialized word embeddings for health-domain named-entity recognition', *Journal of biomedical informatics* **76**, 102–109.

van Der Hooft, C. S., Sturkenboom, M. C., van Grootheest, K., Kingma, H. J. and Stricker, B. H. C. (2006), 'Adverse drug reaction-related hospitalisations', *Drug Safety* **29**(2), 161–168.

Voorhees, E. M. (2013), The trec medical records track, *in* 'Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics', BCB'13, ACM, New York, NY, USA, pp. 239:239–239:246.
URL: *http://doi.acm.org/10.1145/2506583.2506624*

Wang, L., Cao, Z., de Melo, G. and Liu, Z. (2016*a*), Relation classification via multi-level attention cnns, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Association for Computational Linguistics,

pp. 1298–1307.
URL: *http://aclweb.org/anthology/P16-1123*

Wang, L., Cao, Z., de Melo, G. and Liu, Z. (2016*b*), Relation classification via multi-level attention cnns, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Association for Computational Linguistics, pp. 1298–1307.
URL: *http://www.aclweb.org/anthology/P16-1123*

Wang, P., Xu, B., Xu, J., Tian, G., Liu, C.-L. and Hao, H. (2016), 'Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification', *Neurocomputing* **174, Part B**, 806 – 814.

Wang, W., Yang, X., Yang, C., Guo, X., Zhang, X. and Wu, C. (2017), 'Dependency-based long short term memory network for drug-drug interaction extraction', *BMC Bioinformatics* **18**(16), 578.
URL: *https://doi.org/10.1186/s12859-017-1962-8*

Wei, C.-H., Peng, Y., Leaman, R., Davis, A. P., Mattingly, C. J., Li, J., Wiegers, T. C. and Lu, Z. (2015), Overview of the biocreative v chemical disease relation (cdr) task, *in* 'Proceedings of the fifth BioCreative challenge evaluation workshop', pp. 154–166.

Weininger, D. (1988), 'Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules', *Journal of Chemical Information and Computer Sciences* **28**(1), 31–36.
URL: *https://pubs.acs.org/doi/abs/10.1021/ci00057a005*

Wishart, D. S., Knox, C., Guo, A. C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z. and Woolsey, J. (2006), 'Drugbank: a comprehensive resource for in silico drug discovery and exploration', *Nucleic acids research* **34**(suppl 1), D668–D672.

Xiao, M. and Liu, C. (2016), Semantic relation classification via hierarchical recurrent neural network with attention, *in* 'Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers', The COLING 2016 Organizing Committee,

pp. 1254–1263.
URL: *http://aclweb.org/anthology/C16-1119*

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

Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H. and 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', Association for Computational Linguistics, pp. 1785–1794.
URL: *http://aclweb.org/anthology/D15-1206*

Yampolskiy, R. V. (2013), Turing test as a defining feature of ai-completeness, *in* 'Artificial Intelligence, Evolutionary Computing and Metaheuristics', Vol. 427, Springer, Berlin, Heidelberg.

Yi, Z., Li, S., Yu, J., Tan, Y., Wu, Q., Yuan, H. and Wang, T. (2017), 'Drug-drug interaction extraction via recurrent neural network with multiple attention layers', pp. 554–566.

Zeiler, M. D. (2012), 'ADADELTA: an adaptive learning rate method', *CoRR* **abs/1212.5701**.
URL: *http://arxiv.org/abs/1212.5701*

Zeng, D., Liu, K., Lai, S., Zhou, G. and Zhao, J. (2014), Relation classification via convolutional deep neural network, *in* 'Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014), Technical Papers', Dublin City University and Association for Computational Linguistics, Dublin, Ireland, pp. 2335–2344.

Zeng, D., Sun, C., Lin, L. and Liu, B. (2016), Enlarging drug dictionary with semi-supervised learning for drug entity recognition, *in* 'Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on', IEEE, pp. 1929–1931.

Zeng, D., Sun, C., Lin, L. and Liu, B. (2017), 'Lstm-crf for drug-named entity recognition', *Entropy* **19**(6), 283.

Zhang, Y., Lin, H., Yang, Z., Wang, J., Zhang, S., Sun, Y. and Yang, L. (2018), 'A hybrid model based on neural networks for biomedical relation extraction', *Journal of Biomedical Informatics* **81**, 83 – 92.
URL: *https://www.sciencedirect.com/science/article/pii/S1532046418300534*

Zhang, Y., Zheng, W., Lin, H., Wang, J., Yang, Z. and Dumontier, M. (2018), 'Drug–drug interaction extraction via hierarchical rnns on sequence and shortest dependency paths', *Bioinformatics* **34**(5), 828–835.
URL: *http://dx.doi.org/10.1093/bioinformatics/btx659*

Zhao, Z., Yang, Z., Luo, L., Lin, H. and Wang, J. (2016), 'Drug drug interaction extraction from biomedical literature using syntax convolutional neural network', *Bioinformatics* **32**(22), 3444–3453.
URL: *http://dx.doi.org/10.1093/bioinformatics/btw486*

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

Zheng, W., Lin, H., Zhao, Z., Xu, B., Zhang, Y., Yang, Z. and Wang, J. (2016), 'A graph kernel based on context vectors for extracting drug–drug interactions', *Journal of Biomedical Informatics* **61**, 34 – 43.
URL: *http://www.sciencedirect.com/science/article/pii/S1532046416300053*

Zhou, D., Miao, L. and He, Y. (2018), 'Position-aware deep multi-task learning for drug–drug interaction extraction', *Artificial Intelligence in Medicine* **87**, 1 – 8.
URL: *http://www.sciencedirect.com/science/article/pii/S0933365717306310*

Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H. and Xu, B. (2016), Attention-based bidirectional long short-term memory networks for relation classification, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)', Association for Computational Linguistics, pp. 207–212.
URL: *http://aclweb.org/anthology/P16-2034*