UNIVERSIDAD POLITÉCNICA DE VALENCIA

Maximal Frequent Sequences Applied to Drug-Drug Interaction Extraction

Sandra García Blasco

Supervised by:

Paolo Rosso
Universidad Politécnica de Valencia

Roxana M. Danger
Imperial College London

A M.Sc. Thesis Presented in Partial Fulfillment of the Requirements for the Degree Master in Artificial Intelligence, Pattern Recognition and Digital Image

Departamento de Sistemas Informáticos y Computación
February 2012

Abstract

A drug-drug interaction (DDI) occurs when the effects of a drug are modified by the presence of other drugs. DDIs can decrease therapeutic benefit or efficacy of treatments and this could have very harmful consequences in the patient's health that could even cause the patient's death. Knowing the interactions between prescribed drugs is of great clinical importance; it is very important to keep databases up-to-date with respect to new DDI.

In this thesis we aim to build a system to assist healthcare professionals to be updated about published drug-drug interactions. The goal of this thesis is to study a method based on maximal frequent sequences (MFS) and machine learning techniques in order to automatically detect interactions between drugs in pharmacological and medical literature. With the study of these methods, the information technology community will assist healthcare community to update their drug interactions database in a fast and semi-automatic way.

In a first solution, we classify pharmacological sentences depending on whether or not they are describing a drug-drug interaction. This would enable to automatically find sentences containing drug-drug interactions. This solution is completely based on maximal frequent sequences extracted from a set of test documents.

In a second solution based on machine learning, we go further in the search and perform DDI extraction, determining whether two specific drugs appearing in a sentence interact or not. This can be used as an assisting tool to populate databases with drug-drug interactions. The machine learning classifier is trained with several features: bag of words, word categories, MFS, token and char level features, as well as drug level features. We used a Random Forest classifier. With this system we participated at the DDIExtraction 2011 competition, where we obtained 6th position.

Finally, we introduce Maximal Frequent Discriminative Sequences (MFDS), a new method for sequential pattern discovery that extends the concept of MFS to adapt it to classification tasks.

Resumen

Una interacción entre fármacos (*drug-drug interaction*, DDI) ocurre cuando los efectos de un fármaco son modificados por la presencia de otros fármacos. Las DDIs pueden disminuir el beneficio terapéutico o eficacia de los tratamientos y pueden tener consecuencias muy graves para la salud del paciente que podrían incluso llegar causar su muerte. Conocer las interacciones entre los fármacos recetados a un mismo paciente es de vital importancia clínica. Es crucial mantener las bases de datos actualizados con respecto a nuevas interacciones.

El objetivo de esta tesis es construir un sistema para ayudar a los profesionales de la salud a estar actualizados respecto a las nuevas interacciones entre fármacos. En esta tesis estudiamos un método basado en secuencias frecuentes maximales (*maximal frequent sequences*, MFS) y técnicas de aprendizaje automático para detectar automáticamente interacciones entre fármacos en la literatura médica y farmacológica. Con el estudio de estos métodos, la comunidad de las tecnologías de la información podrá ayudar a la comunidad médica a actualizar sus bases de datos de interacciones de una forma rápida y semi-automática.

En nuestra primera aproximación, clasificamos frases extraídas de textos farmacológicos según si incluyen o no la descripción de una interacción entre fármacos. Esto permitirá encontrar automáticamente frases que contengan DDIs. Esta solución está completamente basada en secuencias frecuentes maximales.

En nuestra segunda aproximación, basada en aprendizaje automático, vamos más allá en la búsqueda y realizamos extracción de fármacos que interactúan. Esto es, determinamos si dos fármacos específicos que aparecen en una frase interactúan o no. Esto puede ser usado como herramienta de asistencia para poblar bases de datos con interacciones entre fármacos. El clasificador que hemos construido está entrenado con varios conjuntos de características describiendo cada frase: bolsa de palabras, categorías de palabras, MFS, características a nivel de carácter y token y características a nivel de fármaco. El clasificador usado fue Random Forest. Esta solución fue enviada a la competición DDIExtraction 2011, donde quedó en 6º lugar.

Por último, introducimos las Secuencias Discriminantes Frecuentes Maximales (*maximal frequent discriminative sequences*, MFDS), un nuevo concepto de patrones secuenciales que extiende el concepto de MFS para adaptarlo a tareas de clasificación.

Contents

List of Tables

List of Figures

List of Algorithms

Acknowledgements

First of all, I would like to thank Paolo Rosso and Roxana Danger for their wise comments with respect to my work. To Santiago M. Mola for his invaluable help when defining MFDS and for a whole year of discussions about this thesis.

Thanks to all the professors that taught me all I know and inspired me during many years of education.

I would also like to thank Mirko Degli Esposti, head of the Mathematics Department of the University of Bologna, and his lovely family, for making my stay in Bologna a great experience.

Thanks to Isabel Segura-Bedmar and Paloma Martínez for the DrugDDI corpus and for organizing the DDI Extraction Challenge 2011.

Computational resources for the DDI Extraction Challenge 2011 submission were kindly provided by Daniel Kuehn from Data@UrService.

This work would not have been possible without the financial support of Bitsnbrains S.L. This work is also partially supported by the MICINN research project TEXT-ENTERPRISE 2.0 TIN2009-13391-C04-03 (Plan I+D+i). This thesis is within the VLC/CAMPUS Microcluster on Multimodal Interaction in Intelligent Systems.

Los caminos duros siempre son más fáciles de recorrer si tienes a alguien andando a tu lado. Yo he tenido la suerte de contar con Santiago desde el principio. Él ha sido mi crítico más duro y a la vez mi mayor apoyo. Gracias por ilusionarte con mis proyectos tanto como con los tuyos.

A mi padre, por siempre desafiar mi mente. A mi madre, por su apoyo y cariño incondicionales. A Fran, por ser mi punto de apoyo y hacerme ver siempre el lado positivo. A mi querida hermana Marta, por siempre creer en mi y demostrarme que, trabajando duro, todo es posible.

Chapter 1

Introduction

1.1 Motivation

Drugs can have adverse effects. Pharmacological companies know most of the adverse effects a drug can produce before they market it. Nevertheless, patients on multiple medications can experience unexpected adverse events, caused by the co-administration of several drugs and these are known as drug-drug interactions.

A drug-drug interaction (DDI) occurs when the effects of a drug are modified by the presence of other drugs. Following Stockley (2007),

> "A DDI occurs when one drug influences the level or activity of another, possibly intensifying its side effects or decreasing drug concentrations and thereby reducing its effectiveness."

Some patients require many drug prescriptions at the same time, and therefore are at risk of being affected by DDIs. When co-administrating drugs, the *effective dose* can vary considerably form the expected one. Drug-drug interactions are a serious problem when talking about patient safety (Tatonetti et al., 2011). The proportion of Adverse Drug Reactions (ADRs) due to drug–drug interactions is estimated to be between 6% and 30% and surveillance on the safety profile of the interaction between drugs is challenging (Hauben and Zhou, 2003; Pirmohamed and Orme, 1998).

DDIs can decrease therapeutic benefit or efficacy, leading to an increase of the duration of the patient's stay at the hospital, and its consequent increase of costs. DDIs could have very harmful consequences in the patient's health that could even

cause the patient's death. Knowing the interactions between prescribed drugs is of great clinical importance, therefore it is very important that healthcare professionals keep their databases up-to-date with respect to new DDI.

Every day, hundreds of medical and pharmacological papers are written, some of them describing new drug-drug interactions. MEDLINE is an online bibliographic database on biomedicine and health that contains over 18 million references to journal articles. Since 2005, between 2,000-4,000 completed references are added each day; 700,000 total added in 2010[1]. With this growing amount of articles, it is not possible for healthcare professionals to keep themselves updated with every new drug-drug interaction discovered, and this leaves very clear how necessary is to find efficient methods that help them to better deal with all this information. Therefore, we need a system to assist healthcare professionals retrieve information about new drug-drug interactions published in the literature. Information Extraction techniques aim to automatically extract relevant information from documents. Using these techniques, researches have shown that it is possible to automatically identify drug-drug interactions in texts.

Even though nowadays health care professionals have access to databases containing drug-drug interactions, these are not very exhaustive and their update periods can be as long as three years (Rodríguez Terol et al., 2009)[2].

1.2 Objectives

In this thesis we aim at building a system to assist health-care professionals to be updated about published drug-drug interactions. Pharmacists use a particular vocabulary to describe their DDIs discoveries. This vocabulary, or part of it, could be represented by a set of patterns used frequently in DDI publications. Maximal Frequent Sequences (MFS) can represent the patterns that are somehow repeated in the text. Therefore, MFS could be used to: 1) represent some parts of the vocabulary used for DDI description; 2) help improve the results of statistical models to represent that vocabulary.

The goal of this thesis is to study methods based on maximal frequent sequences and machine learning techniques in order to automatically detect, in

[1] These and other facts about Medline are available at `http://www.nlm.nih.gov/pubs/factsheets/medline.html`

[2] via (Segura-Bedmar, Martínez, and Sánchez-Cisneros, 2011)

pharmacological and medical literature, interactions between drugs. With the study of these methods, the Information Technology (IT) community will assist health care community to update their drug interactions database in a fast and semi-automatic way. For this, we have tackled two different solutions.

In a first solution, we will classify pharmacological sentences depending on whether or not they are describing a drug-drug interaction. This would enable health-care professionals to automatically discard texts that do not contain drug-drug interactions, and therefore decrease the number of documents they have to review.

In a second solution based on machine learning, we will go further in the search and perform DDI extraction, determining whether or not two specific drugs appearing in a sentence interact. This would be an assisting tool to populate databases with drug-drug interactions.

The main tool we will be using in our two systems is Maximal Frequent Sequences. Both systems share the following characteristics:

1) In order to make MFS more flexible, we will add a gap constraint, *i.e.*, allowing a certain distance between the words forming the pattern.

2) The systems are based on the application of supervised binary classification algorithms. This makes possible the evaluation of the performance of the systems.

The description of the rest of this thesis is detailed in Section 1.3.

1.3 Thesis Outline

This Thesis is organized as follows:

In Chapter 2 we define Maximal Frequent Sequences and carry out a survey of the most important pattern mining algorithms. Following, we modify the Generalized Sequential Pattern (GSP) algorithm to obtain a new algorithm to extract MFS. We also present a modification of the algorithm to adapt it to continuous events. Finally, we discuss the limitations of MFS and present a novel concept: Maximal Frequent Discriminative Sequences.

In Chapter 3 we define two problems related to DDI: DDI Sentence Identification and DDI Extraction, as well as the performance measures used for their evaluation. We review the most important contributions of other authors to the field,

as well as those that participated in the First Challenge Task: Drug Drug Interaction Extraction, a competition carried out to evaluate DDI Extraction systems. After describing briefly the systems submitted to the competition, a description of our system is given. Following, further modifications of the system submitted at the competition are presented. At the end, some conclusions are drawn.

Finally, Chapter 4 highlights the most important conclusions about our work and defines new lines of investigation to follow this research.

Chapter 2

A New Maximal Frequent Sequences Extraction Algorithm

Data mining can briefly be described as the "development of efficient algorithms for finding useful high-level knowledge from large amounts of data" (Fayyad et al., 1996). Pattern mining is a data mining problem that involves finding patterns in large amounts of data. Usually, these patterns define behaviours that are repeated and can be used for prediction. Pattern mining is a very extensive field, and it includes many algorithms.

Even thought the first pattern mining algorithms were to mine frequent itemsets, some of them evolved in order to cover other needs, such as the possible sequentiality of the items. The most known algorithm is the Apriori algorithm (Agrawal and Srikant, 1994). This algorithm was used to extract relations between collections of items, and it did not take into account the order in which the items appeared, but just their appearance.

Sequential Pattern Mining is a field of Pattern Mining that extends the problem definition by adding a sequentiality constraint to the patterns. This kind of algorithms are relevant when the data to mine has a sequential nature, for example when the data is composed of words that compose sentences. Section 2.3 describes the different types of sequential pattern mining algorithms and reviews the most important ones: Apriori, AprioriAll, GSP Algorithm, MineMFS (Ahonen) Algorithm, PrefixSpan, GenPrefixSpan, SPADE, cSPADE and DIMASP. The selection of important algorithms is based in the review presented in (García-Hernández, 2007).

The distribution of this Chapter is organized as follows. In Section 2.1 we review some definitions in order to understand Maximal Frequent Sequences. MFS are explained in detail in Section 2.2. In Section 2.3 we carry out a survey of the most important pattern mining algorithms. Following, in Section 2.4 we explain in detail a new algorithm to extract MFS, and we add an adaptation so it can be used with temporal events. In Section 2.5 we introduce a new concept, *Maximal Frequent Discriminative Sequences*, and detail the algorithm that can be used to extract them. Finally, in Section 2.6 we draw some conclusions about the algorithms described.

2.1 Definitions

In order to understand further sections, we first need to review some definitions.

Itemsets A collection of items that occur together without any specific order.

Sequences or Strings A sorted list of k elements with the form:

$$< p_1, p_2, p_3, \dots p_k >$$

Subsequences and Substrings A *subsequence* is a sequence derived from a given sequence by selecting certain of its items and respecting their order. A *substring* is a specific case of subsequence where all the items are consecutive.

Given a sequence <$p = p_1 \dots p_k$> and a sequence q where all the elements p_i appear in q and they do so in the same order in which they appear in p, then p is substring of q. In other words, p is substring of q if exists an integer i that satisfies:

$$p_1 = q_i,$$

$$p_2 = q_{i+1},$$

$$p_3 = q_{i+2},$$

$$\dots$$

$$p_n = q_{i+(n-1)}$$

Subsequences, unlike *substrings*, do not require the items to be consecutive, instead there is a maximum distance allowed between the items. Some authors call this distance *gap* (Agrawal et al., 1996; García-Hernández, 2007).

Frequent Sequences Being S a document collection, where each document consists in a sequence of words, a sequence $p \in S$ is frequent in the document collection S if p is subsequence of at least β documents $\in S$, where β is a given threshold. In this case, we say that p is a sequence β-frequent, or we simplify it by saying that p is frequent.

Please, note that only one occurrence per document will be taken into account. The fact that one sequence appears more than once in the same document will not increase its frequency.

N-grams A n-gram is a contiguous sequence of n words from a given sequence of text. A 2-gram, also known as bigram, is a contiguous sequence of 2 words.

For example, given the sentence:

$$s =< w_1, w_2, w_3, w_4 >$$

The bigrams included in the sentence are:

$$[w_1, w_2], [w_2, w_3], [w_3, w_4]$$

Skip-grams A skip-gram is a sequence of words from a given sequence of text that allows a gap between the words. A k-skip-n-gram is a sequence of n words that allow a maximum gap of k words.

For example, given the sentence:

$$s =< w_1, w_2, w_3, w_4 >$$

The 2-skip-2-grams included in the sentence are:

$$[w_1, w_2], [w_2, w_3], [w_3, w_4]$$
$$[w_1, w_3], [w_2, w_4]$$
$$[w_1, w_4]$$

2.2 Maximal Frequent Sequences

Following the definitions described previously, Maximal Frequent Sequences are defined as follows:

Maximal Frequent Sequence A sequence p is a Maximal Frequent Sequence in S if p is frequent in S and does not exist any other sequence p' in S such as p is subsequence of p' and p' is frequent in S.

MFS are an interesting tool since they can represent the most important parts of texts. Given a text collection, the fact that there are sequences that are repeated in some of the texts shows how relevant is the information that those MFS describe. Also, we must point out the wide applicability of the extraction of MFS since the technique is domain and language independent. The fact that they are sequences and not strings, *i.e.*, they allow gap between words, makes them more flexible and therefore they can capture higher level patterns.

The *gap* is the maximum distance that is allowed between two words of a MFS. Following this, if we set the *gap* to 0, the word in the MFS will be adjacent words in the original text. For example,

$$< w_{i_0}, \ldots, w_{i_n} >, \text{with } i_j \in 1 \ldots k,$$

is a maximal frequent sequence of k words,

$$i_j \leq i_{j-1} + \eta + 1, \text{ when } gap = \eta., \text{ and therefore } i_j = i_{j-1} + 1,\ j > 1, \text{ when } gap = 0$$

For example, given a sentence collection S containing s_1, s_2 and s_3:

Sentence s_1 A drug may increase the effects of other drugs.

Sentence s_2 A given drug may potentiate the side effects of other drugs.

Sentence s_3 Concomitant administration of some drug may lower the desirable effects of other drugs.

Setting minimum frequency (β) to 3 and setting the maximum gap allowed between items to 1, we obtain the following MFS:

<'drug', 'may', 'the', 'effects', 'of', 'other', 'drugs'>

The sequences *('effects', 'of', 'other','drugs')* is also frequent, but since we are searching only for maximal sequences and it is included in a larger sequence, the shorter one is discarded.

MFS have shown to be useful in different tasks such as document clustering (Hernández-Reyes et al., 2006a), text summarization (Ledeneva, Gelbukh, and García-Hernández, 2008), document representation (Hernández-Reyes et al., 2006b), measuring text similarities (García-Blasco, 2009) and authorship attribution (Coyotl-Morales et al., 2006).

Our hypothesis holds that MFS will be a good tool to capture frequent sentence structures (patterns) used by pharmacologist to define DDIs. Since we will be working with texts, allowing a gap between the words will make these patterns more flexible, instead of limiting the search to exact same sentences.

2.3 Related Work

During the last decade, several sequential pattern discovery algorithms have been presented. These algorithms can be divided, according to their search method, into *bottom-up* and *top-down*. The *bottom-up* algorithms use the sequences of length $k-1$ to build sequences of length k, *i.e.*, they go from bottom to top, finding first the shortest sequences and building longer ones upon them. The A priori algorithm (Agrawal and Srikant, 1994), as well as those classified as *pattern growing* algorithms are *bottom-up* algorithms. The *top-down* algorithms search directly long patterns, avoiding having to search for the short ones, for example the SPLMiner algorithm (Seno and Karypis, 2002).

Most of the algorithms for sequential pattern discovery fall into the *bottom-up* algorithms class. Nevertheless, there exist differences between them depending on how they find longer patterns based on the shorter ones. We can divide this kind of algorithms into *a priori* algorithms and *pattern growing* algorithms.

A priori These methods use the information of *k*-length patterns to find the *k+1*-length patterns, *i.e.*, they use the previous information for the following step. They start from the 1-length patterns and keep building longer patterns. This kind of algorithms usually generate candidate patterns and after check in the database if their frequency is still above the threshold. In order to build the patterns with length k, this family of algorithms finds possible candidates by merging the $k-1$ patterns with $k-2$ equal elements. For example, being two patterns $p = p_1, p_2, p_3$ and $q = q_1, q_2, q_3$, the algorithm will merge p and q to form a new candidate pattern only if $p_2 = q_1$ and $p_3 = q_2$, and will do so by linking q_3 to p. The new pattern z will be of the form: $z = p_1, p_2, p_3, q_3$. Samples of these algorithms are the Apriori algorithm (Agrawal and Srikant, 1994) and the Generalized Sequential Pattern (GSP) algorithm (Agrawal et al., 1996).

Pattern growing A difference of the *A priori* algorithms, *pattern growing* algorithms do not generate candidate patterns. After building a structure that represents the documents, these algorithms build the maximal frequent sequences navigating through it. Samples of these algorithms are the MineMFS algorithm (Ahonen-Myka, 1999), PrefixSpan (Mortazavi-Asl et al., 2004; Pei et al., 2001), GenPrefixSpan (Antunes and Oliveira, 2003), SPADE (Zaki, 2001), cSPADe (Zaki, 2000) and DIMASP (García-Hernández, Martínez-Trinidad, and Carrasco-Ochoa, 2004).

Yang (2006) described the problem of sequential pattern mining as NP-hard. In order to find a Maximal Frequent Sequence of length k, any algorithm would have to review $2^m - 1$ combinations of elements, having to check the frequency of each one of the candidate sequences.

We are interested in extracting Maximal Frequent Sequences from a document collection. We want to be able to have a gap constraint, *i.e.*, allowing certain flexibility between the elements of the MFS. Most of the algorithms did not contemplate the gap constraint at their first definition, but efforts have been made afterwards to add it.

In the following sections, we will briefly describe the most important algorithms, and their modifications to allow the gap constraint. We are specially interested in the GSP algorithm (Agrawal et al., 1996) since it is the one we implemented, with some modifications, to extract MFS from the document collection.

2.3.1 Apriori Algorithm

The Apriori algorithm (Agrawal and Srikant, 1994) is the most well-known and influential algorithm for extracting frequent itemsets. It works on a *botton-up* fashion. Given a frequency threshold β, the Apriori algorithm retrieves all the itemsets that appear at least β times. For doing so, the algorithm first calculates the frequency of each item. With the list of frequent items, builds a list of possible pairs of items, and then calculates their frequencies, keeping only the ones that are β-frequent. In other words, for each iteration, it uses the β-frequent k-itemsets (itemsets with size k) to find the β-frequent $k+1$-itemsets. The process stops when no more β-frequent itemsets are found. This algorithm does not take into account the sequentiality of the items.

The Apriori algorithm significantly reduces the search space with the *Apriory Property*, defined as follows:

Apriori Property If an itemset p is not frequent, then for any item q, $p \cup q$ is not frequent for any p. In other words, no superset of p, *i.e.*, itemsets containing p, will be frequent.

Gunopulos et al. (2003) analyze the Apriori algorithm, proving that it is optimal when the search is done within a small search space, *i.e.*, a small amount of documents. However, the problem comes when working with large datasets.

The Apriori algorithm and its variations have been successfully used to solve data mining problems (Agrawal and Srikant, 1994; Mannila and Toivonen, 1997; Mannila, Toivonen, and Verkamo, 1994, 1995).

To adapt the Apriori algorithm to the sequential pattern mining problem, Agrawal et al. (1996) propose two new algorithms: the AprioriAll algorithm and the Generalized Sequential Pattern (GSP) algorithm. Even thought AprioriAll and GSP both allow to retrieve the sequential patterns, the GSP algorithm is an evolution of the AprioriAll algorithm and it allows gap constraints.

2.3.2 Generalized Sequential Pattern Algorithm

Generalized Sequential Pattern (GSP) algorithm (Agrawal et al., 1996) is a generalization of the Apriori (Agrawal and Srikant, 1994) algorithm whose main property is handling sequential patterns. It also integrates the following concepts:

- **Taxonomies**: Given a directed acyclic graph of itemsets, describing an *is-a* hierarchy, and given two sequences s and r, for the purpose of determining if s is a subsequence of r, we will consider s_i equal to s_j if the latter is an ancestor of the former.

- **Sliding windows**: A data-sequence contributes to the support of a sequence only if it appears inside a time interval, known as sliding window.

- **Time constratints**: Two thresholds are defined, **min gap** and **max gap**. In order to consider two elements consecutive, their time difference should be between min and max gap.

GSP overall method can be summarized as follows. Given a number of β-frequent k-sequences of itemsets, it will generate $(k+1)$-sequences with the following steps:

1. **Join** all contiguous k-sequences to obtain $(k+1)$ sequences.

2. **Prune** candidate $(k+1)$-sequences that do not have enough *support* (*i.e.*, the sequence does not appear in the database more than a threshold).

While the main procedure of the GSP algorithm and the Apriori algorithm is the same, the GSP algorithm, unlike the Apriori algorithm, handles sequential patterns and it allows gaps between the items of the sequences.

2.3.3 PrefixSpan and GenPrefixSpan

PrefixSpan (Mortazavi-Asl et al., 2004; Pei et al., 2001) constructs recursively the patterns with the help of projected databases. An α-projected database is the set of subsequences in the database that follow α, *i.e.*, that are suffixes of sequences that have prefix α. This reduces the search space in each step.

PrefixSpan does not accept gap constraints, and it was adapted for this task by Antunes and Oliveira (2003), becoming GenPrefixSpan. The spirit of the algorithm is the same, but it redefines the method used to construct the projected databases. Instead of looking only for the first occurrence of the item, every occurrence is considered.

The problem with these two algorithms is that they need to do as many projections of the database as frequent sequences are found. However, they are

depth-first traversal algorithms, which makes not necessary having all projected databases in memory at a time. Also, the search space is reduced at each projection, which makes the algorithm faster.

2.3.4 SPADE and cSPADE

Sequential Pattern Discovery using Equivalence classes, SPADE (Zaki, 2001), is an algorithm for discovering the set of all frequent sequences. The algorithm uses a vertical id-list database format, where each sequence is associated to the list of documents where it appears, along with their positions. The algorithm decomposes the original search space into smaller pieces using lattice theory, that can be processed independently in main-memory, reducing I/O costs by reducing database scans. Also, it allows two different search strategies for enumerating the frequent sequences: breath-first and depth-first search, which can minimizes computational costs.

2.3.5 MineMFS

The basic idea of MineMFS (Ahonen-Myka, 1999, 2002) is to combine bottom-up and greedy methods. This avoids generating all the frequent subsequences of the maximal frequent sequences. On the one hand, maximal sequences are constructed from shorter sequences, on the other hand a frequent sequence that is not contained in any known maximal sequence is expanded until the longer sequence is not frequent anymore.

Starting from a set of frequent pairs, the algorithm takes a pair and adds an item to it, in a greedy manner, until the longer sequence is no more frequent. In the same way, the algorithm goes through all the pairs, but it only tries to expand a pair if it is not already a subsequence of some maximal sequence, in order to avoid the same maximal sequence being discovered more than once. When all the pairs are processed, every pair belongs to some maximal sequence. If some pair can not be expanded, it is itself a maximal sequence. The same process is repeated with all the frequent sequences that the algorithm finds. Any frequent sequence that is not contained in any known maximal sequence is expanded until the longer sequence is not frequent anymore.

2.3.6 DIMASP

The DIMASP algorithm (García-Hernández, Martínez-Trinidad, and Carrasco-Ochoa, 2004; García-Hernández, Martínez Trinidad, and Carrasco-Ochoa, 2006; García-Hernández, 2007) follows the pattern-growth strategy where small frequent sequences are found first with the objective of growing them to obtain MFSs.

In their work, the authors present two versions of the DIMASP algorithm, with and without gap constraint. We will focus on the version with the gap constraint DIMASP-C_n, where n is the maximum gap allowed.

DIMASP is divided in 4 stages:

Stage 1: Documents Transformation. Pairs of words are extracted from the original documents. Each pair of word is assigned with an id. Each pair of words is composed of two consecutive words, taking into account the gap constraints. This is, given the document <$w_1, w_2, w_3, \ldots, w_n$>, and setting the $gap = 1$, the pair of words that would be taken into account would be: <w_1, w_2>, <w_1, w_3>, <w_2, w_3>, <w_2, w_4>, ..., <w_{n-1}, w_n>, <w_{n-2}, w_n>.

Stage 2: Data Structure Building. With the pairs of words, a tree data structure is build with the characteristic that all pairs of words are linked and it is possible to reconstruct the document from the tree.

Stage 3: Frequent Sequences Search. Given a frequency threshold β and the gap constraint defined by the user, the algorithm searches for frequent sequences through the tree. For each pair of words, the longest frequent sequence that can be built out of the documents in the collection is stored.

Stage 4: Maximal Frequent Sequences Identification. Out of the frequent sequences found in Stage 3, the algorithm finds the ones that are maximal, *i.e.*, the maximal frequent sequences. This search is done fast using a prefix tree.

We need to point out that, since the algorithm does not have a minimum length constraint, every word that is β-frequent and it is not contained in any MFS is also returned by the algorithm as a MFS. This is done right after Stage 3 of the algorithm is completed.

2.4 A New Algorithm for MFS Extraction

In this section, a new algorithm to extract MFS is presented. The algorithm 1 is based on the GSP Algorithm (Agrawal et al., 1996), but with some differences. We are interested in extracting sequences of simple elements, *i.e.*, sequences of words, not itemsets as the GSP algorithm does, and we do not consider taxonomies. As the GSP algorithm, our algorithm will allow a gap between the elements of the found sequences, nevertheless we fixed the minimum gap to 0, *i.e.*, words could always be consecutive. Additionally, we want the sequences to be frequent and maximal, therefore the algorithm must discard any shorter frequent sequence included in a maximal one.

The algorithm takes as input the following parameters:

$\boldsymbol{docs}$ The collection of documents we want to extract MFS from.

$\boldsymbol{freq_{min}}$ The minimum frequency that the MFS must have, *i.e.*, minimum number of documents where it must appear. This parameter is equivalent to the *support count* parameter in GSP.

$\boldsymbol{length_{min}}$ The minimum length that the MFS must have, *i.e.*, minimum number of items in the MFS.

$\boldsymbol{gap}$ Maximum distance allowed between the items of the MFS, *i.e.*, maximum number of items that it is allowed to skip.

The output of the algorithm is a list of the MFS with information about which documents they appear in, as well as the positions of each item of the MFS in the referred documents.

In each iteration, the algorithm only keeps in memory the sequences of length k, and the MFS of length $< k$. Patterns will grow until there are no more patterns to merge. The algorithm can be divided into 3 stages: *Getting Skip-grams*, *Candidate Generation* and *Prune*.

2.4.1 Stage 1: Getting Skip-grams

In the first stage, the algorithm extracts all the gap-skip-bigrams.

The function **getSkipgrams**(gap,$list_{sentences}$,*n*) retrieves all the gap-skip-n-grams from the sentences in $list_{sentences}$. Each skip-gram contains information

about its frequency, as well as the documents, sentences and positions where it appears.

Only skip-grams with a frequency equal or greater than $freq_{min}$ are kept.

Algorithm 1: A new Maximal Frequent Sequences Extraction Algorithm.

Input: $docs$, $freq_{min}$, $length_{min}$, gap
Output: $list_{mfs}$

```
1  list_sentences ← split(docs)
2  skipbigrams ← getSkipgrams(gap,list_sentences, 2)
3  candidates_k ← {∀s ∈ skipgrams|s_freq ≥ min_freq}
4  list_mfs ← candidates_k
5  k ← 1
6  while candidates_k ≠ ∅ do
7      k ← k + 1
8      foreach c1 ∈ candidates_k do
9          c1_prefix ← Prefix(c1,k − 1)
10         foreach c2 ∈ candidates_k do
11             if Sufix(c2,k − 1) = c1_prefix then
12                 candidate ← c2+ Sufix(c1,1)
13                 if candidate_freq ≥ min_freq then
14                     candidates_{k+1} ← candidates_{k+1} ∪ {candidate}
15                     toRemove ← toRemove ∪ {c1, c2}
16     list_mfs ← list_mfs ∩ {toRemove}
17     list_mfs ← list_mfs ∪ candidates_{k+1}
18     candidates_k ← candidates_{k+1}
19 foreach mfs ∈ list_mfs do
20     if mfs_length ≥ min_length then
21         list_mfs ← list_mfs ∪ mfs
22 return list_mfs
```

2.4.2 Stage 2: Candidate Generation

After stage 1, the algorithm has found all the skip-bigrams, *i.e.*, patterns of length 2, that appear in at least $freq_{min}$ sentences.

The *Candidate Generation* consists in growing patterns by merging them to make them maximal. The merge step is performed as follows:

In iteration k, each pair of patterns of length k that end and start, respectively, with the same $k-1$ sequence will be candidates to become a pattern with length

$k + 1$. The sentences that contain the new pattern will be the intersection of the sentences that contained each of the patterns that were merged.

For example, in the first iteration, we have patterns:

$S =$<**a,b,c**> in sentences (1,3,4,5)
$R =$<**b,c,f**> in sentences (1,2,3,5)

S and R are candidates to become a pattern $Q =$<**a,b,c,f**> with length 4. Pattern Q appears in sentences (1,3,5).

2.4.3 Stage 3: Prune

If the selected patterns to merge in the candidate generation step are frequent and therefore can be considered for the next iteration of candidate generation, the algorithm removes the two k-length patterns that generated the $(k + 1)$-length pattern. This is because, since they were merged, they are now contained in the new and longer pattern and we are only interested in maximal patterns.

Following the previous sample, if $freq_{min}$ is set to 2 or 3, then Q is a frequent pattern and S and R will be merged into Q and removed. Otherwise, if $freq_{min}$ is set to a value greater than 3, the candidate pattern would not be frequent and it would be discarded, keeping S and R as maximal frequent patterns.

At the end, the algorithm removes all mfs that do not fulfill the min_{length} restriction.

2.4.4 Adaptation to Continuous Events

The algorithm has been adapted to extract MFS from sequences of continuous events[1], *i.e.*, temporal events. In this case, the *position* of the events is a *timestamp*, and the input of the algorithm is a long document containing all the events, rather than a list of documents. The gap is also a *timestamp* and defines the maximum time that the algorithm allows between two consecutive events, *i.e.*, if an event of the MFS occurs in t_1, the following event can occur in t_2, if $t_2 \leq t_1 + gap$.

The biggest difference is that we go from element distributed in a discrete way to items distributed continuously with possible overlaps. The concept of

[1]We have applied this algorithm to time series internally at Bitsnbrains S.L `http://bitsnbrains.net`. However, we have not performed an evaluation with a public corpus yet.

document is eliminated, and in this case, the frequency of an event is not in how many documents it appears, but how many times it appears.

2.5 Maximal Frequent Discriminative Sequences Extraction

Sometimes maximal frequent sequences are more restrictive that we would like them to be. When setting the parameters for running the algorithm, we tend to set a low frequency threshold in order to find as many MFS as possible. This is good but just to some extent. When using MFS as input for a predictive model, it is not crucial to find the maximal sequences, but the most relevant. In this case, frequency, by itself, is not a good criterion to stop looking into longer sequences.

For example, we would like to classify sentences as describing a friendship. We have the following list of sentences:

s_1 Bob is best friend of Alice. (Positive)

s_2 Carol is best friend of Dave. (Positive)

s_3 Charlie is friend of Diana. (Positive)

s_4 Chuck has no friends. (Negative)

s_5 Mallory is a bad friend. (Negative)

If we extract MFS with $min_{freq} = 2$, $gap = 1$ and $min_{length} = 3$, we would extract *"is best friend of"*. If we decided to use this pattern to classify friendships, we could classify correctly s_1 and s_2 as positive, and s_4 and s_5 as negative. But we would classify s_3 incorrectly as negative.

If we had a criterion that extracted *"is friend of"* instead of *"is best friend of"*, we could have classified every sentence correctly.

In situations like this, by extracting MFS we could be loosing patterns that have a higher discriminative power. This problem has already been expressed in the literature (Karunaratne, 2011).

We modified the algorithm in order to introduce a discriminative power criteria, that will determine whether or not the growing of a sequence should continue. This would retrieve the sequences that have more discriminative power respect to

the corpus they are extracted from. The importance of a frequent sequence is hard to determine, but a good starting point is information gain.

Information Gain is defined as follows:

Information Gain (IG) Let *Attr* be the set of all attributes and Ex the set of all training examples, $value(x, a)$ with $x \in Ex$ defines the value of a specific example x for attribute $a \in Attr$, H specifies the entropy, and $|x|$ is the number of elements in the set x. The information gain for an attribute $a \in Attr$ is defined in Equation 2.1:

$$IG(Ex, a) = H(Ex) - \sum_{v \in values(a)} \frac{|\{x \in Ex | value(x,a) = v\}|}{|Ex|} \times H(\{x \in Ex | value(x, a) = v\} \quad (2.1)$$

Following this, the algorithm would change in the *Prune* step. Instead of looking if the merged $(k-1)$-length sequences still keep the min_{freq} condition, the algorithm would look now if the new k-length candidate sequence has more IG than the $(k-1)$-sequences that would have to be merged. If the new one has more IG, then the merge is done, discarding the shortest sequences. Otherwise, the merge is not done and the shortest sequences are kept.

Even in this algorithm, the output will not contain sequences that are subsequence of any other sequence retrieved by the algorithm, so the sequences will still be maximal.

The algorithm still needs a parameter to shorten the number of results returned. This parameter could either be a threshold for IG, under which all sequences are discarded or, as it is in our case, a min_{freq} constraint. There should be also some criteria to look up to $k + n$ length sequences for higher IG than k, even if $k + 1$ do not increase IG. We have performed experiments with the algorithm as described as an early proof-of-concept. Note that IG is only one of the possible criteria that could be used as discriminative power measure.

We name Maximal Frequent Discriminative Sequences (MFDS) the patterns that are extracted using this process.

Following with the previous example, with $min_{freq} = 2$, $min_{length} = 3$ and $gap = 1$, the following 1-skip-bigrams are generated: "is best", "is friend", "best of", "friend of".

In iteration $k = 2$, the candidate sequences to merge are the ones shown in Table 2.1. We can also observe in that table the result of the merging, *i.e.*, the candidate sequences for iteration $k = 3$.

Table 2.2 shows each, for each candidate sequence, the comparison of the IG values of $c1$, $c2$ and the merged candidate, as well as the actions performed by the algorithm in each case. If IG($c1$) or IG($c2$) is greater than IG($merged$) then, keep $c1$ and $c2$ and discard $merged$. Discard $c1$ and $c2$ and keep $merged$ in other case.

Since $min_{length} = 3$, the algorithm outputs:

"is friend of", "is best of", "is best friend".

As we can see, the best sequence "is friend of" has been found by the algorithm. This increases the number of sequences found and this should be addressed in further work.

Table 2.1: 1-skip-bigrams and merged sequences for sample, with their respective IG values.

Sequence	**IG**
iteration $k = 2$	
is best	0.29
best friend	0.29
is friend	0.67
best of	0.29
friend of	0.67
iteration $k = 3$	
is friend of	0.67
is best of	0.29
is best friend	0.29
best friend of	0.29

Table 2.2: Candidate sequences to be merged, their IG values and actions taken by the algorithm.

c1	**c2**	**merged**	**Case and Action**
is best	best of	is best of	IG(c1) = IG(c2) = IG(merged) Keep merged, discard c1 and c2.
is friend	friend of	is friend of	IG(c1) = IG(c2) = IG(merged) Keep merged, discard c1 and c2.
is best	best friend	is best friend	IG(c1) = IG(c2) = IG(merged) Keep merged, discard c1 and c2.
best friend	friend of	best friend of	IG(c1) = IG(merged) $<$ IG(c2) Keep c1 and c2, discard merged.

2.6 Conclusions

In this chapter we have presented a new algorithm for MFS extraction inspired in the GSP algorithm (Agrawal et al., 1996). The algorithm allows gaps between the items of the sequences. The gap makes the MFS more flexible.

We have further modified the algorithm to handle continuous events, going from items distributed in a discrete way to items distributed continuously. In this scenario, each item has a *timestamp* as a position, and therefore there could be overlaps.

After analyzing the limitations of MFS, we have modified the algorithm in order to introduce a discriminative power criteria, *e.g.* Information Gain, that determines whether or not the growth of a sequence should continue. With this novel method, we retrieve what we named as Frequent Discrimitative Sequences.

Chapter 3

MFS for Drug-Drug Interaction Extraction

In order to go beyond bag of words for DDI detection, we need to capture more complex patterns such as multi-word terms, or grammatical patterns. Our hypothesis holds that we can model these patterns as common subsequences with high probability of either describing DDI or not describing it.

Using a training set of sentences, we can determine, for each extracted maximal frequent sequence, how likely is for it to be describing an interaction between drugs. Those patterns and their probabilities will help to identify new drug-drug interactions.

We propose two solutions for helping health care professionals to be updated about published drug-drug interactions. The first solution is aimed at determining whether or not a sentence included a drug-drug interaction description. The second solution goes further in the search, and performs DDI extraction, *i.e.*, determining if two given drugs in a sentence interact or not.

Our first approximation is completely based on maximal frequent sequences extraction, while the second one is a machine learning approximation, that uses maximal frequent sequences, among others, as features. Following, we will describe in detail both approximations, and we will see the effectiveness of maximal frequent sequences for this particular task.

This Chapter is organized as follows: Section 3.1 contains the problem definition. Section 3.2 describes the performance measures that will be used to evaluate the systems. Section 3.3 explains the start of the art of DDI extraction. Section 3.4

describes the corpus used to build and evaluate our systems. In Section 3.5 we tackle the problem of DDI sentence identification and describe our system. In Section 3.6 we explain the DDI Extraction Challenge 2011, and following, in Section 3.7 we describe the system submitted to the competition. Section 3.8 presents some improvements made to the system submitted to the competition. Finally, in Section 3.10, we draw some conclusions.

3.1 Problem Definition

In this thesis we tackle two problems related to drug-drug interaction. The first task is drug-drug interaction sentence identification. This problem consists in determining whether or not a given sentence is describing one or more interactions between drugs. For this, we built classification models in the following form:

Given the sentence S, we build a classifier such as

$$c\colon S \rightarrow \{0, 1\}$$

Where $c(S)$ will determine whether or not the sentence S contains a DDI for any drugs.

The second problem we tackle is the one of drug-drug interaction extraction. This problem is defined as follows:

Given the sentence S, extracted from a pharmacological text,

$$S = w_1, w_2, \ldots, d_1, \ldots, w_n, \ldots d_2, w_{n+k} \ldots$$

containing a set of drugs

$$Ds = \{d_1, d_2, \ldots\}, \text{ with } |Ds| > 1$$

Ds, with $|Ds| > 1$, our task is to create a classification model

$$c\colon S, d_1, d_2 \rightarrow \{0, 1\}$$

that determines if sentence S is describing a DDI between drugs d_1 and d_2, where $(d_1, d_2) \in \binom{Ds}{2}$.

If S contains n different drugs, with $n > 1$, there will be $\frac{n(n-1)}{2}$ pairs $\in Ds_2$ that are potential interactions, and the classifier will have to determine, for each pair, if the sentence is describing a DDI between the drugs in the pair.

For example, in the sentence:

S = *Quinidine and procainamide doses should be reduced when either is administered with amiodarone.*

Ds = {quinidine, procainamide, amiodarone}

$\binom{Ds}{2}$ = {(quinidine, procainamide), (quinidine, amiodarone), (procainamide, amiodarone)}

In this case, the sentence describes an interaction only between quinidine and amiodarone, and procainamide and amiodarone. So an ideal classifier would result in:

$$c(s, quinidine, procainamide) = 0$$
$$c(s, quinidine, amiodarone) = 1$$
$$c(s, procainamide, amiodarone) = 1$$

Some classifiers output a value in [0,1] when classifying a sample that represent its confidence. In our case, 0 means not DDI and 1 means DDI. The closest that value is to 1, the most confident the classifier is about the sample being a DDI. Since this is a binary classification problem, we need to define a confidence threshold over which the classification will be positive (1) and under which the classification will be negative (0), and this way change the output of the classifier to a discrete, binary classification {0,1}.

3.2 Performance Measures

Drug-drug interaction sentence identification and DDI Extraction are supervised binary classification problem. In order to evaluate classifiers, we will use the common measures for this kind of problems. In this section, we describe such measures.

Table 3.1 shows a confusion matrix example, composed by the basic mea-

sures: true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). With these measures the performance measures Precision (3.1), Recall (3.2), F-Measure (3.3) and Accuracy (3.4) are calculated.

Table 3.1: Confusion matrix example.

		Actual classification	
		Positive	**Negative**
Predicted classification	**Positive**	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	False positive (FP)

Precision (P) is the fraction of samples classified as positives that are actually positives.

$$Precision = \frac{TP}{TP + FP} \tag{3.1}$$

Recall (R) is the fraction of positive samples correctly classified.

$$Recall = \frac{TP}{TP + FN} \tag{3.2}$$

F-Measure (F) is the harmonic mean of Precision and Recall.

$$F\text{-}Measure = 2 \cdot \frac{P \cdot R}{P + R} \tag{3.3}$$

Accuracy Right answers (positive and negative) over all answers given.

$$Accuracy = \frac{TP + FP}{total} \tag{3.4}$$

In order to compare different classifiers in a comprehensive and intuitive way we can plot the Precision-Recall curve (PR-curve), that consists in plotting Precision against Recall for every confidence threshold. The PR-curve is a quick way

to visualize every possible set up of the system. The area under the PR-Curve, AUC-PR, is a good measure to compare different classifiers (Davis and Goadrich, 2006) and will also be used in this thesis.

Analogously, we can plot F-Measure and confidence threshold, F-measure curve, to visualize the optimum threshold with respect to F-Measure.

3.3 Related Work

Drug-drug interaction identification is a relatively new field in the computational community. An increasing interest has been shown during the last few years, and new corpora has been presented as well as evaluation campaigns.

Duda et al. (2005) presents an approximation to the DDI sentence identification problem. The authors present an approximation to automatically extract articles that talk about DDIs from MEDLINE. Their system consists in a SVM classifier trained on a dataset of stemmed text words and MeSH terms.

Segura-Bedmar (2010) presents two techniques for DDI Extraction in biomedical texts. Both approaches were evaluated using the DrugDDI corpus (Segura-Bedmar, Martínez, and Pablo-Sánchez, 2010). The DrugDDI corpus is described in detail in Section 3.4.

The first approach is a hybrid approach, combining shallow parsing and pattern matching to extract relations between drugs from pharmacological texts. Complex sentences were split into clauses, and appositions and coordinate structures were detected using shallow syntactic and semantic information as given by MMTx. Pattern matching was applied to the split clauses in order to extract relations using patterns described by a experienced pharmacist after observing the training corpus. With this approximation, the author obtained 0.487 Precision and 0.257 Recall.

The second approach is based on kernel methods and combines two sequence kernel methods to integrate the information of the whole sentence where the relation occurs (global context kernel) and the context information about the interacting entities (local context kernel). This approximation used shallow syntactic information such as sentence splitting, tokenization and lemmatization, as well as part-of-speech (PoS) tagging. The system obtained 0.55 Precision and 0.84 Recall when evaluated with the DrugDDI corpus.

Sánchez-Cisneros, Segura-Bedmar, and Martínez (2006) presents the first on-

line tool for detecting drug-drug interactions from biomedical texts called DDIExtractor. The tool allows users to search by keywords in the Medline 2010 baseline database and then detect drugs and DDIs in any retrieved document.

Protein-Protein Interaction (PPI) extraction is an area of research very similar to DDI extraction that has received a bigger attention from the scientific comunity. The BioCreative III Workshop hosted two tasks of PPI document classification and interaction extraction (Arighi, Cohen, et al., 2010). Some of the features present in a wide range of participants were bag-of-words, bigrams, cooccurrences and character ngrams. This kind of features will have a key role in our system. In (Hakenberg et al., 2010) the authors use patterns as one of their main features to extract PPI. (Bui, Katrenko, and Sloot, 2011) uses a hybrid approach with clustering and machine learning classification using Support Vector Machines (SVM).

3.4 Corpus

The DrugDDI corpus (Segura-Bedmar, Martínez, and Pablo-Sánchez, 2010) is a corpus annotated with linguistic information, named entities and drug-drug interactions.

The corpus is composed of documents extracted from the DrugBank[1] database (Wishart et al., 2008). The DrugBank database is an online resource that offers information about over 4,900 pharmacological substances, including drug synonyms, brand names, chemical compositions and interactions. For the corpus, only the text available in the field *interactions*, containing unstructured information about known interactions, was retrieved. A total of 579 documents with an average of 10.3 sentences and 5.46 interactions per document were collected, each one describing interactions for a given drug. These documents were later annotated with drug-drug interactions by an experienced pharmacist. The corpus considers only DDIs at the sentence level, not taking into account DDIs that are described across several sentences.

The corpus was divided into two datasets. The first one consists of 446 documents and was used as the training dataset. The second set consists of 133 documents and was used as the test dataset. Drugs are tagged in the corpus, according to their type. Table 3.2 shows the different types of interacting drugs that

[1] `http://drugbank.ca`

were tagged in the corpus, as well as the presence of each drug in the training and test datasets. As we can observe, the distribution of drug types in both datasets is balanced. Table 3.3 shows some statistics on the DrugDDI corpus.

Table 3.2: Types of drugs present in the corpus and their relative frequency.

Type	**Test presence**	**Train presence**
Clinical Drug (clnd)	26 (0.6%)	105 (0.66%)
Pharmacological Substance (phsu)	3237 (76%)	11987 (76%)
Antibiotic (antb)	198 (4.67%)	670 (4.25%)
Biologically Active Substance (bacs)	228 (5.37%)	945 (5.99%)
Chemical Viewed Structurally (chvs)	27 (0.64%)	70 (0.44%)
Amino Acid, Peptide or Protein (aapp)	524 (12.36%)	1979 (12.56%)

Table 3.3: Statistics on the DrugDDI corpus. Table extracted from (Segura-Bedmar, Martínez, and Sánchez-Cisneros, 2011).

	Number	**Avg./document**
Documents	579	
Sentences	5,806	10.03
Phrases	66,021	114.02
Tokens	127,653	220.47
Sentences with at least one DDI	2,044	3.53
Sentences with no DDI	3,762	6.50
DDIs	3,160	5.46 (0.54 per sentence)

Two versions of the DrugDDI corpus are available, with different formats: MMTx format (Aronson, 2001) and Unified format (Pyysalo et al., 2008). The Unified format only contains labels for drugs and interactions. This format is the most used format in protein-protein interaction corpora. The MMTx[2] format contained, in addition to drugs and interactions, sentence splitting, tokenization, POS-tagging, shallow syntactic parsing and linking of phrases with UMLS Metathesaurus concepts. In this thesis, we only used the Unified format, therefore, we will not go deeper into the MMTx format of the corpus[3].

Figure 3.1 shows a fragment of one of the documents in Unified format. We can observe the different drugs tagged, as well as for each pair of drugs a boolean indicative of whether they are being described as DDI or not. Table 3.4 shows the

[2]Analyzed by the UMLS MetaMap Transfer (MMTx) tool.

[3]More information about the MMTx format can be found at (Segura-Bedmar, Martínez, and Pablo-Sánchez, 2010).

corpus statistics. Note that these statistics cover only documents and sentences that contain, at least, one drug pair.

```
- <sentence id="DrugDDI.d529.s3" origId="s3" text="Elevated serum levels of cyclosporine have been reported
  with concomitant use of cyclosporine with norfloxacin.">
    <entity id="DrugDDI.d529.s3.e0" origId="s3.p29" charOffset="25-37" type="drug" text="cyclosporine"/>
    <entity id="DrugDDI.d529.s3.e1" origId="s3.p34" charOffset="81-93" type="drug" text="cyclosporine"/>
    <entity id="DrugDDI.d529.s3.e2" origId="s3.p35" charOffset="99-110" type="drug" text="norfloxacin"/>
    <pair id="DrugDDI.d529.s3.p0" e1="DrugDDI.d529.s3.e0" e2="DrugDDI.d529.s3.e1" interaction="False"/>
    <pair id="DrugDDI.d529.s3.p1" e1="DrugDDI.d529.s3.e0" e2="DrugDDI.d529.s3.e2" interaction="False"/>
    <pair id="DrugDDI.d529.s3.p2" e1="DrugDDI.d529.s3.e1" e2="DrugDDI.d529.s3.e2" interaction="True"/>
  </sentence>
```

Figure 3.1: Fragment of the document *Norfloxacin_ddi.xml*.

Table 3.4: DrugDDI corpus statistics on sentences containing at least one drug pair.

	Training	Test	Total
Documents	399	134	533
Sentences	2812	965	3777
Pairs of drugs	23827	7026	30853
Interactions	2397	755	3152

Table 3.5: Distribution of positive and negative examples in training and testing datasets. Extracted from (Segura-Bedmar, Martínez, and Sánchez-Cisneros, 2011) and simplified.

Set	Documents	Examples	Positives	Negatives
Training	437 (75.5%)	25,209	2,421 (9.6%)	22,788 (90.4%)
Test	142 (24.5%)	5,548	739 (13.3%)	4,809 (86.7%)
Total	579	30,757	3,160 (10.27%)	27,597 (89.73%)

3.5 DDI Sentence Identification

As we have seen in Section 3.1, the problem of DDI Sentence Identification consists in determining whether a sentence is describing a drug-drug interaction or not. In this section we will describe the system we built for this matter.

In this first approach[4], maximal frequent sequences were used to identify patterns from a set of sentences extracted from biomedical texts. For each MFS

[4]This section is based on our publication (García-Blasco, Danger, and Rosso, 2010).

extracted, we calculated which percentage of the sentences in which it appeared were positive samples, and that gave us a confidence threshold. These MFS were afterwards used to identify, out of a test set of documents, sentences that contained drug-drug interactions.

In Section 3.5.1, we describe some details about the corpus preprocessing performed for this experimentation. Section 3.5.2 describes the experiments performed for this task. In Section 3.5.3 we can find a review, as well as a discussion, of the results obtained.

3.5.1 Corpus Preprocessing

The corpus used was the DrugDDI corpus (Segura-Bedmar, Martínez, and Pablo-Sánchez, 2010), explained in detail in Section 3.4. In this case, the corpus was divided into two datasets. The first one consisted of 446 documents and was used as training dataset. The second one consisted of 133 documents and was used as test dataset. As we saw, drugs were tagged in the corpus, according to their type, *e.g.* clinical drug, pharmacological substance, antibiotics, etc. Prior to the extraction of MFS, we performed some preprocessing to the corpus, in order to obtain different versions of the corpus that would result in different kinds of MFS extracted.

Taking advantage of the annotations in the corpus, two different preprocessing methods were applied to the original training dataset. The first one consisted in replacing all drug names that appeared in the text with their type, *e.g.* each clinical drug was replaced with the token *clnd*, pharmacological substance with the token *phsu* , antibiotic with *antb*, etc. We refer to this dataset as *6drugs*. The second preprocessing method consisted in replacing all drug names with the token *#drug#*. We refer to this dataset as *#drug#*. When we talk about the dataset *norm*, we refer to the original dataset, without any preprocessing. To make the different datasets generated more clear, Table 3.6 shows a sentence and its modification according to each dataset preprocessing.

3.5.2 Experiments

The objective of this experiment is to identify drug-drug interactions in biomedical texts using maximal frequent sequences.

Table 3.6: A sample sentence for each dataset generated after preprocessing.

Dataset	Example
norm	barbiturates may decrease the effectiveness of oral contraceptives, certain antibiotics, quinidine, theophylline, corticosteroids, anticoagulants, and beta blockers.
6drugs	*phsu* may decrease the effectiveness of oral contraceptives, certain *antb*, *phsu*, *phsu*, *phsu*, *phsu*, and *phsu*.
#drug#	*#drug#* may decrease the effectiveness of oral contraceptives, certain *#drug#*, *#drug#*, *#drug#*, *#drug#*, *#drug#*, and *#drug#*.

Different sets of MFS were extracted from the training set using different parameters. The algorithm was executed with the three different versions of the corpus and the following values for the parameters:

Table 3.7: Parameters of the experiments.

preprocessing	*norm*, *6drugs*, *#drug#*
$\boldsymbol{freq_{min}}$	10, 15, 20
gap	0, 1, 2
$\boldsymbol{min_{length}}$	4

The MFS detected were rated using a new function that we define as *likeliness* 3.5 and represents the probability of the MFS describing a DDI.

$$likeliness(MFS_i) = \frac{\text{times } MFS_i \text{ identifies DDI}}{\text{times } MFS_i \text{ appears in the corpus}} \tag{3.5}$$

3.5.3 Results

The MFS found had an average length between 4.09 and 4.51 depending on the parameters and the preprocessing of the corpus.

As explained in Section 3.5.2, each MFS has associated a *likeliness* value, that is an indicator of how likely is the MFS to describe a drug-drug interaction. Figure 3.2 shows the amount of MFS found for the different corpus, with $freq_{min} = 20$. The bars are also divided according to the likeliness of the MFS.

The algorithm detected more patterns in the *#drug#* dataset. With this dataset and for $gap = 2$ and $min_{lenght} = 10$ the greatest amount of patterns were found, since they are the less restrictive parameters.

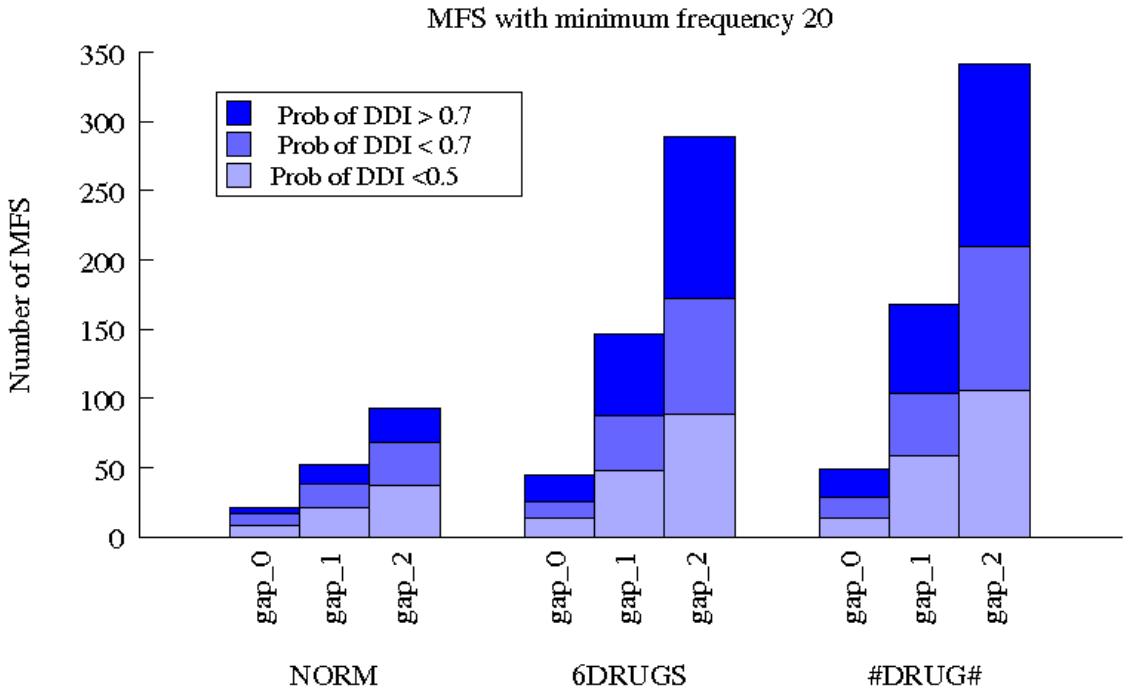

Figure 3.2: Number of MFS and their likeliness.

For example, using the *#drug#* corpus, with req_{min}=*10* and $gap = 1$, the following MFS was found:

'#drug#', 'may', 'the', 'effects', 'of', '#drug#'

This MFS was extracted from sentences like:

- Acetazolamide *may* increase *the effects of* other folic acid antagonists.
- Alcohol *may* potentiate *the* side *effects of* bromocriptine mesylate.
- Dopamine D2 receptor antagonists (e.g., phenothiazines, butyrophenones, risperidone) and isoniazid *may* reduce *the* therapeutic *effects of* levodopa.
- Concomitant administration of other sympathomimetic agents *may* potentiate *the* undesirable *effects of* Foradil.

We define a threshold for the *likeliness* value of each maximal frequent sequence extracted. Above this threshold a maximal frequent sequence will be considered as a descriptor of a drug-drug interaction. With this, we will classify the sentences included in the test dataset, and evaluate the performance of the method. This threshold will play an important role in the performance of the method.

In Figures 3.3, 3.4 and 3.5 the F_1-measure over the *likeliness threshold* is shown for the different preprocessing and $gap = 0$, 1 and 2 respectively. With a greater *gap*, Recall grows but it obtains less Precision.

For datasets *#drug#* and *6drugs*, the best threshold is in the range $[0.6, 0.7]$. For the normal text, without preprocessing, the best threshold is in the range $[0.1, 0.5]$.

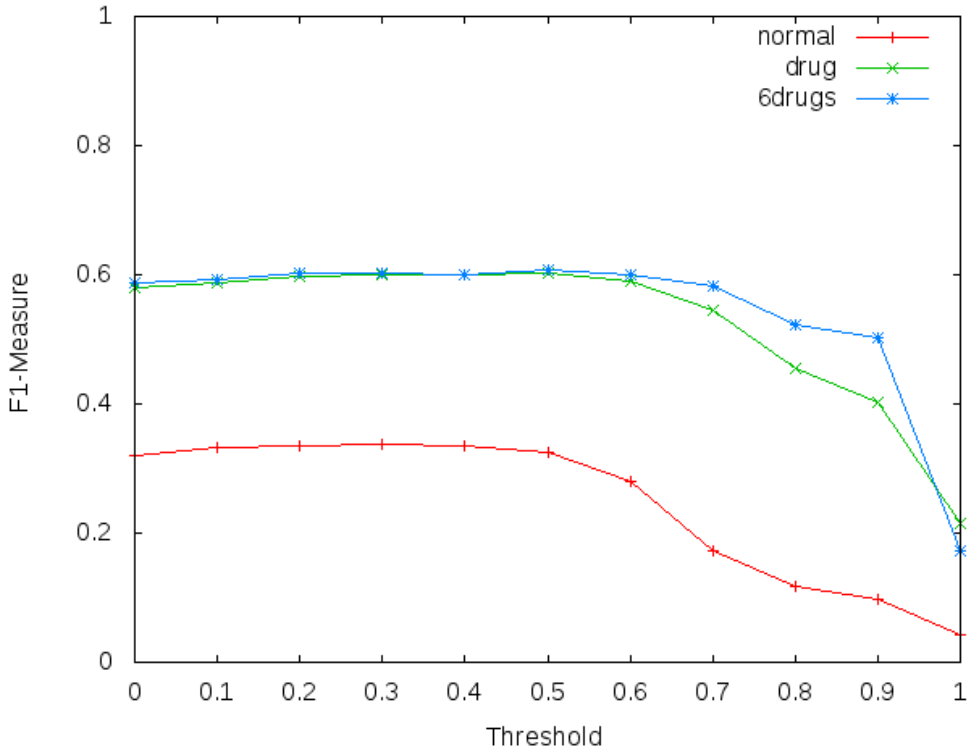

Figure 3.3: F_1 for $freq_{min}$=10, with $gap = 0$.

Observing the maximal frequent sequences extracted, we can find different types of sequences. Those that have a high value of *likeliness* can be mostly divided in two big groups, those which contain verbs that denote effects, i.e. *increase*, *decrease*, *enhance*, etc., and those which contain 2 or more drugs. Table 3.8 shows some examples of this two types of maximal frequent sequences extracted from the documents, their frequency and *likeliness*.

Table 3.9 gives an overview of the results obtained in the experiments, with *gap=2* and *$freq_{min}$=10*.

The test set consists of 1151 sentences, with 461 of them describing DDI. The baseline for this task is *allDDIs*, in which all sentences are labeled as containing

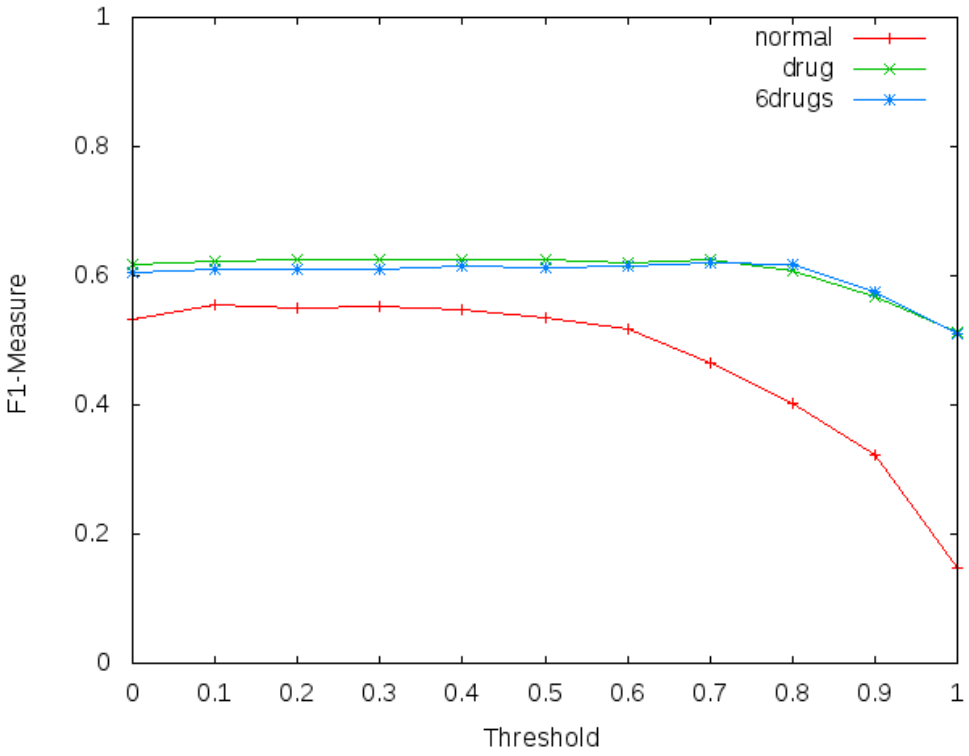

Figure 3.4: F_1 for $freq_{min}$=10, with $gap = 1$.

Table 3.8: Examples of extracted MFS.

MFS Sample	***freq***	***likeliness***
With verbs denoting effects:		
('#drug#', 'may', 'increase', 'of')	30	0.93
('may', 'decrease', 'the', 'of')	21	0.90
('#drug#', 'may', 'enhance', 'the', 'of')	10	1.0
('with', '#drug#', 'increase', 'the', 'of')	10	1.0
('#drug#', 'is', 'administered', 'with')	21	0.81
With 2 or more drugs:		
('#drug#', 'may', 'the', 'effects', '#drug#')	13	1.0
('#drug#', 'should', 'not', 'be', 'with', '#drug#')	11	1.0
('#drug#', 'reduce', 'the', 'of', '#drug#')	15	0.93

DDI. Table 3.9 contains a relation of the results obtained in this research.

As Table 3.9 shows, some of the parameters give a very high Recall value (0.95). Drug-drug interactions are described by the researchers using a reduced vocabulary and similar sentence structures, i.e. *"Amiodarone should be used with*

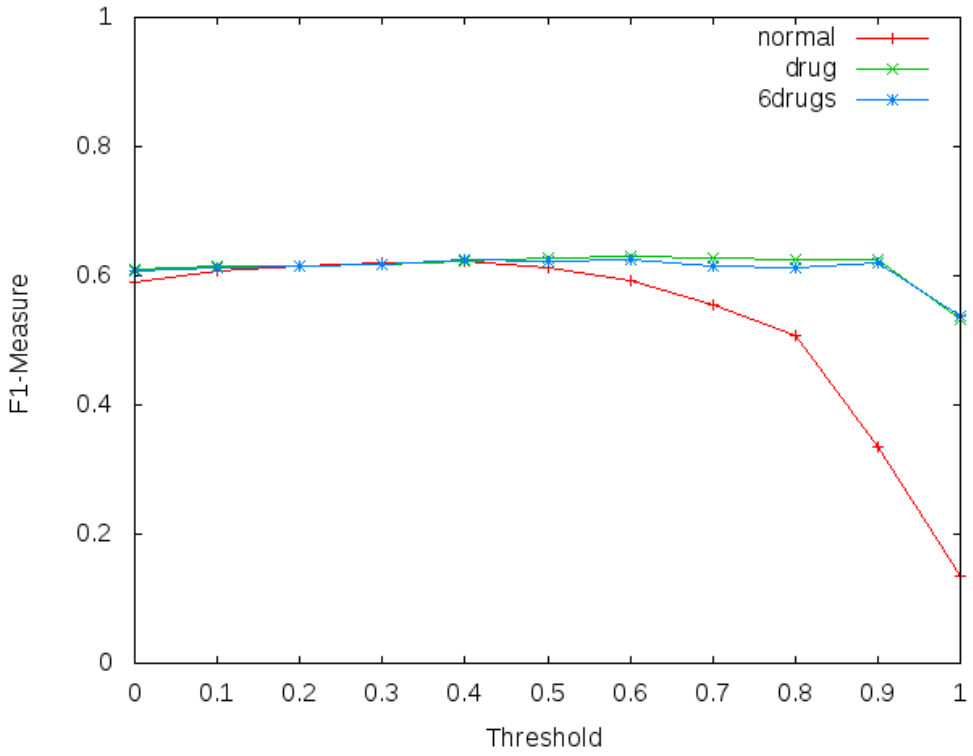

Figure 3.5: F_1 for $freq_{min}$=10, with $gap = 2$.

Table 3.9: Comparison of results.

	Precision	Recall	F_1
baseline	*0.40*	*1*	*0.28*
6drugs	0.48	0.93	**0.63**
norm	**0.68**	0.41	0.51
#drug#	0.46	**0.95**	0.62

caution in patients receiving propranolol". This allows us to find a set of MFS that retrieve the great majority of the DDIs described. However, the same sentence structures are sometimes used in other contexts, i.e. *"It should be used with caution in patients with diabetes"*. This sentence does not define a DDI, but it does contain a MFS with high likeliness value and it will be labeled as DDI descriptor, decreasing Precision.

3.6 DDI Extraction

3.6.1 First Challenge Task: Drug Drug Interaction Extraction

DDIExtraction2011 (Segura-Bedmar, Martínez, and Sánchez-Cisneros, 2011) proposes a first challenge task in Drug-Drug Interaction Extraction to compare different techniques for DDI extraction and to set a benchmark that will enable future systems to be tested. Each team participating in the challenge was allowed to submit up to 5 runs with different settings of their system. A total of 10 teams from different parts of the world participated in the challenge, submitting a total of 40 runs.

The goal of the competition was, for every pair of drugs in a sentence, decide whether an interaction is being described or not.

The corpus used for the competition was the DrugDDI corpus, presented in (Segura-Bedmar, Martínez, and Pablo-Sánchez, 2010). The corpus is described in Section 3.4. In Section 3.6.1.2 a review of the different systems submitted to the competition is carried out.

3.6.1.1 Evaluation

The runs submitted to the competition were evaluated according to their F-Measure. Table 3.10 shows the results obtained by the best run submitted by each team in the competition[5]. For each team we can see Precision, Recall and F-Measure obtained, as well as the Accuracy.

In the following Section, we shortly describe the most relevant systems that participated in the competition. In general, approaches based on kernels methods achieved better results than the classical feature-based methods.

3.6.1.2 Systems Submitted to the Competition

Ten teams participated in the competition, submitting a total of 40 runs. Each team presented a different approximation. Following, we do a short description of the most relevant systems submitted, as well as the results obtained by each system in the competition. The systems are sorted by the position in the final

[5]Our team was BNB_NLEL, named after Bitsnbrains S.L. `http://bitsnbrains.net` and Natural Language Engineering Lab `http://www.dsic.upv.es/grupos/nle`.

Table 3.10: Precision, Recall, F-Measure and Accuracy obtained by the best run of each team in the DDI Extraction Challenge 2011 and their system description. Table extracted from (Segura-Bedmar, Martínez, and Sánchez-Cisneros, 2011) and simplified.

Team	Description	P	R	F	Acc
WBI	Combination of several kernels and a case-based reasoning (CBR) system using a voting approach.	0.6054	0.7192	0.6574	0.9194
LIMSI-FBK	A feature-based method using SVM and a composite kernel-based method.	0.5859	0.7046	0.6398	0.9147
FBK-HTL	composite kernels using the MEDT, PST and SL kernels.	0.5839	0.7007	0.6370	0.9142
UTurku	Machine learning classifiers such as SVM and RLS; DrugBank and MetaMap.	0.5804	0.6887	0.6299	0.9130
LIMSI-CNRS	A feature-based method using libSVM and SVMPerf	0.5518	0.6490	0.5965	0.9056
BNB-NLEL	Feature-based method using Random Forests	**0.6122**	**0.5563**	**0.5829**	**0.9145**
Laberinto-UHU	A feature-based method using classical classifiers such as SVM, Naive Bayes, Decision Trees, Adaboost	0.5000	0.4437	0.4702	0.8925
DrIF	Two machine learning-based (CFFs and SVMs) and one hybrid approach which combines CFFs and a rule-based technique.	0.4037	0.4887	0.4422	0.8675
ENCU	A feature-based method using SVM.	0.2957	0.4649	0.3615	0.8235
IUPUITMGroup	All paths graph (APG) kernel	0.1170	0.2556	0.1605	0.7126

ranking. We skip the description of our system, which was ranked as 6th. It will be described in detail in Section 3.7.

I Relation Extraction for Drug-Drug Interactions using Ensemble Learning

In this approximation, Thomas et al. (2011) built a majority voting system that uses several classifiers. They had two types of classifiers, *i.e.*, kernel and case-based reasoning. The best run submitted used a voting system with two kernels (all-paths graph and shallow linguistic) and a case-based reasoning classifier. The

system archived Precision 0.6054, Recall 0.7192, and F-Measure 0.6574, being the best performing system.

II Two Different Machine Learning Techniques for Drug-Drug Interaction Extraction

The approximation presented in (Chowdhury et al., 2011) consists of the combination of two different machine-learning approaches. The first one is a feature-based method using a SVM classifier with a set of lexical, morphosyntactic and semantic features (*e.g.* trigger words, negation). The second one is a kernel composed of a mildly extended dependency tree (MEDT) kernel, a phrase structure tree (PST) kernel and a shallow linguistic kernel. This system reached position two in the classification, with a 0.5859 Precision, 0.7046 Recall and 0.6398 F-Measure.

III Drug-drug Interaction Extraction Using Composite Kernels

After trying different types of kernels, Chowdhury and Lavelli (2011) obtained their best performing result with a system that combined three kernels: a mildly extended dependency tree (MEDT) kernel, a phrase structure tree kernel and a global context kernel. This system obtained 0.5839 Precision, 0.7007 Recall and 0.6370 F-Measure, being the third best performing team.

IV Drug-Drug Interaction Extraction from Biomedical Texts with SVM and RLS Classifiers

The system presented by Bjorne et al. (2011) is based on the publicly available Turku Event Extraction System (Neves, Carazo, and Pascual-Montano, 2009) which abstracts event and relation extraction by using an extendable graph format. The system extracts information in two main steps: detection of trigger words (nodes) denoting entities in the text, and detection of their relationships (edges). With this information, the authors built a support vector machine classifier and a regularized least-squares (RLS) classifier. The best results were obtained with the RLS classifier, reaching the fourth position in the classification with Precision 0.5804, Recall 0.6887 and F-Measure 0.6299.

V Feature Selection for Drug-Drug Interaction Detection Using Machine-Learning Based Approaches

In a first step, Minard et al. (2011) built a knowledge database including the pair of drugs in the corpus that always interact, to later combine this information with the decisions of their classifier. The systems use machine learning methods based on SVM by using LIBSVM and SVMPerf tools with different sets of features that included lexical, morphosyntactic and semantic features, as well as corpus based features (*e.g.* most frequent drug in the document). This system reached fifth position in the evaluation ranking with Precision 0.5518, Recall 0.6490 and F-Measure 0.5965.

VI A Machine Learning Approach to Extract Drug-Drug Interactions in an Unbalanced Dataset

Mata et al. (2011) present a machine learning approach using as features a set of manually selected words which included mostly all of the verbs, some nouns, and prepositions, adverbs and conjunctions that might express negation of frequency. Afterwards, this features were ranked with a chi-squared feature selection method, selecting 496 features. The authors also use SMOTE algorithm to tackle the problem of the unbalanced corpus. The final classifier used by the authors was Random Forest with 50 iterations, which gave them the seventh position in the classification ranking with Precision 0.5000, Recall 0.4437 and F-Measure 0.4702.

3.7 Our DDI Extraction 2011 Submission

In this section we will describe the system presented for the *First Challenge Task: Drug Drug Interaction Extraction*. We used the DrugDDI corpus, described in Section 3.6[6]. We built a system to perform DDI Extraction, *i.e.*, to identify interactions between two specific drugs, instead of just determine whether a sentence describes an interaction between drugs or not as we did in Section 3.5.

As explained in Section 3.1, we need a classifier

$$c\colon S, d_1, d_2 \rightarrow \{0, 1\}$$

[6]This section is based on our publication (García-Blasco et al., 2011)

that determines if sentence S is describing a DDI between drugs d_1 and d_2.

In order to train the classifier[7], we had to define a feature set to represent each sample. Each sample is one possible interaction, *i.e.*, each unique combination of two drugs appearing in a sentence of the corpus. Given the small size of the corpus and the difficulty to properly estimating a model, it was necessary to reduce the dimensionality of the feature space.

The first step was to preprocess the corpus. For doing so, each sentence was tokenized[8] with standard English tokenization rules (e.g. split by spaces, removal of apostrophes, conversion to lower case, removal of punctuation marks) with the following particularities:

- Each token or group of tokens that represent a drug were replaced by the token *#drug#*.
- Numbers were replaced by *_num_*.
- Stop words were not removed.
- Stemming was applied[9].
- Percentage symbols were preserved as independent tokens.

Following, we will describe the different features used in the system.

3.7.1 Bag of Words

The first feature set is a classic bag of words. From the set of all words appearing in the preprocessed corpus, we discarded those with a frequency lower than 3 and stop words. With the resulting set of words, we generated a dataset where each sample was a possible interaction in the corpus and each feature was the presence or not of each word between the two drugs of the potential interaction. Using this dataset, every word was ranked using Information Gain Ratio (IGR) with

[7]We used RapidMiner for every classification and clustering model. Available at `http://rapid-i.com/`.

[8]The tokenization was performed with Apache Lucene. Available at `http://lucene.apache.org`.

[9]The stemming algorithm used was Snowball for English. Available at `http://snowball.tartarus.org`.

respect to the target class [10]. Then, every word with IGR lower than 0.0001 was discarded[11]. The presence of each of the remaining words was a feature in the final dataset. Finally, 1.010 words were kept. Samples of words with a high gain ratio are: *exceed*, *add*, *solubl*, *amphetamin*, *below*, *lowest*, *second*, *defici*, *occurr*, *stimul* and *acceler*.

3.7.2 Word Categories

In biomedical literature complex sentences are used very frequently. MFS and bag of words are not able to capture relations that are far apart inside a sentence. To somehow reflect the structure of the sentence, we defined some word categories. This way, we can have some information about dependent and independent clauses, coordinate and subordinate structures, etc. Some of this categories were also included in (Segura-Bedmar, 2010). We added two categories that include absolute terms and quantifiers, as well as a category for negations. Table 3.11 enumerates the words included in each category.

Table 3.11: Word categories.

Category	**Words included**
Subordinate	*after, although, as, because, before, if, since, though, unless, until, whatever, when, whenever, whether, while*.
Independent markers	*however, moreover, furthermore, consequently, nevertheless, therefore*.
Appositions	*like, including, e.g., i.e*.
Coordinators	*for, and, nor, but, or, yet, so*.
Absolute	*never, always*.
Quantifiers	*higher, lower*.
Negations	*no, not*.

For each word category we defined two features. One indicating how many times any word in the category appeared in the sentence, and the other indicating how many times they appeared between the two drugs of the potential interaction. So, if a word appeared in a sentence between two drugs, both features would be set to 1. If two words of the same category appeared between two drugs (or the same word appeared twice), both features would be set to 2.

[10]Information Gain Ratio was calculated using Weka. Available at `http://www.cs.waikato.ac.nz/ml/weka/`.

[11]The threshold for IGR was manually adjusted fixed by analyzing intuitively the results.

3.7.3 Maximal Frequent Sequences

Similar to bag of words, we used sequences of words as features. For this, we used maximal frequent sequences.

We extracted all the MFS from the training corpus, with a minimum frequency of 10 and minimum length of 2. Given the size of the corpus, sometimes very long MFS have no capability to generalize knowledge because they sometimes represent full sentences, instead of patterns that should be frequent in a kind of sentence. To avoid this, we restricted the MFS to a maximum length of 7 words. With this, we obtained 1.010 patterns.

Since we setted the minimum frequency of the MFS to 10, many of the patterns extracted do not have enough samples in the training corpus to estimate correctly the model. For this reason, we decided to group different patterns with similar characteristics into clusters of MFS.

Clusters were calculated with the Kernel K-Means algorithm (Zhang and Rudnicky, 2002), using radial kernel, with respect to the following relative frequencies of the MFS and the words that it contains:

Sentence Frequency Percentage of sentences containing the MFS.

Sentence Frequency with Interaction Percentage of sentences, with at least one interaction, containing the MFS.

Pair Frequency Percentage of times the MFS appears between two drugs.

Pair Frequency with Interaction Percentage of times the MFS appears between two drugs that are interacting.

Average Word Frequency Average frequency in the corpus of the words contained in the MFS.

Average Word with Interaction Average frequency of the words contained in the MFS in sentences that contain interactions.

With this, we obtained 274 clusters. Each of these clusters is a feature of the final dataset which is set to 1 if, at least, one of the MFS of the cluster matches with the potential interaction. For this matter, we define the following Matching Algorithm 2. Note that this algorithm is specific for the problem of DDI extraction.

Algorithm 2: MFS Matching Algorithm.

Input: mfs, $sentence$, $drug1index$, $drug2index$
Output: $match$

$startThreshold \leftarrow 0$
$endThreshold \leftarrow 0$
if *"#drug#"* $\in mfs$ **then**
    $startThreshold \leftarrow$ First index of "$\#drug\#$" in mfs
    $endThreshold \leftarrow length(mfs) -$ last index of "$\#drug\#$" in mfs

$startIndex \leftarrow drug1index - startThreshold * (gap + 1)$
if $startIndex < 0$ **then**
    $startIndex \leftarrow 0$
$endIndex \leftarrow drug2index + endThreshold * (gap + 1)$
if $endIndex > length(sentence)$ **then**
    $endIndex \leftarrow length(sentence)$

$textToCompare \leftarrow$ Substring of $sentence$ from index $startIndex$ to $endIndex$
if mfs *is subsequence of* $textToCompare$ **then**
    $match \leftarrow 1$
else
    $match \leftarrow 0$
return $match$

If the MFS contains the token *#drug#* then the *startThreshold* is set to the first index of *#drug#* in the MFS, if not it it set to 0. The same for the *endThreshold*, if the MFS contains the token *#drug#*, then it is set to difference between the length of the MFS and the last index of *#drug#* in the MFS, otherways it is set to 0.

For example, given the MFS:

$MFS_1 =$ <'administration', '#drug#', 'may', 'the', 'effects', '#drug#'>

startThreshold would be set to 1, and *endThreshold* would be set to 0. In other words, *startThreshold* and *endThreshold* represent how many words appear in the pattern before the first and after the last appearance of the token *#drug#*, respectively.

With this thresholds and the input sentence, we calculate the piece of the sentence that we have to compare with the MFS.

For example, given the sentence:

s_1 = The administration of drug1 may increase the effects of drug2.

and the thresholds calculated in the previous sample, the part of the sentence that we would have to compare with the MFS would be:

$textToCompare$ = "administration of drug1 may increase the effects of drug2".

After preprocessing, the sentence would not have the name of the actual drugs in it, but just the token *#drug#*. Therefore, the text to compare would be:

$textToCompare$ = "administration of #drug# may increase the effects of #drug#".

Since the gap is set to 1, the sentence does contain the MFS, and the matching algorithm would return 1.

In case the MFS does not contain the token #drug#, then the MFS is matched with the text in between the drugs that compose the potential interaction description.

3.7.4 Token and Char Level Features

At the token and char level, several features were defined. We must recall that, during preprocessing, every token or group of tokens labeled as drugs where replaced by the token *#drug#*. Table 3.12 describes this subset of features. Each one of these features appears twice in the final dataset, once computed on the whole sentence and once computed only in the text between the two drugs of the potential interaction.

Table 3.12: Token and char level features.

Feature	**Description**
Tokens	Number of tokens.
Token *#drug#*	Number of times the *#drug#* token appears.
Chars	Number of chars.
Commas	Number of commas.
Semicolons	Number of semicolons.
Colons	Number of semicolons.
Percentages	Number of times the character % appears.

3.7.5 Drug Level Features

With the features defined so far, we have not taken into account the two drugs of the potential interaction. We believe this is important in order to have more information when deciding whether they interact or not.

For each document, we calculated the *main drug* as the drug after which the document was named, this is, the name of the article of the DrugBank database where the text was extracted from. In the case of scientific articles, the main drug would be calculated as the drug or drug names appearing in the title of the article, if any. Also for each document, we calculated the *most frequent drug* as the token labeled as drug that appeared more times in the document.

We noticed that, sometimes, drugs are referred to using their trade names. To ensure good treatment of drugs in the drug level features, we replaced each trade name with the original drug name[12]. Table 3.13 describes the drug level features.

Table 3.13: Drug level features for candidate interactions (CI).

Feature	**Description**
Main drug	True if one of the two drugs in the CI is the document name.
Most frequent drug	True if one of the two drugs in the CI is the most frequent drug in the document.
Cross reference	True if, at least, one of the two drugs in the CI is *drug*, *medication* or *medicine*.
Alcohol	True if, at least, one of the two drugs in the CI is *alcohol* or *ethanol*.
Is same drug	True if both drugs in the CI are the same.

3.7.6 Classification Model

During preliminary research, we explored the performance of a wide range of classification models, notably Support Vector Machines, Decision Trees and multiple ensemble classifiers such as Bagging, MetaCost and Random Forests (Breiman, 2001).

Our best choice was Random Forest. Random Forest has two parameters that we needed to set. The number of iterations and the number of attributes that the algorithm considers in each iteration.

[12]Trade names were extracted from the KEGG DRUG database, from the Kyoto Encyclopedia of Genes and Genomes. Available at `http://www.genome.jp/kegg/drug/`

For each each label, our model outputs a confidence value that represents the probability of the pair of drugs to be interacting. In order to decide the label, we define a confidence threshold above which the decision will be positive and below which it will be negative.

3.7.7 Experiments

We performed experiments to evaluate the performance of our system for the test dataset, with and without MFS. The number of iterations for Random Forest was set to 100, and the number of attributes for each iteration was set to 100. After some experiments, MFS were extracted using $min_{length} = 2$, $max_{length} = 7$, $min_{freq} = 5$ and $gap = 0$.

3.7.8 Results and Discussion

Figure 3.7 shows PR and F curves for both settings. The PR curves are convex, which makes the decision of an optimum threshold much easier and less risky. Table 3.14 shows Precision, Recall, F-Measure, AUC-PR, Precision at Recall 0.8 and Recall at Precision 0.8 for test with MFS.

As observed in Table 3.14, in the case of 10-fold cross-validation results are above the rest. We must point out that the MFS extraction phase was run with the whole training set without cross-validation. This means that some information about the test samples of each cross-validation iteration is leaked to the training phase. We still include these results since we think that with a sufficient big corpus, the tendency should be the same.

Table 3.14: Performance measures for test with, and without MFS.

	P	**R**	**F**	**AUC-PR**	**P@R 0.8**	**R@P 0.8**
Test	0.6122	0.5563	0.5829	0.6341	0.4309	0.3205
Test w/o MFS	0.6069	0.5563	0.5805	0.6142	0.4113	0.2808
Cross-validation	0.6235	0.6914	0.6453	0.6696	0.5167	0.3199

The submitted run obtained a Precision of 0.61, a Recall of 0.55 and a F-measure of 0.58.

The competition classification was made based on the F-Measure scores. Even though each participating team was allowed to submit up to five runs, we only submitted one. Table 3.15 shows the confusion matrix of our submit.

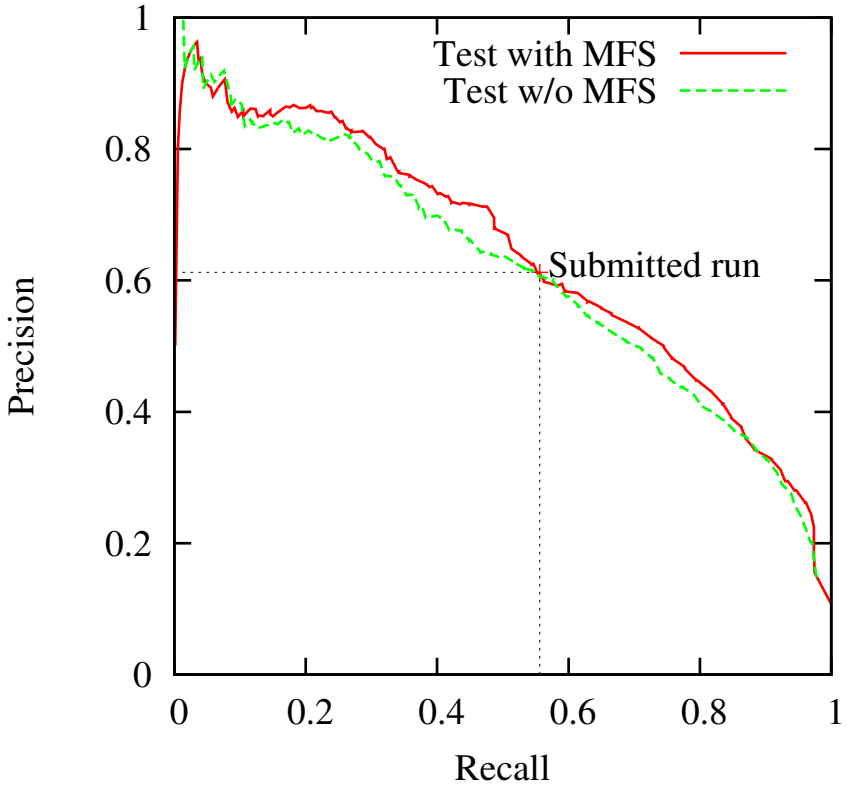

Figure 3.6: Precision-Recall curves for the results given by our system for the test, with and without MFS.

Table 3.15: Confusion matrix.

		Actual classification	
		Positive	**Negative**
Predicted classification	**Positive**	420	266
	Negative	335	6005

Our team[13] obtained 6th position in the final classification ranking, according to F-Measure. The best performing run had an F-measure of 0.65, 0.07 points apart from our 0.58. The team following us, obtained an F-measure of 0.47, 0.11 points away from us. Therefore, we are closer to the first classified team than to the one following us. Our system is in the 5th position according to Accuracy, 0.006 points away from the first one. Note that the accuracy values are considerably high due to the fact that it also takes into account the non-interaction, which

[13]Our team was BNB_NLEL, named after Bitsnbrains S.L. `http://bitsnbrains.net` and Natural Language Engineering Lab `http://www.dsic.upv.es/grupos/nle`.

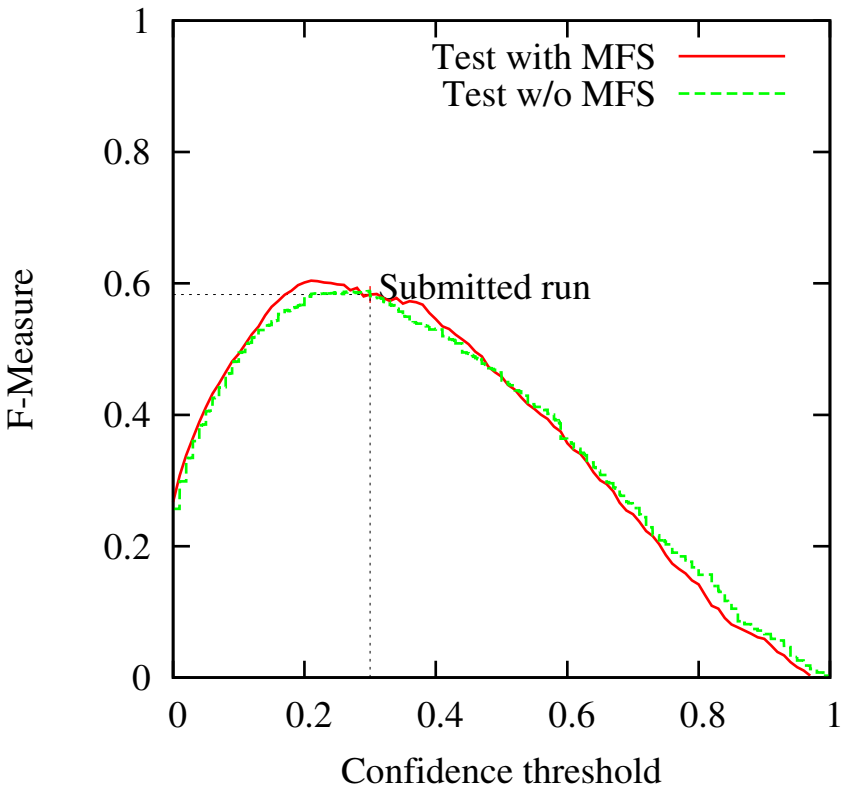

Figure 3.7: F-measure curves for the results given by our system for the test, with and without MFS.

are the vast majority and therefore are much easier to guess right.

MFS improve moderately the performance of the system, increasing about 0.02 in AUC-PR. Even though we expected more influence of MFS, adding MFS does detect new interactions that were not detected before.

Following, we will discuss some cases when the system with MFS detects DDIs that are not detected without them. In each sample sentence, the tokens tagged as drugs in the corpus are underlined.

For example, the sentence:

> *Drugs that induce hepatic enzymes such as phenobarbital, phenytoin and rifampin may increase the clearance of corticosteroids and may require increases in corticosteroid dose to achieve the desired response.*

The interaction between *rifampin* and *corticostedoids* is only detected when

adding MFS.

Another example is shown in Table 3.16 where we can see that all the interactions in the sentence where detected with MFS and none without them.

> *Drugs such as troleandomycin and ketoconazole may inhibit the metabolism of corticosteroids and thus decrease their clearance.*

		Solution	With MFS	Without MFS
Drugs	troleandomycin	0	0	0
Drugs	ketoconazole	0	0	0
Drugs	corticosteroids	1	1	0
troleandomycin	ketoconazole	0	0	0
troleandomycin	corticosteroids	1	1	0
ketoconazole	corticosteroids	1	1	0

Table 3.16: Comparison of solution and prediction with and without MFS for a sample sentence.

But, sometimes MFS do the inverse effect, tagging as DDI pairs of drugs that are not interacting. For example, in the sentence:

> *Other: Neither fosinopril sodium nor its metabolites have been found to interact with food.*

In this case, the classifier that contains MFS does classify the interaction as true, and the classifier without MFS does not. In this case, therefore, it works better without MFS.

There are inaccuracies in the corpus that affect the training and prediction of the DDI. For example, the in sentence:

> *In patients who have received muscle relaxants, doxapram may temporarily mask the residual effects of muscle relaxant drugs.*

The fact that the last drug tagged comprehends only the text *"drugs"* and not *"muscle relaxant drugs"* as it should be, makes the sentence to not contain a MFS that would detect the interaction because of the gap restriction. Due to this tagging mistake, the token *#drug#* is two positions displaced. This also means that our current approximation for the MFS in this task is very sensitive to errors in the named entity recognition step.

Another sample is the following sentence:

Administration of doxapram to patients who are receiving sympathomimetic or monoamine oxidase inhibiting drugs may result in an additive pressor effect.

Entity e_3 =*"additive pressor effect"* is incorrectly tagged as a drug. Therefore, the only interacting drugs are e_0 =*doxapram* with e_1 =*sympathomimetic* and e_0 =*doxapram* with e_2 =*monoamine oxidase*. In this case, without MFS the answer is right, it detects only those two interaction. However, with MFS the system not only detects those two interactions, but it also detects an interaction between e_0 and e_3, which is probably due to the fact that e_3 is incorrectly tagged as a drug.

In the sentence:

Acetazolamide may prevent the urinary antiseptic effect of methenamine.

The token *prevent* is tagged as drug. Since *prevent* will be replaced by the token *#drug#*, we will loose information about the verb between the two actual drugs, missing any pattern that contains the word *prevent*. This would have been mitigated if we also considered patterns without the substitution of drugs with the token *#drug#*.

3.8 System Improvements

After the competition, we performed further experiments in order to improve our results. The first experiment consisted in changing the number of trees and number of attributes considered in each iteration for Random Forest. We found that better results were obtained when setting the number of iterations to 5,000, and the number of attributes considered in each iteration to the default value: $logm+1$, with m = total number of attributes.

MFSs were extracted with the following parameters: $min_{freq} = 5$, $min_{length} = 2$, $max_{length} = 7$ and with $gap = 0$ and $gap = 1$. Also, we run the experiments without clustering, in order to see if we could skip that step and still get good results.

In the case of $gap = 0$ without clustering, we had 2,409 MFS. In the case of $gap = 1$ without clustering, we obtained over 13,000 MFSs, and this made our model's training inviable. Therefore, we needed to somehow reduce the number

of MFS. For doing so, we detected groups of MFSs of the same length that appeared in the exact same set of sentences. Out of each group, we kept only one of the MFSs. With this co-occurrence pruning, we reduced the number of MFSs to 7,913.

Figure 3.8 shows Precision over Recall for each configuration of the system. Figure 3.9 shows F-Measure for different confidence threshold values. Table 3.17 presents a numerical description of results were we can see, for each configuration Precision and Recall for the best F-Measure value, AUC-PR, Precision at Recall 0.80 and Recall at Precision 0.80.

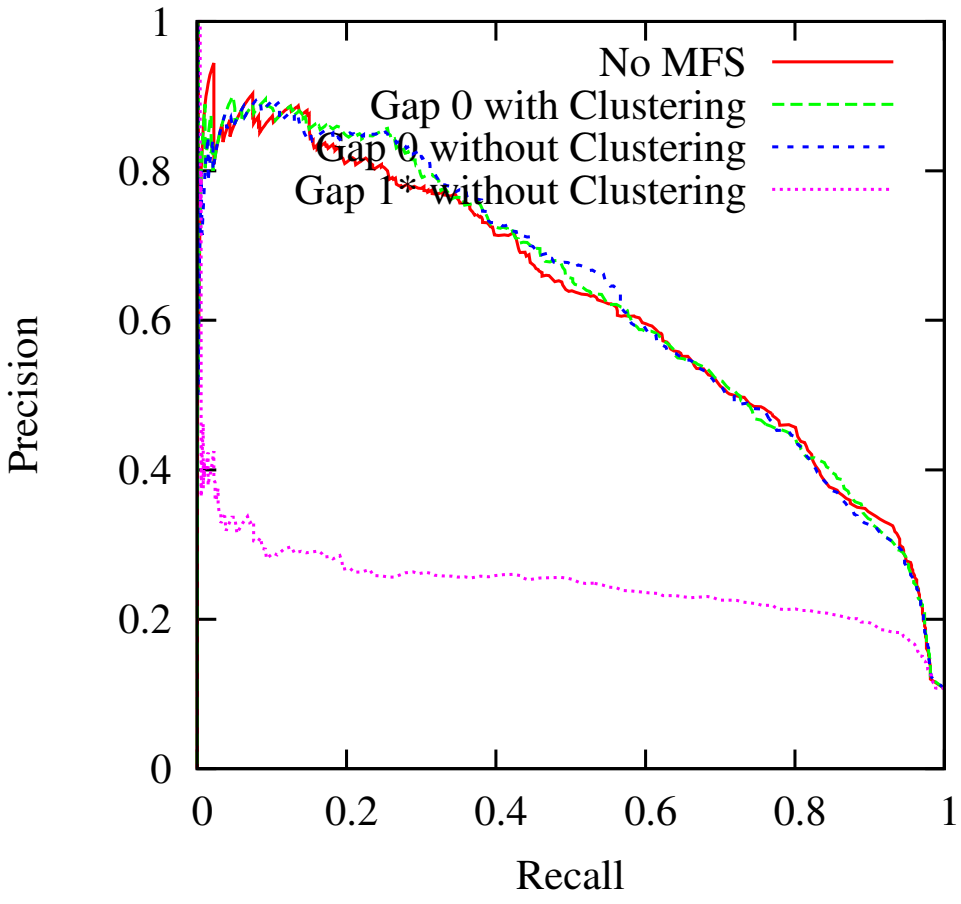

Figure 3.8: Precision-Recall curves for the results given by our system for the test, for different configurations: without MFS, MFS with $gap = 0$ and clustering, MFS with $gap = 0$ without clustering and MFS with $gap = 1$ with co-occurrence pruning (*) and without clustering.

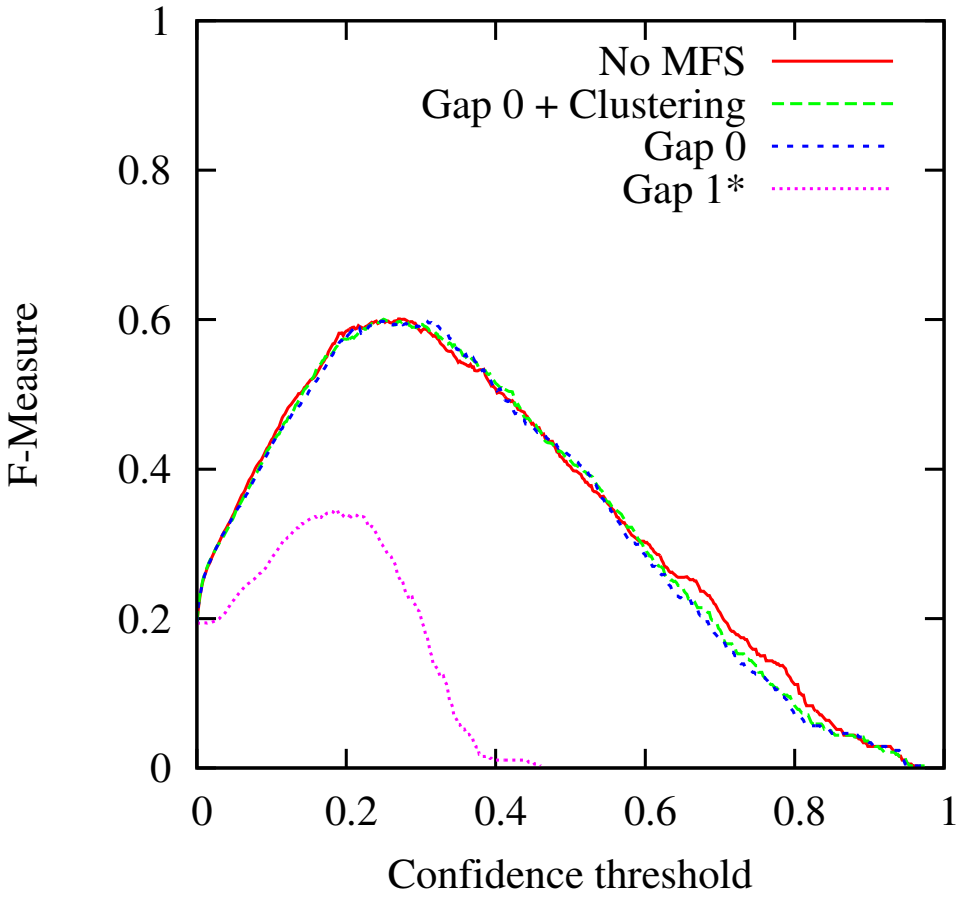

Figure 3.9: F-measure curves for the results given by our system for the test, for different configurations: without MFS, MFS with $gap = 0$ and clustering, MFS with $gap = 0$ without clustering and MFS with $gap = 1$ without clustering and with co-occurrence pruning (*).

Table 3.17: Performance measures for test with, for different configurations of the system: without MFS, MFS with $gap = 0$ and clustering, MFS with $gap = 0$ without clustering and MFS with $gap = 1$ without clustering.

	P	**R**	**F**	**AUC-PR**	**P@R 0.8**	**R@P 0.8**
Without MFS	0.5828	0.6207	0.6012	0.6253	0.4569	0.2546
MFS $gap = 0$ **(clust.)**	0.5365	0.6817	0.6005	**0.6323**	0.4415	0.2971
MFS $gap = 0$	0.6415	0.5623	0.5993	**0.6323**	0.4441	0.0145
MFS $gap = 1$	0.2303	0.6830	0.3445	0.2503	0.2139	0.0053

As we can see, performance of MFS with $gap = 1$ and co-occurrence pruning is extremely low. This is probably because there were still too many attributes to train the model correctly.

As with respect to clustering, we can observe that given a big number of iterations, in this case 5,000, adding the clustering step does not improve the results. In fact, using clustering produced a model with much better Recall, but with an equivalent loss of Precision. Therefore, for the sake of simplicity, we discarded the clustering step.

MFS with $gap = 0$ produced a slight improvement according to AUC-PR, with respect to the system without using MFS. We think this can be caused by the fact that there are thrice as many attributes with MFS $gap = 0$. That leads to the next section, were we try to mitigate this problem with the use of MFDS.

3.9 Applying MFDS

As explained in Section 2.5, it is not crucial to find the maximal sequences, but the most relevant. For this reason, we extracted Maximal Frequent Discriminative Sequences from the document collection. The discriminative power criterion used was Information Gain. With this technique, we extracted a total of 1,190 MFDS for $gap = 0$ and 2,335 MFDS for $gap = 1$ which is a significant reduction compared to MFS. Figure 3.10 plots Precision over Recall. As we can observe, in some points of the curve, the configuration with MFS and $gap = 0$ goes over the other configurations, but in general, the value of AUC-PR shows that MFDS with $gap = 1$ outperforms the rest.

Figure 3.11 shows F-Measure for different confidence threshold values.

Table 3.18: Performance measures for test with, for different configurations of the system: without MFS, MFS with $gap = 0$ and clustering, MFS with $gap = 0$ without clustering and MFS with $gap = 1$ without clustering.

	P	**R**	**F**	**AUC-PR**	**P@R 0.8**	**R@P 0.8**
Without MFS	0.5828	0.6207	0.6012	0.6253	0.4569	0.2546
MFS $gap = 0$	0.6415	0.5623	0.5993	0.6323	0.4441	0.0145
MFDS $gap = 0$	0.5210	0.7069	0.5999	0.6214	0.4383	0.2759
MFDS $gap = 1$	0.5124	0.7109	0.5956	**0.6347**	0.4495	0.3395

Table 3.18 presents a numerical description of results.

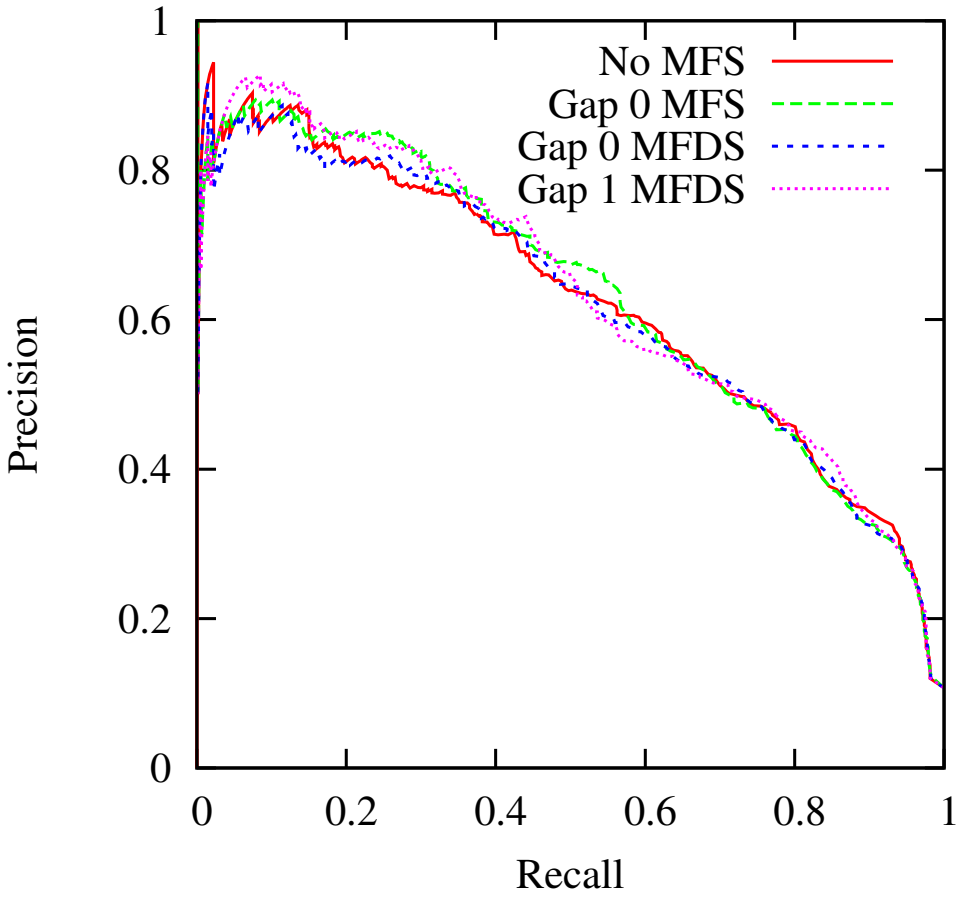

Figure 3.10: Precision-Recall curves for the results given by our system for the test, for different settings: MFDS with $gap = 0$ and MFDS with $gap = 1$, compared to without MFDS nor MFS and the best performing setting for MFS that was with $gap = 0$.

We observe a considerable Recall improvement as we would expect using $gap = 1$. This can be seen in the 8% improvement of Recall in the optimal point for F-Measure, as well as the 8.5% improvement in Recall for 80% Precision. We can also see a reduction in Precision, nevertheless the overall performance is still better according to AUC-PR.

MFDS have made possible the use of sequences of $gap = 1$ by producing a reduced and more relevant set of sequences.

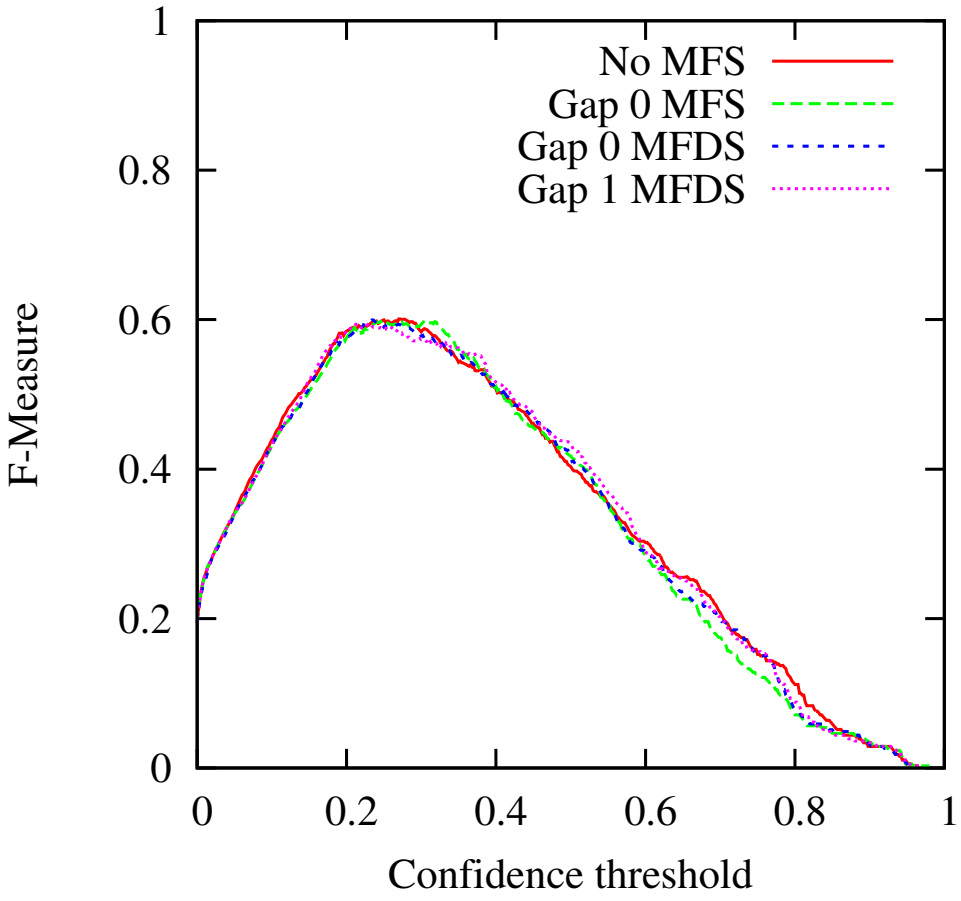

Figure 3.11: F-measure curves for the results given by our system for the test, for different settings: without MFS, with MFS $gap = 0$, with MFDS $gap = 0$ and MFDS $gap = 1$.

3.10 Conclusions

In this chapter, we have proposed two solutions to identify DDI. The first solution determines whether or not a sentence included a drug-drug interaction description. The second solution performs DDI extraction, and determines whether or not two given drugs in a sentence interact.

In Section 3.5 we described a system to determine whether or not a sentence contained a drug-drug interaction. Maximal frequent sequences obtained moderately good results, reaching a Precision of 0.68 with a Recall of 0.41, and Preci-

sion of 0.46 with 0.95 Recall. These results are promising for a first approximation taking into account that they can be improved with further preprocessing of the sentences, such as POS-tagging or stemming.

In Section 3.7 we presented a system for DDI extraction based on machine learning with bag-of-words, maximal frequent sequences and other features. This was developed for the *DDIExtraction2011* competition. Our submission obtained a F-Measure of 0.5829 and a AUC-PR of 0.6341 for the test corpus, obtaining the 6th position in the participants ranking. Our system can be set up to reach Recall of 0.3205 with a Precision of 0.8, or Precision of 0.4309 and a Recall 0.8. The use of maximal frequent sequences increased AUC-PR by 0.02.

One of the main problems we have encountered during the research was the complexity of the language structures used in biomedical literature. Most of the sentence contained appositions, coordinators, etc. Therefore, it was very difficult to reflect those structures using maximal frequent sequences. The reduced size of the corpus was also a serious limitation for our approach.

MFS could be an useful tool for representing such type of information, however, as we relax MFS constraints, we obtain an unmanageable amount of attributes. We approached this problem using clusters of MFS, which produced a considerable Recall increase, but the overall performance considering Precision did not produce better results. Also, the fact that MFS extraction is performed without knowledge of corpus annotations causes that a high amount of the extracted patterns are irrelevant for classification.

In order to solve the limitations of MFS for classification, we introduced the concept of MFDS where our MFS extraction algorithm was extended to consider a discriminative criterion. In our case, we used Information Gain. This produced a list of more relevant patterns, and a manageable total amount of them. MFDS made practical the use sequences with $gap = 1$, which produced an 8% improvement of Recall. Also, with MFDS training the model is much faster due to the reduced number of features. These results suggest that a further relaxation of MFDS constraints, *i.e.*, $gap > 1$, as well as more sophisticated pruning strategies should be the next steps.

Our system should be improved by complementing it with other state of the art techniques used in the Protein-Protein Interaction field that have not been explored yet during our research, such as character n-grams and co-occurrences. Also, we could find patterns in the paths of the dependency trees.

Chapter 4

Conclusions and Further Work

4.1 Conclusions

A drug-drug interaction occurs when the effects of a drug are modified by the presence of other drugs. DDIs can decrease therapeutic benefit or efficacy of treatments and this could have very harmful consequences in the patient's health that could even cause the patient's death. Knowing the interactions between prescribed drugs is of great clinical importance, therefore it is very important to keep databases up-to-date with respect to new DDI.

Maximal frequent sequences are an iteresting tool since they can represent the most important parts of texts. Given a text collection, the fact that there are sequences that are repeated in some of the texts shows how relevant is the information that those MFS describe. MFS have a wide applicability since the technique is domain and language independent. The fact that they are sequences and not strings, i.e., they allow gap between words, makes them more flexible and therefore they can capture higher level patterns.

In this thesis we have presented a new algorithm for MFS extraction inspired in the GSP algorithm (Agrawal et al., 1996). Our algorithm allows gaps between the items of the sequences, making the MFS more flexible and therefore enabling them to capture common sentence patterns rather than just repeated sentences. We have further modified the algorithm to handle continuous events, where each item has a *timestamp* instead of a position. This is, going from items distributed in a discrete way to items distributed continuously with possible overlaps.

During the development of this work we have also analized some of the limi-

tations of MFS. When looking for Maximal Frequent Sequences, it is difficult to balance the minimum frequency threshold, and sometimes the MFS extracted are more restrictive than we would like them to be. When using MFS as input for a predictive model, it is not crucial to find the maximal sequences, but the most relevant ones. In this case, frequency by itself is not a good criterion to stop looking into longer sequences. For these reasons, we modified the algorithm in order to introduce a discriminative power criterion, that will determine whether or not the growth of a sequence should continue. This algorithm retrieves the sequences that have more discriminative power respect to the corpus they are extracted from. We name Maximal Frequent Discriminative Sequences the patterns that are extracted using this process. In order to calculate the discriminative power of a frequent sequence we have used Information Gain; nevertheless other discriminative criteria, such as Information Gain Ratio, should be explored.

With the help of MFS we have developed two systems to solve two common problems in the field of DDI:

1) DDI Sentence Identification This problem consists in determining whether or not a sentence incudes a drug-drug interaction description.

2) DDI Extraction This problem consists in determining whether or not two given drugs in a sentence interact.

MFS are able to capture complex patterns such as multi-word terms, or grammatical patterns. Our hypothesis held that we can model these patterns as common subsequences with high probability of either describing DDI or not describing it. We have developed both systems based on MFS.

The first system, to approach the DDI Sentence Identification problem, was completely based on MFS. For this approximation we generated three different versions of the corpus. The first one consisted in replacing all drugs by their type (*e.g. phsu*, *antb*, *clnd*, ...). We named this version of the corpus *6drugs*. The second version of the corpus consisted in replacing all drugs by the token *#drug#* and we named it *#drug#*. The third version of the corpus was the original version, *i.e.*, each drug was with its original name, and we named this version *norm*.

We obtained moderately good results, reaching a Precision of 0.68 with a Recall of 0.41 with the MFS extracted from the *norm* version of the corpus, and Precision of 0.46 with 0.95 Recall with the MFS extracted from the *#drug#* version of the corpus.

We believe these results could be further improved by building a classifier that takes into account the MFS extracted from each different version of the corpus, since the *norm* version obtains a high Recall and the *#drug#* version of the corpus obtains high Precision.

The second system presented in this thesis, to approach DDI Extraction, was based on machine learnnig. The system had different features, namely bag of words, word categories, MFS, token and char level features, as well as drug level features. The classifier we used was a Random Forest. We participated with this system at the First DDI Challenge 2011 competition and obtained 6th position, with 0.6122 Precision, 0.5563 Recall and 0.5829 F-Measure and 0.6341 AUC-PR.

The system was tested with and without MFS in order to determine the influence MFS had. It obtained an increase of 0.02 in AUC-PR with MFS. However, MFS were able to capture structures of the sentences that bag of words were not able to describe.

Further improvements of the DDI Extraction system were made by replacing the MFS extracted with MFDS. This produced a list of more relevant patterns, and a manageable total amount of them. MFDS made practical the use sequences with $gap = 1$, which produced an 8% improvement of Recall. Also, with MFDS training the model was much faster.

We could either balance Precision and Recall or improve one at the expense of the other. Depending on the kind of application that we are developing, we can be more interested in one or the other. For example, if we are building an autonomous application, without human supervision, to automatically tag sentences containing drug-drug interactions, we might want to have a high precision. However, if we can not afford losing sentences with drug-drug interaction, even if we retrieve also sentences that do not contain them, then we should go for the parameters that give high recall but less precision.

In general, we believe that MFDS are able to capture high level patterns repeated in the sentences and therefore are a good tool not only for DDI Sentence Identification and DDI Extraction, but also in other applications where patterns can be described with item sequences. We also believe that MFDS is the natural extension of the MFS concept when applied to classification tasks, and that it should be our main track for further research.

4.2 Further Work

Further work for the algorithm to extract MFDS:

1) The algorithm is in its early stage. Other discriminative power criteria, such as Information Gain Ratio, must be tested.

2) In the version of the algorithm presented, in order to merge two candidate sequences, the new generated candidate sequence must have a higher discriminative power. Candidate sequences are merged until the discriminative power does not increase anymore. It could be the case that, even if the discriminative power does not increase in iteration k, it could increase in further iterations. We believe an interesting experiment would be to not stop merging when the discriminative value decreases, but trying for a few more iterations if the discriminative value increases. If it does not, then go back to the highest discriminative value found.

3) In our current approach, when a merged k-length sequence has no higher Information Gain than their $(k-1)$-length subsequences, both subsequence are kept. This leads to redundant patterns. New strategies to prune these sequence should be explored.

Further work for the preprocessing of the corpus:

1) Each word of a sentence can be replaced by its POS tag. This way, we would be able to extract, with the MFS algorithm, structures of sentences that are repeated in a grammatical way.

2) The preprocessing of the corpus could be integrated in the MFS/MFDS extraction algorithm in the form of taxonomies, as used in the GSP algorithm. This would enable the patterns to mix tokens with different levels of preprocessing, for example a word, a POS tag and a *#drug#* token. Nevertheless, this would be a challenge when it comes to algorithm time complexity.

Further work for experimentation:

1) All the experiments in this thesis have been tested with a closed test dataset. More comprehensive experimentation should be performed, with 10-fold

cross-validation, in order to see a more accurate results for each configuration. This way, we would be able to do a better and more informed decision when selecting the best configuration of the systems.

2) With cross-validation, perform an exhaustive search of the best parameters for the algorithm, *i.e.*, number of iterations for Random Forest and number of attributes to consider in each iteration. It would be specially useful to investigate the relation between number of attributes, iterations and performance.

3) When performing DDI Sentence Identification, we modified the corpus in order to obtain three different versions: *norm*, *#drug#*, *6drugs*. We obtained a high Recall with the *norm* version of the corpus and high Precision with the *#drug#* version. Building a classifier that takes into account the MFS extracted from each different version of the corpus could improve the overall results.

4) We think an interesting experiment would be to merge the two approaches into one DDI Extraction approach. The first system could be used to retrieve the potential sentences that describe DDI and use those sentences to extract the DDI with the help of the second system.

5) Even thought the algorithm adapted to continuous events has already been used internally at Bitsnbrains S.L. with a private corpus, we would like to apply it to public corpora to be able to analyze its performance and compare it with approximations of other authors.

6) Our results suggest that a further relaxation of MFDS constraints, *i.e.*, $gap > 1$, as well as more sophisticated pruning strategies should be the next steps.

7) Our system should be improved by complementing it with other state of the art techniques used in the Protein-Protein Interaction field that have not been explored yet during our research, such as character n-grams and co-occurrences. Also, we could find patterns in the paths of the dependency trees.

Further work about the DrugDDI corpus:

Generating a corpus of these characteristics was a big contribution of Segura-Bedmar (2010). The corpus was annotated by one experienced pharmacist. Nevertheless, as we have seen, the corpus has some inaccuracies that can affect the performance of the system, both in the training phase and in the evaluation phase. We would like to set up a comprehensive review of the corpus by reviewing every tagged drug and the tagged interactions. Each possible DDI should be tagged by at least 3 annotators with 2/3 agreement on each sample. Such a set up would allow us to 1) improve and/or guarantee the quality of the tags; 2) measure the difficulty of the task.

References

Agrawal, Rakesh and Ramakrishnan Srikant (1994). "Fast Algorithms for Mining Association Rules in Large Databases". In: *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 487–499.

Agrawal, Rakesh, Hiekki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo (1996). "Fast Discovery of Association Rules". In: *Advances in Knowledge Discovery and Data Mining*. Ed. by M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy. Menlo Park, CA. USA: AAAI/MIT Press, pp. 307–328.

Ahonen-Myka, Helena (1999). "Finding All Maximal Frequent Sequences in Text". In: *Proceedings of the 16th International Conference on Machine Learning (ICM-99)*. Slovenia, pp. 11–17.

— (2002). "Discovery of Frequent Word Sequences in Text." In: *Pattern Detection and Discovery*. Ed. by David J. Hand, Niall M. Adams, and Richard J. Bolton. Vol. 2447. Lecture Notes in Computer Science. London, UK: Springer Verlag, pp. 180–189.

Antunes, Cláudia and Arlindo L. Oliveira (2003). "Generalization of Pattern-Growth Methods for Sequential Pattern Mining with Gap Constraints". In: *Machine Learning and Data Mining in Pattern Recognition*. Ed. by Petra Perner and Azriel Rosenfeld. Vol. 2734. Lecture Notes in Computer Science. Springer, pp. 239–251.

Arighi, Cecilia, Kevin Cohen, et al., eds. (2010). *Proceedings of BioCreative III Workshop*. Bethesda, MD, USA.

Aronson, Alan R. (2001). "Effective Mapping of Biomedical Text to the UMLS Metathesaurus: the MetaMap Program." In: *Proceedings of the AMIA Symposium*. Bethesda, MD, USA., pp. 17–21.

Bjorne, Jari, Antti Airola, Tapio Pahikkala, and Tapio Salakoski (2011). "Drug-Drug Interaction Extraction from Biomedical Texts with SVM and RLS Classifiers". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 35–42.

Breiman, Leo (2001). "Random Forests". In: *Machine Learning* 45.1, pp. 5–32.

Bui, Quoc-Chinh, Sophia Katrenko, and Peter M. A. Sloot (2011). "A Hybrid Approach to Extract Protein–Protein Interactions". In: *Bioinformatics* 27.2. Code available at http://staff.science.uva.nl/ bui/PPIs.zip, pp. 259–265.

Chowdhury, Faisal Mahbub and Alberto Lavelli (2011). "Drug-drug Interaction Extraction Using Composite Kernels". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 27–33.

Chowdhury, Faisal Mahbub, Asma Ben Abacha, Alberto Lavelli, and Pierre Zweigenbaum (2011). "Two Different Machine Learning Techniques for Drug-Drug Interaction Extraction". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 19–26.

Coyotl-Morales, Rosa M., Luis Villaseñor Pineda, Manuel Montes-y Gómez, and Paolo Rosso (2006). "Authorship Attribution using Word Sequences". In: *Proceedings of the 11th Iberoamerican Congress on Pattern Recognition, CIARP*. Vol. 4224. Lecture Notes in Computer Science. Springer, pp. 844–853.

Davis, Jesse and Mark Goadrich (2006). "The Relationship Between Precision-Recall and ROC Curves". In: *Proceedings of the 23rd international conference on Machine Learning, ICML*. Pittsburgh, Pennsylvania, pp. 233–240.

Duda, Stephany., Constantin. Aliferis, Olph. Miller, and Alexander Statnikov (2005). "Extracting Drug-Drug Interaction Articles From MEDLINE to Improve the Content of Drug Databases". In: *AMIA Annual Symposium Proceedings*. Vol. 2005, pp. 216–220.

Fayyad, Usama M., Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy (1996). *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press.

García-Blasco, Sandra (2009). "Extracción de Secuencias Maximales de una Colección de Textos". Final Degree Project. Valencia, Spain: Univerisad Politécnica de Valencia.

García-Blasco, Sandra, Roxana Danger, and Paolo Rosso (2010). "Drug-Drug Interaction Detection: A New Approach Based on Maximal Frequent Se-

quences". In: *Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN)* 45, pp. 263–266.

García-Blasco, Sandra, Santiago M. Mola-Velasco, Roxana Danger, and Paolo Rosso (2011). "Automatic Drug-Drug Interaction Detection: A Machine Learning Approach With Maximal Frequent Sequence Extraction". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 51–58.

García-Hernández, René, José Martínez-Trinidad, and Jesús Carrasco-Ochoa (2004). "A Fast Algorithm to Find All the Maximal Frequent Sequences in a Text". In: *Progress in Pattern Recognition, Image Analysis and Applications*. Ed. by Alberto Sanfeliu, José Martínez Trinidad, and Jesús Carrasco Ochoa. Vol. 3287. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 305–320.

García-Hernández, René A., J. F. Martínez Trinidad, and J. A. Carrasco-Ochoa (2006). "A New Algorithm for Fast Discovery of Maximal Sequential Patterns in a Document Collection". In: *International Conference on Intelligent Text Processing and Computational Linguistics, CICLing*. Lecture Notes in Computer Science, pp. 514–523.

García-Hernández, René Arnulfo (2007). "Desarrollo de Algoritmos para el Descubrimiento de Patrones Secuenciales Maximales". PhD thesis. Puebla, México: Instituto Nacional de Astrofísica, Óptica y Electrónica.

Gunopulos, Dimitrios, Roni Khardon, Heikki Mannila, Sanjeev Saluja, Hannu Toivonen, and Ram Sewak Sharma (2003). "Discovering All Most Specific Sentences". In: *ACM Transactions on Database Systems* 28.2, pp. 140–174.

Hakenberg, Jörg, Robert Leaman, Nguyen Ha Vo, Siddhartha Jonnalagadda, Ryan Sullivan, Christopher Miller, Luis Tari, Chitta Baral, and Graciela Gonzalez (2010). "Efficient Extraction of Protein-Protein Interactions from Full-Text Articles". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 7.3, pp. 481–494.

Hauben, Manfred. and Xiaofeng Zhou (2003). "Quantitative Methods in Pharmacovigilance: Focus on Signal Detection". In: *Journal of Drug Safety* 26, pp. 159–86.

Hernández-Reyes, Edith, René Arnulfo García-Hernández, Jesús Ariel Carrasco-Ochoa, and José Francisco Martínez Trinidad (2006a). "Document Clustering Based on Maximal Frequent Sequences". In: *5th International Confrence on*

Natural Language Processing. Vol. 4139. Lecture Notes in Artificial Intelligence. Springer-Verlag, pp. 257–267.

Hernández-Reyes, Edith., J. Ariel Carrasco-Ochoa, J. Francisco Martínez Trinidad, and René Arnulfo García-Hernández (2006b). "Document Representation Based on Maximal Frequent Sequence Sets". In: *XI Iberoamerican Congress on Pattern Recognition, CIARP*. Vol. 4225. Lecture Notes in Computer Science. Springer-Verlag, pp. 854–863.

Karunaratne, Thashmee (2011). "Is Frequent Pattern Mining useful in building predictive models?" In: *Collective Learning and Inference on Structured Data, CoLISD*. Athens, Greece.

Ledeneva, Yulia, Alexander Gelbukh, and René Arnulfo García-Hernández (2008). "Terms Derived from Frequent Sequences for Extractive Text Summarization". In: *Proceedings of CICLing 2008*. Lecture Notes in Computer Science. Berlin, Germany: Springer-Verlag, pp. 593–604.

Mannila, Heikki and Hannu Toivonen (1997). "Levelwise Search and Borders of Theories in Knowledge Discovery". In: *Data Mining and Knowledge Discovery*. Vol. 1. 3, pp. 241–258.

Mannila, Heikki, Hannu Toivonen, and A. Inkeri Verkamo (1994). "Efficient Algorithms for Discovering Association Rules". In: *Asociation for the Advancement of Artificial Intelligence, Workshop on Knowledge Discovery in Databases*. AAAI Press, pp. 181–192.

— (1995). "Discovering Frequent Episodes in Sequences". In: *Proceedings of the 1st international conference on Knowledge Discovery and Data mining, KDD*. Montreal, Ont. Canada: AAAI Press, pp. 210–215.

Mata, Jacinto, Ramón Santano, Daniel Blanco, Marcos Lucero, and Manuel J. Maña (2011). "A Machine Learning Approach to Extract Drug-Drug Interactions in an Unbalanced Dataset". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 59–65.

Minard, Anne-Lyse, Lamia Makour, Anne-Laure Ligozat, and Brigitte Grau (2011). "Feature Selection for Drug-Drug Interaction Detection Using Machine-Learning Based Approaches". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 43–50.

Mortazavi-Asl, Behzad, Jianyong Wang, Helen Pinto, Qiming Chen, and Mei-Chun Hsu (2004). "Mining Sequential Patterns by Pattern-Growth: The Pre-

fixSpan Approach". In: *IEEE Transactions on Knowledge and Data Engineering* 16.11, pp. 1424–1440.

Neves, Mariana, José M. Carazo, and Alberto Pascual-Montano (2009). "Extraction of Biomedical Events Using Case-Based Reasoning". In: *Proceedings of the BioNLP'09 Shared Task on Event Extraction Workshop at NAACL-HLT*. Boulder, CO, USA, pp. 68–76.

Pei, Jian, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu (2001). "PrefixSpan: Mining Sequential Patterns by Prefix-Projected Growth." In: *ICDE*. IEEE Computer Society, pp. 215–226.

Pirmohamed, Munir and Michael Orme (1998). "Drug Interactions of Clinical Importance". In: *Davies's Textbook of Adverse Drug Reactions. 5th*. Ed. by D. Davies, R. Ferner, and H. de Glanville. London: Chapman & Hall Medical, pp. 888–912.

Pyysalo, Sampo, Antti Airola, Juho Heimonen, Jari Björne, Filip Ginter, and Tapio Salakoski (2008). "Comparative Analysis of Five Protein-Protein Interaction Corpora". In: *BMC Bioinformatics* 9.Suppl 3:S6.

Rodríguez Terol, A., M.O. Caraballo Camacho, D. Palma Morgado, and Others (2009). "Quality of Interaction Database Management Systems". In: *Farmacia hospitalaria: órgano oficial de expresión científica de la Sociedad Española de Farmacia Hospitalaria* 33. In Spanish, pp. 134–146.

Sánchez-Cisneros, Daniel, Isabel Segura-Bedmar, and Paloma Martínez (2006). "DDIExtractor: A Web-Based Java Tool for Extracting Drug-Drug Interactions from Biomedical Texts". In: *Natural Language Processing and Information Systems*. Vol. 6716. Lecture Notes in Computer Science. Springer, pp. 274–277.

Segura-Bedmar, I., P. Martínez, and D. Sánchez-Cisneros (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*. Vol. 761. Huelva, Spain, pp. 1–9.

Segura-Bedmar, Isabel (2010). "Application of Information Extraction Techniques to Pharmacological Domain: Extracting Drug-Drug Interactions". PhD thesis. Madrid, Spain: Universidad Carlos III Madrid.

Segura-Bedmar, Isabel, Paloma Martínez, and César de Pablo-Sánchez (2010). "Extracting Drug-Drug Interactions from Biomedical Texts." In: *BMC Bioinformatics* 11 (Suppl. 5).

Seno, Masakazu and George Karypis (2002). "SLPMiner: An Algorithm for Finding Frequent Sequential Patterns Using Length-Decreasing Support Constraint". In: *In Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM)*, pp. 418–425.

Stockley, HI (2007). *Stockley's Drug Interactions*. By Karen Baxter. Pharmaceutical Press.

Tatonetti, N. P., J. C. Denny, S. N. Murphy, G. H. Fernald, G. Krishnan, V. Castro, P. Yue, P. S. Tsau, I. Kohane, D. M. Roden, and Others (2011). "Detecting Drug Interactions From Adverse-Event Reports: Interaction Between Paroxetine and Pravastatin Increases Blood Glucose Levels." In: *Clinical Pharmacology & Therapeutics* Vol. 90.Num. 1, pp. 133–142.

Thomas, Philippe, Mariana Neves, Illés Solt, Domonkos Tikk, and Ulf Leser (2011). "Relation Extraction for Drug-Drug Interactions using Ensemble Learning". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 11–18.

Wishart, D.S., C. Knox, A.C. Guo, D. Cheng, S. Shrivastava, D. Tzur, B. Gautam, and M. Hassanili (2008). "DrugBank: A Knowledgebase for Drugs, Drug Actions and Drug Targets". In: *Nucleic Acids Research (Database Issue)* 36.

Yang, Guizhen (2006). "Computational Aspects of Mining Maximal Frequent Patterns". In: *Theoretical Computer Science* 362.1-3, pp. 63–85.

Zaki, Mohammed J. (Nov. 2000). "Sequence Mining in Categorical Domains: Incorporating Constraints". In: *Proceedings of the 9th International Conference on Information and Knowledge Management*. Washington D.C, USA, pp. 422–429.

— (2001). "SPADE: An Efficient Algorithm for Mining Frequent Sequences". In: *Machine Learning Journal, Special Issue on Unsupervised Learning*. Vol. 42, pp. 31–60.

Zhang, Rong and Alexander I. Rudnicky (2002). "A Large Scale Clustering Scheme for Kernel K-Means". In: *16th Conference on Pattern Recognition*. Vol. 4. Los Alamitos, CA, USA: IEEE Computer Society, pp. 289–292.

Appendix A

Contributions

During the research described in this thesis, we produced the following publications:

Esposti, Mirko Degli, Roxana Danger, Rosso Paolo, and Sandra Garcia-Blasco (2010). "Visual characterization of biomedical texts with word entropy". In: *Network Tools and Applications in Biology (NETTAB 2010), Biological Wikis*. Ed. by Angelo Facchiano and Paolo Romano. Napoli, Italy, pp. 139–142.

García-Blasco, Sandra, Roxana Danger, and Paolo Rosso (2010). "Drug-Drug Interaction Detection: A New Approach Based on Maximal Frequent Sequences". In: *Sociedad Española para el Procesamiento del Lenguaje Natural (SEPLN)* 45, pp. 263–266.

García-Blasco, Sandra, Santiago M. Mola-Velasco, Roxana Danger, and Paolo Rosso (2011). "Automatic Drug-Drug Interaction Detection: A Machine Learning Approach With Maximal Frequent Sequence Extraction". In: *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*. Vol. 761. Huelva, Spain, pp. 51–58.