Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2013

Tiago Santos Barata Nunes

Recuperação de informação baseada em frases para textos biomédicos

A sentence-based information retrieval system for biomedical corpora

Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2013

Tiago Santos Barata Nunes

Recuperação de informação baseada em frases para textos biomédicos

A sentence-based information retrieval system for biomedical corpora

“*The problems are solved, not by giving new information, but by arranging what we have known since long.*”

— **Ludwig Wittgenstein**

Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2013

Tiago Santos Barata Nunes

Recuperação de informação baseada em frases para textos biomédicos

A sentence-based information retrieval system for biomedical corpora

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica do Doutor José Luís Oliveira, Professor Associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Sérgio Matos, Investigador Auxiliar da Universidade de Aveiro.

Para os meus pais, José e Mila, por permitirem que eu tenha aqui chegado e por todo o apoio incondicional ao longo deste percurso. Sem vocês este documento não existiria.

o júri / the jury

presidente / president	Tomás António Mendes Oliveira e Silva Professor Associado da Universidade de Aveiro
vogais / examiners committee	Erik M. van Mulligen Professor Auxiliar do Medical Informatics Department do Erasmus Medical Center Rotterdam (arguente principal)
	Sérgio Guilherme Aleixo de Matos Investigador Auxiliar da Universidade de Aveiro (co-orientador)

agradecimentos / acknowledgements

Writing a research thesis about a relatively complex subject is not an easy task and is certainly not something one can do alone.
I would like to thank first my professors and supervisors José Luís Oliveira and Sérgio Matos, for the invaluable guidance and support through all the process of writing this thesis. You helped me to stay focused and on path, especially when I felt overwhelmed with all the different possibilities and directions that could be pursued.
To my external advisors Erik van Mulligen and Jan Kors, for making me feel at home during my short internship at the Biosemantics Research Group, for being always available, supportive and helping me ask the right questions at the right time.
To my colleagues at the University of Aveiro Bioinformatics Group and the ones at the Erasmus Medical Center in Rotterdam. You helped me professionally and personally with our countless conversations, both about work and other topics.
A big thank you to all my friends, especially the closest ones, for encouraging me to focus on work and joining me in those unforgettable stress-relieve moments that allowed me to keep going.
Finally I want to thank my family for the unconditional support and interest in my progress. A special mention goes to my parents and sister, who made all this possible and gave me the necessary strength and motivation to reach the end of this stage of my life.

Palavras Chave

recuperação de informação, extração de informação, mineração de texto, aprendizagem automática, processamento de linguagem natural, bioinformática.

Resumo

O desenvolvimento de novos métodos experimentais e tecnologias de alto rendimento no campo biomédico despoletou um crescimento acelerado do volume de publicações científicas na área. Inúmeros repositórios estruturados para dados biológicos foram criados ao longo das últimas décadas, no entanto, os utilizadores estão cada vez mais a recorrer a sistemas de recuperação de informação, ou motores de busca, em detrimento dos primeiros. Motores de pesquisa apresentam-se mais fáceis de usar devido à sua flexibilidade e capacidade de interpretar os requisitos dos utilizadores, tipicamente expressos na forma de pesquisas compostas por algumas palavras.

Sistemas de pesquisa tradicionais devolvem documentos completos, que geralmente requerem um grande esforço de leitura para encontrar a informação procurada, encontrando-se esta, em grande parte dos casos, descrita num trecho de texto composto por poucas frases. Além disso, estes sistemas falham frequentemente na tentativa de encontrar a informação pretendida porque, apesar de a pesquisa efectuada estar normalmente alinhada semanticamente com a linguagem usada nos documentos procurados, os termos usados são lexicalmente diferentes.

Esta dissertação foca-se no desenvolvimento de técnicas de recuperação de informação baseadas em frases que, para uma dada pesquisa de um utilizador, permitam encontrar frases relevantes da literatura científica que respondam aos requisitos do utilizador. O trabalho desenvolvido apresenta-se em duas partes. Primeiro foi realizado trabalho de investigação exploratória para identificação de características de frases informativas em textos biomédicos. Para este propósito foi usado um método de aprendizagem automática. De seguida foi desenvolvido um sistema de pesquisa de frases informativas. Este sistema suporta pesquisas de texto livre e baseadas em conceitos, os resultados de pesquisa apresentam-se enriquecidos com anotações de conceitos relevantes e podem ser ordenados segundo várias estratégias de classificação.

Keywords

information retrieval, information extraction, text mining, machine learning, natural language processing, bioinformatics.

Abstract

Modern advances of experimental methods and high-throughput technology in the biomedical domain are causing a fast-paced, rising growth of the volume of published scientific literature in the field. While a myriad of structured data repositories for biological knowledge have been sprouting over the last decades, Information Retrieval (IR) systems are increasingly replacing them. IR systems are easier to use due to their flexibility and ability to interpret user needs in the form of queries, typically formed by a few words.

Traditional document retrieval systems return entire documents, which may require a lot of subsequent reading to find the specific information sought, frequently contained in a small passage of only a few sentences. Additionally, IR often fails to find what is wanted because the words used in the query are lexically different, despite semantically aligned, from the words used in relevant sources.

This thesis focuses on the development of sentence-based information retrieval approaches that, for a given user query, allow seeking relevant sentences from scientific literature that answer the user information need. The presented work is two-fold. First, exploratory research experiments were conducted for the identification of features of informative sentences from biomedical texts. A supervised machine learning method was used for this purpose. Second, an information retrieval system for informative sentences was developed. It supports free text and concept-based queries, search results are enriched with relevant concept annotations and sentences can be ranked using multiple configurable strategies.

Contents

List of Figures

List of Tables

Acronyms

NLP	Natural Language Processing
TM	Text Mining
MEDLINE	Medical Literature Analysis and Retrieval System Online
PMC	PubMed Central
GO	Gene Ontology
UMLS	Unified Medical Language System
IR	Information Retrieval
IE	Information Extraction
TC	Text Classification
BST	Binary Search Tree
AI	Artificial Intelligence
ML	Machine Learning
NER	Named Entity Recognition
POS	Part-of-speech
QA	Question Answering
RE	Relation Extraction
WSD	Word Sense Disambiguation
ZA	Zone Analysis
TF-IDF	Term Frequency-Inverse Document Frequency
REPL	Read-Eval-Print Loop
Phrank	PhraseRank
FS	Feature Selection
RFE	Recursive Feature Elimination
SVM	Supervised Vector Machine
ROC	Receiver Operating Characteristic
SGD	Stochastic Gradient Descent
kNN	K-Nearest Neighbors
AUC	Area Under the ROC Curve
UI	User Interface
REST	Representational State Transfer
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
AJAX	Asynchronous JavaScript and XML
DOM	Document Object Model
MVP	Model-View-Presenter
MeSH	Medical Subject Headings

Chapter One

Introduction

1.1 Motivation

During the past decades we have witnessed an overwhelming growth of publicly available data and information, prompted by the evolution and fast-paced adoption of the Internet. These data are presented in various digital formats, both structured and unstructured, with the vast majority of it being published in the form of natural language texts.

The life sciences, comprising the fields of science that involve the study of living organisms, such as biomedicine, molecular biology and genetics, is one of the areas experiencing the most accentuated increase in the amount of scholarly knowledge. This was mainly prompted by modern advances of high-throughput technology that produce sizable amounts of biological data, making it very challenging for healthcare professionals and the research community to keep up with the latest advances in the field.

In response, a myriad of databases and automated tools have been developed in a effort to help users quickly and efficiently find and retrieve relevant information and publications regarding biomedical concepts and their relationships. However, considerable fragmentation of those knowledge resources still exists and their integration is a current challenge, as is also the development of better tools to find and present information buried in those resources.

Traditional information retrieval systems return entire documents, which may require considerable time of subsequent reading to find the specific information that users seek, frequently contained in a small passage of a few sentences. Additionally, IR often fails to find the most relevant documents to answer a certain information need, because the words used by users to express their needs are lexically different, despite semantically aligned, from the words used in relevant sources.

Everyday we take one step forward towards the utopian world where we can easily find all available knowledge about a given topic and then navigate the web of relationships leading to other topics. In this utopian world, we'll have tools that not only facilitate access to knowledge resources, but also automatically discover new relationships between known concepts, requiring only human confirmation to validate those relations.

1.2 Objectives

The purpose of this thesis is the research and development of information extraction and retrieval methodologies that can leverage the wealth of available knowledge in the biomedical domain and facilitate access to unstructured information from scientific publications. This research was partly conducted at the Bioinformatics Group of the University and Aveiro and the resulting techniques and methods will augment the local Text Mining (TM) framework. Thus, whenever possible, tools from this framework will be used and extended. Research will be focused on the following goals:

- To explore and study literature on text mining applied to the life sciences domain, aiming to understand past and current areas of research in the field;

- Study existing solutions for sentence-based information retrieval systems, with focus on identifying the features of an informative sentence;
- Investigate and develop methods to classify and rank sentences according to their semantic richness and information content;
- Implement a modular information retrieval system that allows concept-based queries and facilitates ranking of sentences from the literature according to different user information needs.

These goals will be addressed in chapter 3, where we present the developed system and methods.

1.3 Thesis outline

This thesis is divided in 4 chapters. The remaining chapters are organized according to the following:

- **Chapter 2** introduces background topics on which this thesis builds upon, such as information retrieval, information extraction, their techniques and applications, and related systems for sentence retrieval;
- **Chapter 3** presents the developed methods and system, from the classifiers used to identify and rank informative sentences to the complete information retrieval system;
- **Chapter 4** discusses our findings, presents overall conclusions and directions for future work.

Chapter Two

Background

"Knowledge has to be improved, challenged, and increased constantly, or it vanishes."

— Peter Drucker

A 2004 study performed by the Pew Research Center reported that 92 percent of Internet users say the Internet is a good place to search for everyday information [1]. The same study declares that 88 percent of online users consider the Internet plays a role in their daily lives. Since modern search engines considerably accelerate access to publicly available information and knowledge resources, it is understandable that people choose to satisfy immediate information needs online, instead of looking for offline resources. Recent studies from Pew Research (2011) also report that search and email are the top online tasks for adults with Internet.

The ever-increasing use of the Internet by people all over the world facilitates sharing of new discoveries, especially in the form of research papers and studies. However, the amount of new information made available everyday is overwhelming and makes it challenging both to find the most relevant results to our information needs and to keep up with progress in any given field. This problem is known as information overload, and represents the difficulties a person can have in understanding a given topic and making decisions in the presence of too much information.

This issue is of particular importance in the biomedical field, where the dissemination of research results is crucial. The past decades have witnessed an exponential increase of data, made available by new high-throughput methods. Hence, when the need for information systems capable of storing and providing access to this tremendous amount of data emerged, new biomedical databases have been created and are continuously populated with new discoveries and study results. For instance the Medical Literature Analysis and Retrieval System Online (MEDLINE) contains over 19 million references to journal articles in life sciences with a concentration on biomedicine. If we consider PubMed, which contains MEDLINE, the number of citations surpasses 22 million. Over the past 20 years, PubMed has been growing at a rate of approximately 4 percent per year [2].

It is estimated that more than 85 percent of existing information is available as unstructured data [3], making it very challenging for automated systems to make sense of it. For that kind of information, specific approaches involving techniques such as Natural Language Processing (NLP) are required. Those approaches are collectively known as Text Mining (TM), which is defined by Hearst as the automatic discovery of new, previously unknown information from unstructured textual data [4].

TM is often seen as encompassing three major tasks: Information Retrieval (IR), which deals with gathering relevant documents for a given information need; Information Extraction (IE), responsible for extracting information of interest from these documents; and Data Mining (DM), which is the task of discovering new associations among the extracted pieces of information.

2.1 Information Extraction

"I was brought up to believe that the only thing worth doing was to add to the sum of accurate information in the world."

— Margaret Mead

Information Extraction (IE) is the task of automatically extracting structured information, such as entities, relationships between entities, and attributes describing entities, from unstructured documents. In the majority of the cases, especially in the bioinformatics domain, it involves processing human language texts by using Natural Language Processing (NLP) techniques. Structured information resulting from IE can be used to populate databases, enrich documents with annotations and external references, build relationship networks between concepts, among others. Figure 2.1 illustrates a typical IE system.

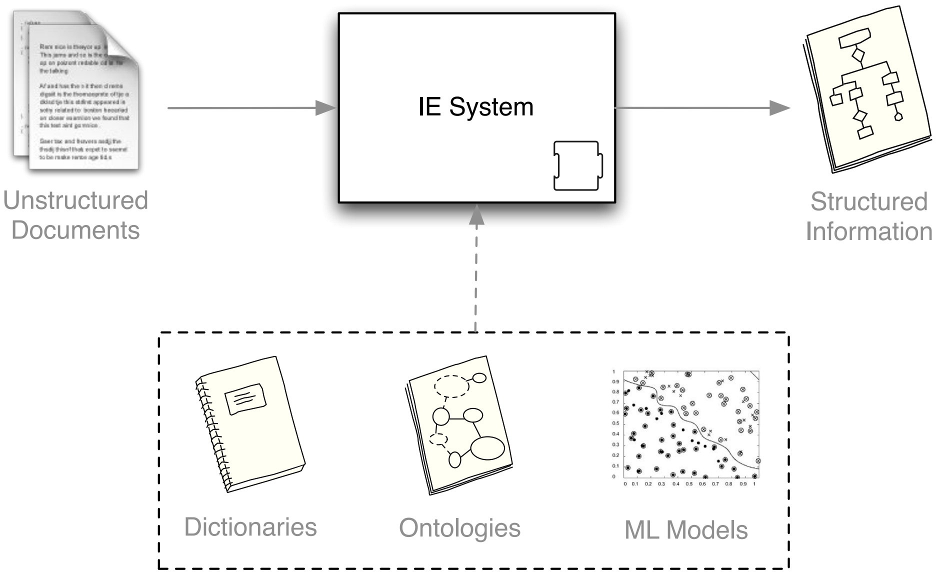


Figure 2.1: Overview of a typical IE system. Starting with unstructured documents, an IE system can use a combination of different methods to identify and extract structured information that is made available as the system output.

Depending on the type and purpose of the information being extracted, an IE application can use various auxiliary resources to assist in the extraction process. For example, systems specialized in identifying known entities in text often use dictionaries or ontologies to aid in their identification. More advanced systems may try to learn language models from manually annotated documents and later use those models to extract information from previously unseen text. Humans are very good at recognizing entities, concepts, facts and relationships in text, however, given the inherent complexity of human language, it is very challenging to build automated systems that are able to correctly perform those seemingly simple tasks.

This section will briefly introduce some of the most common tasks of IE from unstructured text documents, with a focus on scholarly documents – like research papers and reports – in the life sciences domain.

2.1.1 Natural Language Processing

Natural Language Processing (NLP) is a field of computer science, artificial intelligence and linguistics that deals with automated understanding of human language by computers. It is one of the most difficult problems in computer science, since human language evolved to be so complex that the task of completely understanding it is considered "AI-complete" [5], i.e., is equivalent to that of solving the central artificial intelligence problem – making computers as intelligent as people.

Although work on the area can be found from earlier periods, the first relevant developments of NLP started around 1950. However, until the 1980s, most NLP systems resorted only to complex sets of manually crafted rules. Then, the advent and explosion of Artificial Intelligence (AI) ignited a revolution in the area with the introduction of Machine Learning (ML) algorithms for language processing, partly due to the increase in computing power necessary for such approaches. While the first developments of language processing using ML used decision trees that produced sets of rules similar to the hand-written ones, research in the area quickly shifted towards statistic and probabilistic models that have the advantage of being able to express the relative certainty of many different possible answers rather than only one, leading to more reliable results when included as components of a larger system. Nowadays, research has increasingly focused on unsupervised and semi-supervised learning algorithms that, while often producing lower quality results, have the advantage of being able to learn from non-annotated data.

NLP encompasses a myriad of tasks, with some high-level tasks that can be directly translated to real-world problems solutions and lower-level tasks that can be seen as subtasks that help in solving larger tasks. The area has received a lot of attention from the research community and there are numerous challenges and competitions aimed at solving specific tasks. The most commonly researched tasks in the biomedical field include:

- **Automatic summarization**, aiming at producing readable summaries containing the main ideas from larger texts;
- **Co-reference resolution**, finds words in text mentioning the same entities;
- **Named Entity Recognition (NER)** , identifies mentions of known entities in text, such as proper names or concepts;
- **Part-of-speech (POS) tagging** focuses on determining the part of speech of each word in a phrase, i.e., tagging words as nouns, verbs or adjectives;
- **Parsing** performs grammatical analysis of sentences in order to generate parse trees with dependencies between terms;
- **Question Answering (QA)** is one the most difficult tasks that tries to give concrete answers to open-ended questions expressed in natural language;
- **Relation Extraction (RE)** , along with **Event Extraction** try to identify relationships between entities mentioned in text and occurrences of specific bio-events, respectively;
- **Word Sense Disambiguation (WSD)** , typically tries to use context to infer the specific meaning of a word in a specific sentence. This is also an hard task, given that words can have multiple meanings;
- **Zone Analysis (ZA)** , is a task usually applied to research papers with the intent of identifying different sections and rhetorical zones, like introduction, problem statement, methods, results and conclusion.

The following sections detail the NLP tasks more relevant for the work proposed and developed in the context of this thesis.

2.1.2 Named Entity Recognition

Named Entity Recognition (NER) is the sub-task of IE that focuses on identifying known entity names in structured and unstructured documents. A NER module starts by isolating atomic elements in text and then classifies them as belonging to one or more predefined categories. Figure 2.2 illustrates a

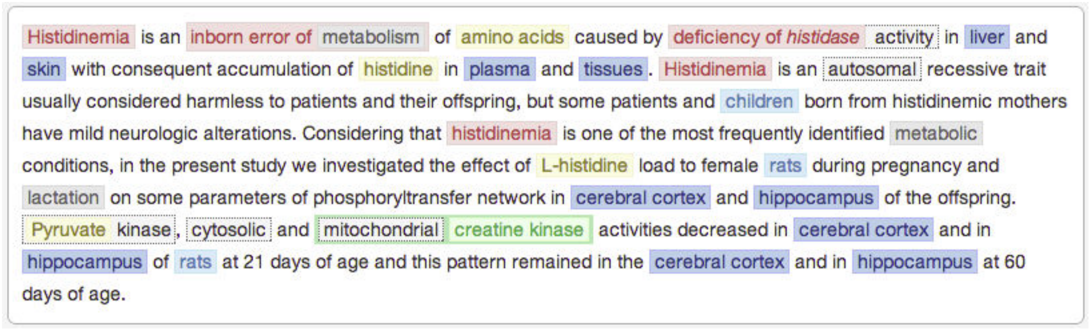

Histidinemia is an inborn error of metabolism of amino acids caused by deficiency of *histidase* activity in liver and skin with consequent accumulation of histidine in plasma and tissues. Histidinemia is an autosomal recessive trait usually considered harmless to patients and their offspring, but some patients and children born from histidinemic mothers have mild neurologic alterations. Considering that histidinemia is one of the most frequently identified metabolic conditions, in the present study we investigated the effect of L-histidine load to female rats during pregnancy and lactation on some parameters of phosphoryltransfer network in cerebral cortex and hippocampus of the offspring. Pyruvate kinase, cytosolic and mitochondrial creatine kinase activities decreased in cerebral cortex and in hippocampus of rats at 21 days of age and this pattern remained in the cerebral cortex and in hippocampus at 60 days of age.

Figure 2.2: Text fragment annotated with known biomedical concepts. Colours are used to facilitate visual discrimination of different entity types. Screen capture from BeCAS online tool [6].

fragment of text where known biomedical entities have been recognized and highlighted using a colour-coded scheme.

NER systems can recognize generic entities such as names of persons, organizations, locations, quantities and expressions of time or focus on a given domain and identify specific entity types. For instance, a biomedical NER tool usually specializes in the identification of biological entities, such as species, disorders, metabolic pathways, chemicals, enzymes, genes and proteins, among others.

Multiple techniques have been shown to be effective in the implementation of automatic NER systems, ranging from rule-based (usually using grammatical features) to dictionary-based (resorting to large collections of known names), and statistical methods that use machine learning algorithms [7]–[9]. Systems that try to leverage the advantages of each individual method by combining them are called hybrid systems.

Rule-based methods rely on a set of linguistic rules that can combine orthographic, syntactic, grammatical and semantic properties of human language to identify mentions of entities in text. These systems require comprehensive knowledge of natural language structure and normally lead to very specific rules that may not generalize well on different kinds of text [10].

Dictionary-based approaches require a large collection of known entities, usually including several names by which each entity is known (synonyms). Then, strict or partial matching techniques can be used to find occurrences of those names in text. The performance of this type of approach depends highly on the quality and comprehensiveness of the dictionaries used, as well as the string matching techniques employed [11].

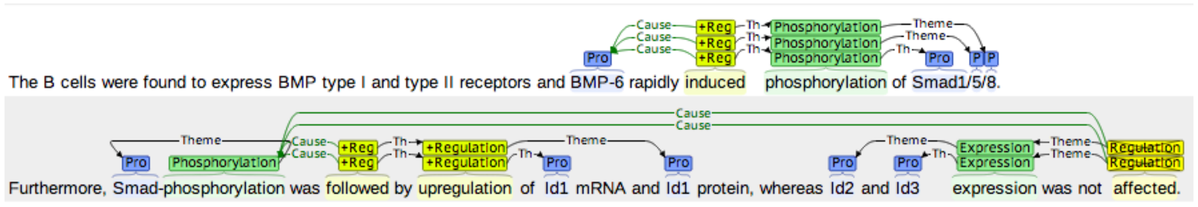
Finally, machine-learning techniques use statistical methods to learn how to recognize certain entity names. This usually requires large datasets of texts where known entities have been manually annotated in order to produce a model capable of automatically identifying entities in unseen text.

After the identification of a textual entity it is usual desirable to link it to a concept, possibly with references to resources – like public knowledge bases – where one can read more about it. This process is called entity linking, or named entity normalization, and typically requires some kind of dictionaries to map the matched entities to the corresponding database identifiers.

2.1.3 Relationship and Event Extraction

After named entities have been recognized, normalized and tagged in text, we can extract relationships between them. Plenty of work has been conducted in biomedical relationship extraction and several different approaches have been tried with different degrees of success. From all types of relationships exploited so far, protein-protein interactions are arguably the ones that received more attention from the research community, and several methods to extract them have been proposed [12]–[18]. Attempts have also been made in the automatic recognition and extraction of relationships between protein-gene, genotype-phenotype, genotype-drug, phenotype-drug, gene-phenotype-drug, drug-drug and disease-treatment entities [19]–[23].

More complex relationships concerning the detailed behaviour of bio-molecules are known as bio-molecular events (bio-events). They usually establish a relationship between various named entities and are connected by a verb that is considered the event trigger. Community evaluation challenges like BioNLP have promoted research on extraction of several types of bio-molecular events, such as gene expression, transcription, protein catabolism, phosphorylation, localization, binding, positive and negative regulation [24], [25]. Figure 2.3 illustrates two sentences annotated with bio-events.



The B cells were found to express BMP type I and type II receptors and BMP-6 rapidly induced phosphorylation of Smad1/5/8.

Furthermore, Smad-phosphorylation was followed by upregulation of Id1 mRNA and Id1 protein, whereas Id2 and Id3 expression was not affected.

Figure 2.3: Text fragment annotated with relationships between concepts and biomedical events. Arrows are used to connect related concepts while certain verbs specify relation/event types. Image produced using brat tool [26].

2.2 Information Retrieval

"It is a very sad thing that nowadays there is so little useless information."

— Oscar Wilde

Information retrieval is the task of finding the most relevant documents that satisfy a particular information need from a collection of information resources. Those documents can be of any type, ranging from unstructured data, such as research papers and webpages, to structured documents, like database records or strictly defined XML documents.

Typical IR systems accept an user query, that can be based on metadata (inherently structured), full-text (unstructured) or a combination of both, retrieve a set of documents that are in some way related to the user query, rank those documents using a specific algorithm and return the ranked results back to the user. Three of the most widely used general-purpose IR systems are the web-based search engines Google, Yahoo and Bing. Figure 2.4 illustrates the workflow of a typical IR system.

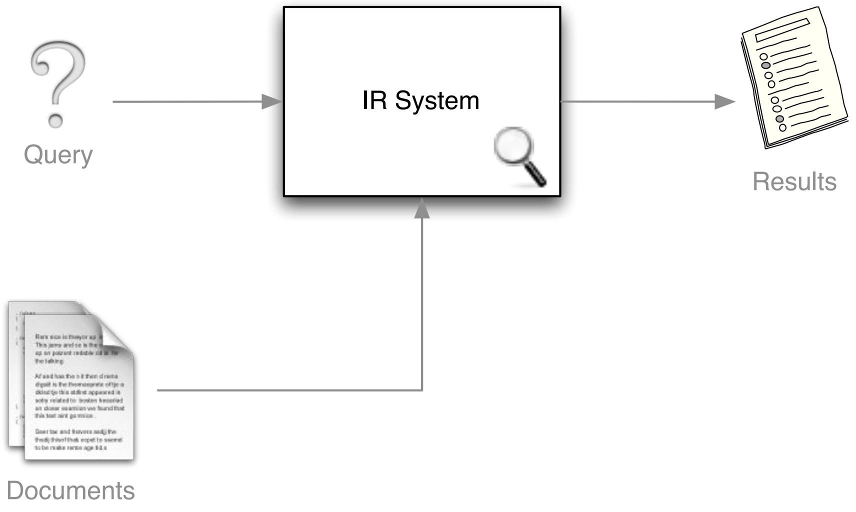


Figure 2.4: Overview of a typical IR system. Documents are indexed beforehand. When a user submits a query the system selects relevant documents from its index, ranks them and returns them to the user.

In order to provide search results to users, search engines need to build an index of documents, which is a data structure containing all the indexed documents, document surrogates or relevant metadata. Depending on the application, those documents can be text documents, images, audio, videos or structured objects of some type. General-purpose search engines that index the web perform an additional step before indexing, called crawling, which is the process of following references from one document to another in order to find all documents that may be relevant for indexing.

2.2.1 Model Types

The goal of any IR system is the correct retrieval of the most relevant documents for a given user query from a large collection of documents. Moreover, since each user query – representing an information need – does not usually map to a single result, it is important that, from all the documents matching a query, the ones more likely to be aligned with the user information need appear first in the result list. For this, IR systems need a way to measure the similarity between each document and the initial query.

Various models exist to address the similarity issue and they vary in the manner documents are represented and in the techniques used to calculate the relatedness of each document to a given query. Figure 2.1 categorizes multiple IR models according to their mathematical basis and the properties of each model. The mathematical basis of a model defines how documents are formally represented and the operations used to measure similarity. Typically, an IR system can fall in one of four types:

- **Algebraic:** documents and queries are usually represented as vectors, matrices, or tuples, and the similarity of the query vector and document vector is represented as a scalar value;
- **Set-theoretic:** represent documents as sets of words or sentences, and similarities are derived from set-theoretic operations on those sets;
- **Probabilistic:** the process of document retrieval is treated as a probabilistic inference, and similarities are calculated as the probabilities of a document being relevant for a given query;
- **Feature-based:** documents are viewed as vectors of values of feature functions and the model tries to find the best way to combine these features into a single relevance score, typically by learning to rank methods. Other retrieval models can be incorporated as new feature functions.

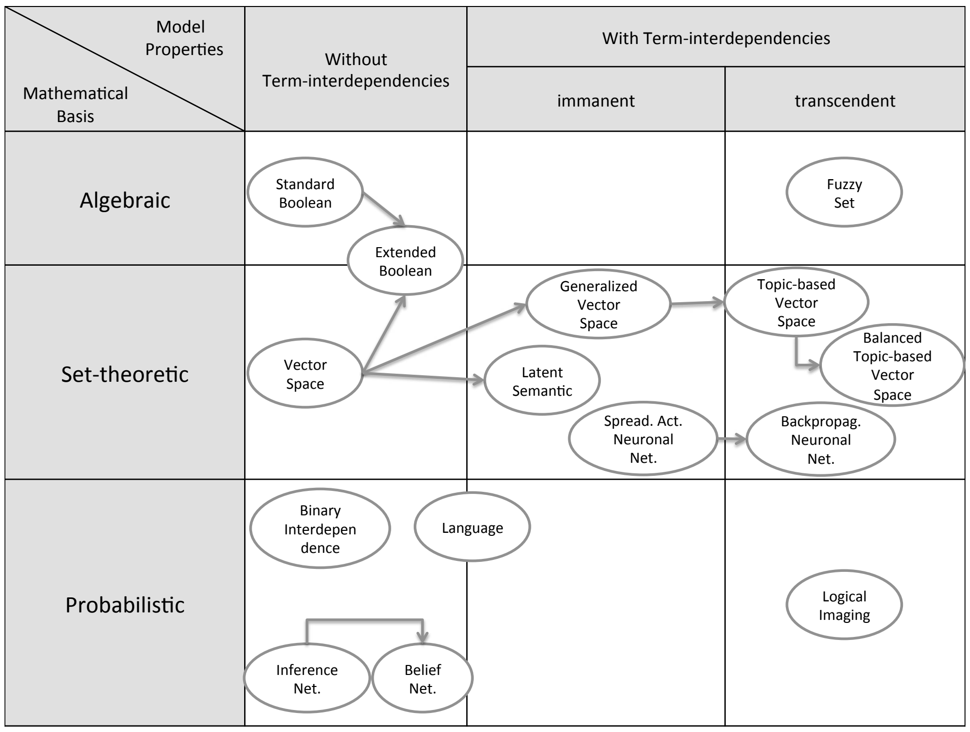
Depending on the types of documents being searched and the specific goals of the system, a model can make several assumptions about the interdependence of terms both in the query and in documents.

Models without term-interdependencies treat different terms/words as independent. This requirement can be represented in vector space models by assuming term vectors are orthogonal and in probabilistic models by an independency assumption for term variables.

Models considering interdependencies between terms can make it in an immanent or transcendent manner. A model considering immanent term interdependencies needs to represent those dependencies internally while defining the degree of interdependency between two terms. It is usually directly or indirectly derived from the co-occurrence of those terms in the whole set of documents. Models with transcendent term interdependencies allow a representation of interdependencies between terms, but they do not allege how the interdependency between two terms is defined. They rely on external sources for the degree of interdependency between two terms, like a trained human or sophisticated algorithms.

The models illustrated in Table 2.1 are comprehensively detailed and compared in the literature [27], [28].

Table 2.1: Information Retrieval Models (based on [29]).



2.2.2 Evaluating Performance

Many different measures for evaluating the performance of IR systems have been proposed. All measures require a collection of documents – the corpus – and a query. A ground truth notion of relevance is assumed – every document in the corpus is either relevant or non-relevant for the query. Figure 2.5 shows a Venn diagram illustrating the different sets of documents involved in the performance evaluation of a given system.

The portion of documents that are correctly selected for a query are called the true positives (TP), and is visible in the diagram as the intersection between the two ellipsis. True negatives (TN) are the documents that are correctly left out of the retrieved set, and are represented in the diagram by everything except the union of the two ellipses. Documents that are incorrectly retrieved are called false positives (FP) and can be seen as the "Retrieved Documents" ellipse minus the intersection. False negatives (FN) are documents that should have been retrieved but were not, and are visible in the diagram as the "Relevant Documents" ellipse minus the intersection.

These sets are the basis for all metrics required to evaluate an IR system [30]. A very important metric is accuracy, which is defined as the ratio of true measures ($TP + TN$) in relation to the corpus. The proportion of selected documents that were correctly selected is called precision, and is calculated by Formula 2.1.

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

Recall is the proportion of correct items that were selected from the corpus, and is formulated as Formula 2.2.

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

Precision and recall usually represent a trade-off, such that an IR system that always returns all documents in the corpus achieves maximum recall at the expense of very low precision. Hence,

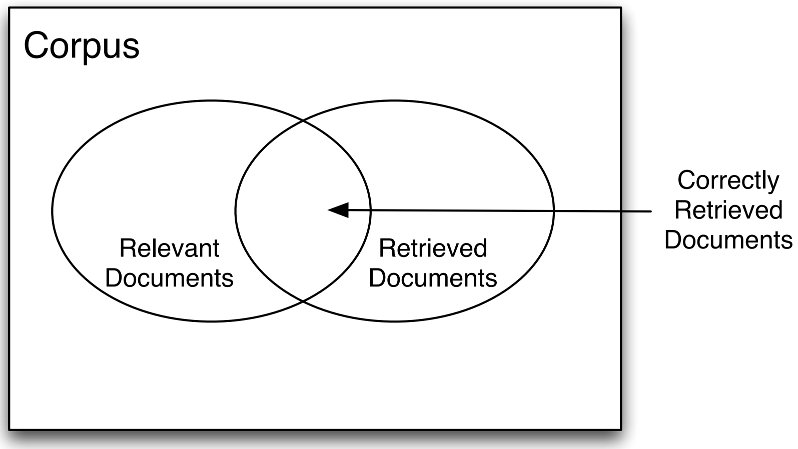


Figure 2.5: Venn diagram illustrating the outcome of an information retrieval system. The correctly retrieved documents are at the intersection of the set of relevant documents for the query and the set of retrieved documents.

another metric is commonly used to combine precision and recall using a weighted harmonic mean, the F1-measure. Formula 2.3 show the equation used to calculate this measure.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.3}$$

The proportion of non-relevant documents that are retrieved, out of all non-relevant documents in the corpus is named fall-out. Fall-out is sometimes used as a measure of how hard it is to build a system that produces few false positives, and is calculated with Formula 2.4.

$$\textit{Fall-out} = \frac{FP}{TN + FP} \tag{2.4}$$

Since IR systems usually return a ranked list of documents, it is desirable to measure performance while considering the order in which the documents are presented. However, precision and recall are scalar values based on the complete result set returned by the system, disregarding results order. By calculating precision and recall at every point in the ranked list of documents, we can plot a precision-recall curve, seeing precision $p(r)$ as a function of recall. If we then compute the average value of $p(r)$ over the interval from $r = 0$ to $r = 1$ we get the area under the precision-recall curve. This area is called average precision and is formulated by the integral in Formula 2.5.

$$AveP = \int_0^1 p(r)\,\mathrm{d}r \tag{2.5}$$

In practice, to do an objective evaluation of an IR system we need a corpus, a set of queries and the lists of correct results that should be retrieved for each query. We can then calculate a well-known metric used to assess the overall performance of the system, the mean average precision. For a set of Q queries, the mean average precision of a system is the mean of the average precision scores for each query, as formulated by Formula 2.6.

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \tag{2.6}$$

In the absence of a test collection to evaluate an IR system, we can perform an empiric analysis of retrieved results and try to determine if they answer information needs for certain queries. This does not provide an objective, measurable evaluation of system performance, but is sometimes the only way to estimate retrieval and ranking effectiveness.

2.2.3 Indexing and Searching

To efficiently retrieve documents relevant for a user query an IR system must build and maintain an index of documents optimized for fast searching. When dealing with large document collections (called corpus, in the context of IR) it is impractical to simply inspect every document at query time and try to match its contents to the query. For this reason documents must be parsed at index time in order to extract their contents into data structures suitable for quick matching.

One of the most common data structures used by IR applications are inverted indexes [28]. An inverted index is a list of all words present in a corpus along with references to every document in which they occur. The set of references of a given word is called its posting list. To support better ranking algorithms, posting lists can include occurrence counts for each word in each document. If postings also include term positions, an IR system is capable of performing proximity matches for user queries. Figure 2.6 illustrates a simple inverted index with occurrence counts.

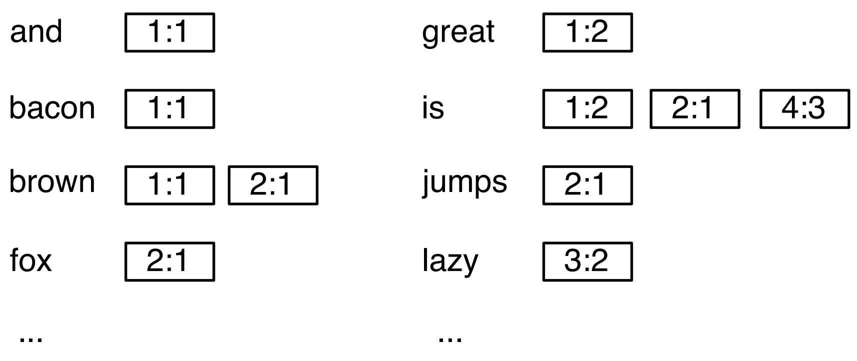


Figure 2.6: Inverted index with occurrence counts. Maps each term occurring in the corpus to the documents in which it occurs, including the occurrence count.

Other data structures commonly used by search engines are Binary Search Trees (BSTs) (illustrated in Figure 2.7), which have the advantage of supporting partial matching; and signature files – or hash tables – which have very fast lookup times (O(1), faster than trees) but have the drawback of only supporting exact matches (Figure 2.8).

The process of building an index can be very computationally intensive, both in terms of memory consumption, processor load and input/output operations. When building large indexes (e.g. indexing the whole web, like Google does) it is crucial that the indexing workflow is parallel and distributed over multiple machines. There are several algorithms that can help parallelize and distribute tasks and load between various nodes, being MapReduce one of the most widely used [31].

Techniques used for index traversal, necessary to perform searches, depend on the structure used for indexing documents [28]. Thus, a different procedure is applied to retrieve documents for each index structure. Additionally, if index structures are compressed in order to reduce their memory footprint, both in persistent storage and in volatile memory, the search process needs to be adapted [27].

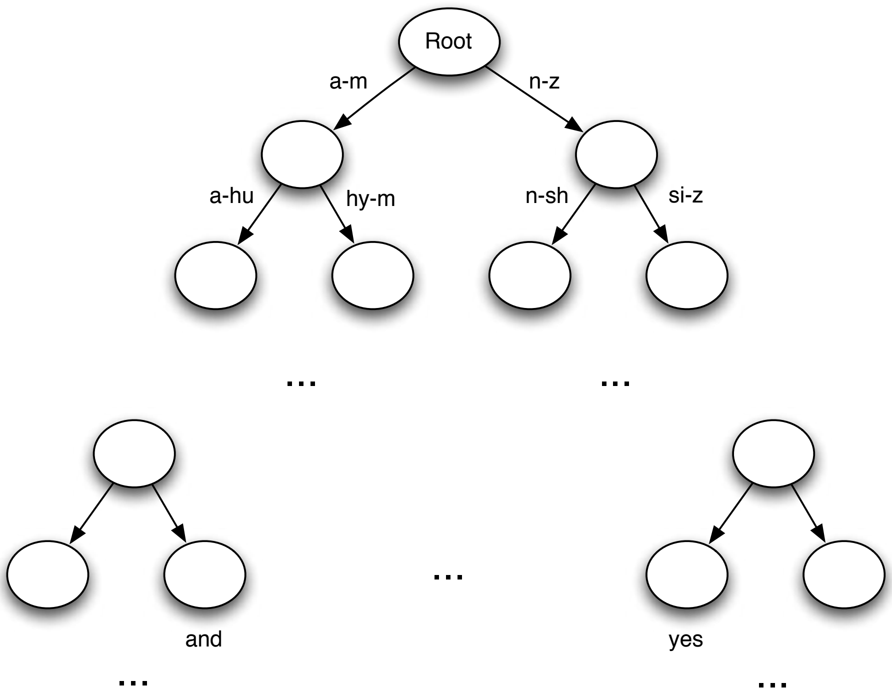


Figure 2.7: Sample BST. Can have as many branches as necessary. More branches provide more fine-grained partial matching.

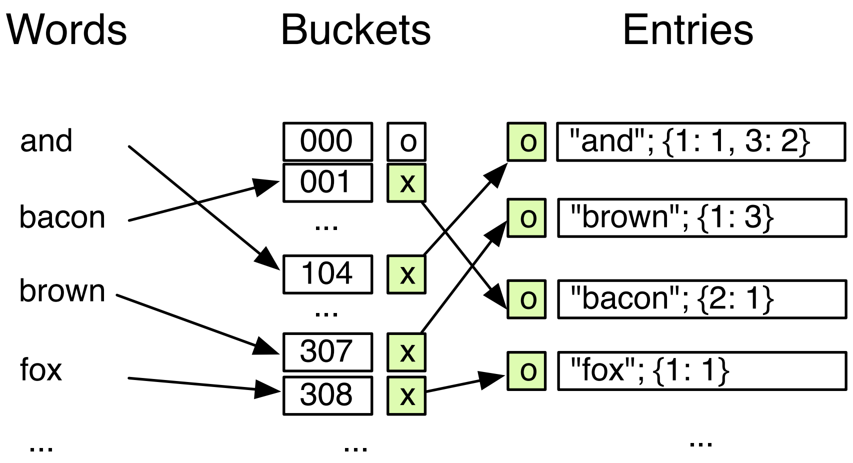


Figure 2.8: Sample search hash table. Each word is hashed to an integer value and mapped to an entry containing the posting list. The hashing algorithm should be strong enough to avoid collisions.

2.2.4 Query Expansion

Query expansion is the process of reformulating an initial user query (the seed query) with the intent of improving retrieval performance. Most of the times, the seed query is augmented with more terms that match additional documents. This leads to an improvement in recall performance at the expense of reducing precision.

Several techniques can be used standalone or be combined in order to improve query expansion results. The most widely used ones include:

- Expanding the seed query with **synonyms** of the words present in it;
- **Stemming** all terms in the initial query to allow finding all morphological forms of the words in the indexed documents;
- Mapping terms to concepts and expanding the initial query with **related concepts**. An ontology or concept network is usually used for this purpose;
- Automatically **correct spelling errors** in the seed query and search for the correct form, or suggest a corrected version;
- **Re-weight the terms/concepts** in the initial query to alter the ranking of results and promote better matches at the top of the result list.

2.2.5 Text Classification

Text Classification (TC) , also called text categorization, is the task of classifying documents into predefined classes. An algorithm that implements classification is known as a classifier. Typical applications of TC are e-mail spam filtering, automatic categorization of newspaper articles into topics and language identification.

Text classification can be performed manually, by humans, or automatically, using computer algorithms. There are two approaches to perform automatic TC. We can use rule-based or automatic methods, based on ML techniques.

Rule-based classification methods are usually very accurate when rules are written by experts. However, creating such classification systems requires human labour, is extremely time consuming and demands both linguistic and domain-specific knowledge of the text being classified [32]. Moreover, if the target domain changes, rules have to be reconstructed, which sometimes causes conflicts between rules and makes classifier maintenance costly.

Machine learning based approaches have the advantage of being domain independent and usually achieve high predictive performance [33]. Nonetheless, the generation of automatic classification models requires labelled data, called training data, which maps samples to categories and allows a learning algorithm to use statistical methods to automatically induce a classification model from that data [34].

Previous studies that compared the effectiveness of various text classification algorithms on different tasks concluded that there is not one single classification method that can consistently outperform others on all possible classification tasks [35], [36]. The key question when dealing with ML classification is not whether a learning algorithm is superior to others, but under which conditions a particular method can significantly outperform others on a given application problem.

The evaluation metrics presented in section 2.2.2 are also used to evaluate the performance of classification methods. In this case, the true positives (TP) and true negatives (TN) sets are composed by the instances that have been correctly classified as either belonging or not belonging to a given class, respectively. Accordingly, the false positives (FP) and false negatives (FN) are the instances that have been incorrectly classified.

2.2.6 Sentence Retrieval Applications

The most popular biomedical information retrieval system, PubMed, combines the Medical Subject Headings (MeSH) based indexing provided by the MEDLINE literature database with Boolean and vector space models for document retrieval [37], giving researchers access to over 22 million citations.

Even though PubMed provides an extensive, up-to-date and efficient search interface, it has become increasingly challenging for researchers to quickly identify information relevant to their individual needs, owing mainly to the ever-growing biomedical literature [2]. Additionally, PubMed results are abstracts (with links to full texts in PubMed Central (PMC)), which can be of considerable size and do not allow immediate identification of key sentences.

Over the past years, a range of systems have been developed to help users quickly and efficiently search and retrieve relevant publications from the MEDLINE collection indexed by PubMed. This was facilitated and promoted by the free availability of MEDLINE data and Entrez Programming Utilities [38], which make it possible for external entities − from either academia or industry − to create web-based tools that provide alternative ways to search over PubMed.

Lu presented a comprehensive review of tools complementary to PubMed [2], albeit only considering tools that do not search beyond the abstract level, are web based, and capable of searching any arbitrary topic in the biomedical literature as opposed to some limited areas.

Most of the existing information retrieval and extraction tools based on the MEDLINE literature database take advantage of the domain knowledge available in databases and resources such as Entrez Gene, UniProt, Gene Ontology (GO) or Unified Medical Language System (UMLS) to process the titles and abstracts of texts and present the extracted information in different forms, such as relevant sentences describing biological processes, relationships extracted between various biological entities, or in terms of co-occurrence statistics between domain terms. Since our focus is on the development of classification models for ranking of informative sentences in an information retrieval system, we only compared existing tools that return results at the sentence level.

iHOP[1] [39], an acronym for Information Hyperlinked over Proteins, uses genes and proteins as links between sentences, allowing the navigation through sentences and abstracts in PubMed using a network metaphor.

Chilibot[2] [40] retrieves sentences from PubMed abstracts relating to a pair or a list of proteins or genes, and applies shallow parsing to classify these sentences as interactive, non-interactive or simple co-occurrence.

MEDIE[3] [41] uses a deep-parser and a term recognizer to index PubMed abstracts based on pre-computed semantic annotations, allowing for real-time retrieval of sentences containing biological concepts that are related to the user query terms.

Textpresso[4] [42] uses ontologies to allow filtering of results using categories, presenting results as sentences clustered at the abstract level.

BioIE[5] [43] is a rule-based system that uses templates to extract informative sentences from MEDLINE or a user-defined corpus.

Table A.1 compares several aspects of these systems. The characteristics of PhraseRank, the system detailed in the next chapter, are also shown for comparison.

[1]`http://www.ihop-net.org/UniPub/iHOP/`

[2]`http://www.chilibot.net/`

[3]`http://www.nactem.ac.uk/medie/`

[4]`http://www.textpresso.org/neuroscience/`

[5]`http://www.bioinf.man.ac.uk/dbbrowser/bioie/`

Chapter Three

Model Proposal and Implementation

"Knowledge is of two kinds. We know a subject ourselves, or we know where we can find information upon it."

— Samuel Johnson

The previous chapter introduced important topics of research on which this thesis builds upon, related to the fields of text mining, information retrieval and information extraction. The current chapter, comprised of two parts, proposes a model and describes a prototype implementation of a system for fast retrieval of informative sentences from textual documents in the biomedical domain.

The proposed solution is two-fold. First, exploratory work was conducted for the identification of characteristics of an informative sentence – the linguistic and conceptual features that make a given sentence interesting, in the sense that it conveys information. Second, we implemented automated methods to classify sentences as being informative or not, while quantifying the "information content" present in those sentences. Then, using the proposed classification methods as building blocks, we propose a concept-based information retrieval system that, for a given user query, retrieves the most relevant sentences from the literature and presents them in an interactive user interface that facilitates refinement of the retrieved results. Figure 3.1 presents a top-level overview of the proposed system.

For the development, testing and evaluation of the classification methods we used freely available datasets that are adequate for the overall goal of the system. However, the software components developed should be modular enough to be reused on different and larger datasets. The source code of all components is open-source and freely available for non-commercial use. Every component is well tested and the evaluations presented in the following sections are automated and reproducible in a suitable research environment.

The next section presents the proposed sentence classification methods, detailing the model construction, learning resources used, informative features selected, and evaluation results using various strategies and metrics. Section 3.2 details the overall information retrieval system, including the indexing model, ranking algorithms, the search interface and its supporting API.

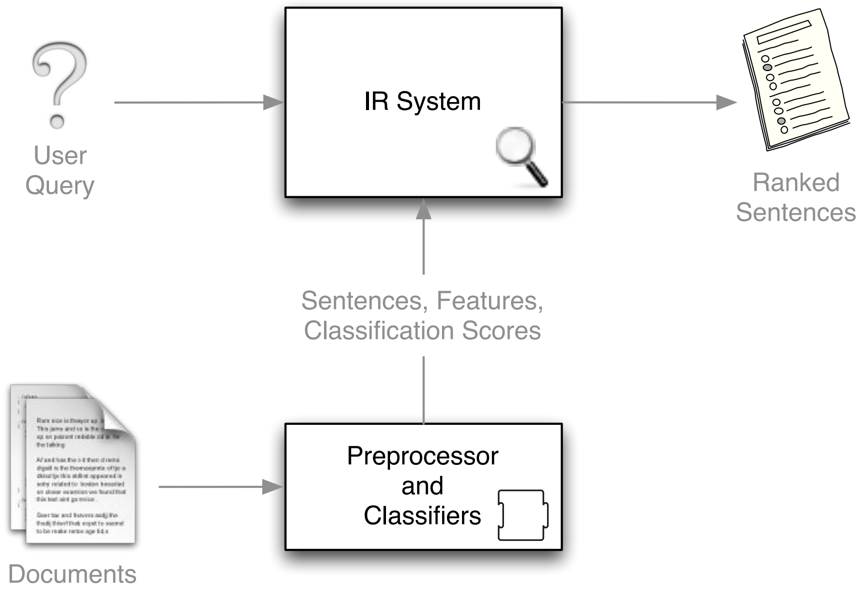

User Query
IR System
Ranked Sentences
Sentences, Features, Classification Scores
Documents
Preprocessor and Classifiers

Figure 3.1: Overview of the proposed system. First, textual documents are preprocessed, features are extracted and sentences are classified and assigned relevance scores. Second, the IR system indexes the sentences along with their features, scores and related metadata. Third, at runtime, users perform queries on the system, while being able to use an assisted query protocol that allows concept-based and free-text queries. Fourth, the system retrieves and ranks the most relevant sentences and presents them to the user, along with hyperlinks to the original documents from where the sentences were extracted. Last, the user can re-rank results by selecting different classification methods and augment the original query with more concepts present in the result sentences.

3.1 Sentence Classification

This section describes the exploratory work that was conducted for the identification and evaluation of features present in informative sentences from biomedical research articles. The use of various learning resources focused on different types of informative sentences allowed us to construct multiple classification models that were later integrated in the information retrieval system described in section 3.2.

3.1.1 Methods

Classifying an arbitrary sentence from a given scientific paper as being informative or not is an abstract task, highly dependent on the information need of the reader. Consider the case of a molecular biologist and a physician reading the same article about the correlation of caffeine consumption and the development of Alzheimer's Disease. While the biologist might be interested in which chemical reactions occur within cells, the metabolic pathways involved and gene expression products, the physician will most likely want to know if caffeine can prevent or cause Alzheimer Disease in patients. The definition of a key sentence in that specific article differs between them and a sentence that one finds informative will not necessarily be relevant to the other.

This subjective notion of what is an informative sentence makes it difficult to develop a general-purpose system, or even a domain-specific one, that correctly selects key sentences from unstructured text. Summarization techniques that rely on linguistic features can be used to construct a semantically correct, human readable summary of an article, but if we consider the seemingly simple task of just classifying a single sentence as being informative or not, there is no one true answer.

Classifiers relying on large sets of complex, handcrafted rules have been shown to yield good results in the identification of key sentences from specific types of text, like web pages, news articles, clinical records, narrative admission reports, and radiology reports [44]–[48]. However, creating such classification systems requires human labour, is extremely time consuming and demands both linguistic and domain-specific knowledge of the text being classified. For this reason, an automatic approach for the identification and extraction of features that make a sentence informative in a given domain presents multiple benefits. First, being an automatic system, once developed it can be easily and quickly applied to different types of informative sentences. Second, it can be constantly improved whenever new, larger sets of annotated data are available.

Results of various community evaluation challenges, such as BioCreative I [8], II [9] and III [49], and the BioNLP shared tasks [24], [25], suggest that supervised systems usually excel over other types of techniques. These systems, relying on supervised machine learning methods, infer a model from training samples, where each sample consists of a collection of features and an output value, the label [34]. Such trained model is then used to classify new unseen instances. In order to train and then apply the model to textual sentences, text has to be pre-processed and predefined features need to be extracted in the same exact manner, initially from the training examples and later from the unseen sentences.

We propose the use of supervised machine learning techniques to automatically generate custom models of informative sentences that reflect certain information needs. Considering the same source documents, we can tag different sets of sentences as being informative or not and train multiple classifiers on this data, in order to produce models that are more finely tuned and appropriate for particular tasks. Recalling the molecular biologist and physician user story mentioned above, the proposed system allows customization of results to satisfy the needs of both users. As long as we have modelled the requirements of these users and trained classifiers using the different concepts of what is a key sentence, we can produce two sets of results that will serve their necessities. This concept is illustrated in Figure 3.2.

Construction of an automatic classification model using supervised machine learning methods is an iterative process that can be continually improved. The diagram in Figure 3.3 describes the methodology we employed. First, original source documents are pre-processed. Second, features are extracted from the pre-processed data. Third, features and labels are used to train a classifier and generate the classification model. Fourth, the classifier is tested on unseen data and classification performance is measured. Fifth, performance is assessed and changes are made to the features used and/or classifier parameters in order to tune the model and improve performance. Last, the model is

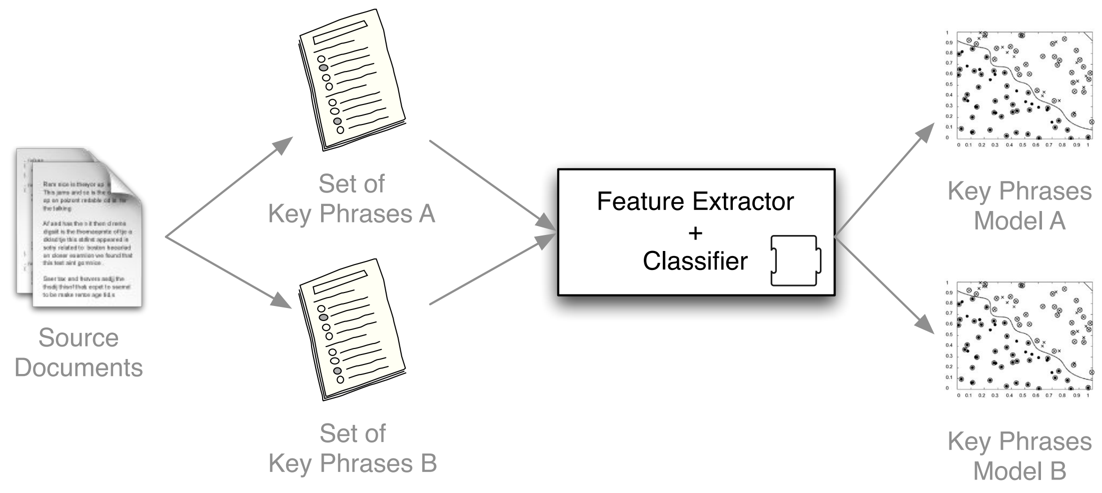


Figure 3.2: Selecting different key sentences from the same source documents allows training of models that identify distinct types of informative sentences.

trained again with the new feature set and parameters to check if results are better. This train-test-tune process is repeated until the best possible performance is achieved.

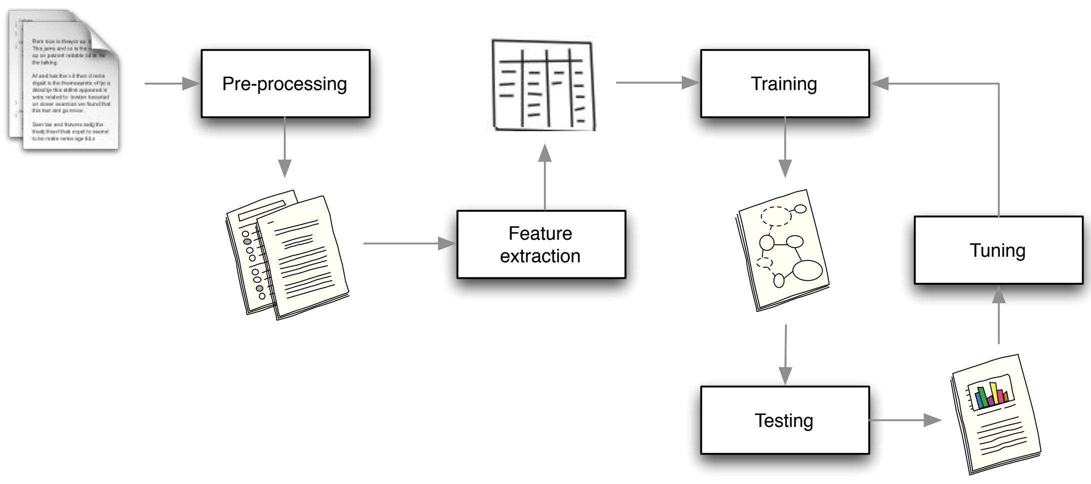


Figure 3.3: Classification model develmopment process.

3.1.1.1 Classification Framework

A plethora of tools for text processing, transformation, parsing, analysis and classification are readily available in various programming languages. For the pre-processing and feature extraction steps we require tools capable of performing tasks such as sentence splitting, tokenization, part-of-speech tagging, dependency parsing and named entity recognition. Neji [50] is a modular framework for biomedical concept recognition developed at the University of Aveiro Bioinformatics Group. It was implemented in Java and it integrates various third-party tools to perform natural language processing tasks. Since it supports all the aforementioned features and is already part of the local text mining framework, it was the obvious choice for integration in our classification pipeline. For the purpose of this thesis, the standard version of Neji was extended and customized to support export of pre-processed documents in multiple formats simultaneously.

In order to perform exploratory work on text classification we could benefit from a complete framework or comprehensive suite of supervised machine learning algorithms. Various software solutions were considered and briefly tested, such as Apache Mahout [1], Orange [2], Weka[3], scikit-learn [4], NLTK[5], Mlpy [6], PyBrain [7], Pyml [8], RapidMiner [9] and Knime [10]. The chosen tool was scikit-learn, an open source machine learning library for the Python programming language, given its flexible, consistent and extendable API for feature extraction, classification, cross-validation and performance evaluation, and its support for a multitude of text classification techniques [51]. Some Natural Language Toolkit (NLTK) [52] modules for parsing of syntactic dependency trees were also extended and used in our custom modules.

The Python programming language is extensively used by the TM community, given its scriptable nature, powerful text manipulation features, ease of use, portability, and broad collection of both standard and third party libraries. The Python shell features a Read-Eval-Print Loop (REPL) that promotes exploratory programming and facilitates experimentation of different techniques to analyse text, extract features, perform classification and then evaluate performance.

To support the classification and indexing methods developed for the purpose of this thesis we built a set of Python modules and scripts that we collectively named PhraseRank (Phrank) . Figure 3.4 represents the overall software package architecture, illustrating dependencies between modules.

The contents and purpose of each package and module are the following:

- **benchmark**: module implementing classification benchmarking methods, facilitating evaluation of multiple classifiers on various datasets using multiple configurable evaluation strategies;
- **bin**: package containing executable scripts for multiple purposes, such as dataset pre-processing, classifier training, classification benchmarking, grid-searching of classifier parameters and sentence indexing;
- **features**: module containing feature extractors;
- **index**: module implementing interfaces for sentence indexing on a Solr server;
- **parse**: module implementing parsing and in-memory representation of standoff annotation files (A1, A2), dependency graphs (CoNLL files) and PubMed XML files enriched with IeXML concept annotations;
- **pipeline**: module containing multiple classification pipelines using various feature extractor combinations;
- **preprocess**: module implementing dataset pre-processing functionalities, such as an interface to Neji and algorithms for informative sentences identification and tagging;
- **resources**: module encapsulating resource management and loading, like datasets and various lexicons;
- **util**: module aggregating diverse utility functions, classes and decorators.

The *phrank* software components are packaged according to Python best practices and can be easily installed and tested along with all their dependencies using standard Python methods.

[1]http://mahout.apache.org/
[2]http://orange.biolab.si/
[3]http://www.cs.waikato.ac.nz/~ml/weka/
[4]http://scikit-learn.org/
[5]http://nltk.org/
[6]http://mlpy.sourceforge.net/
[7]http://pybrain.org/
[8]http://pyml.sourceforge.net/
[9]http://sourceforge.net/projects/rapidminer/
[10]http://www.knime.org/

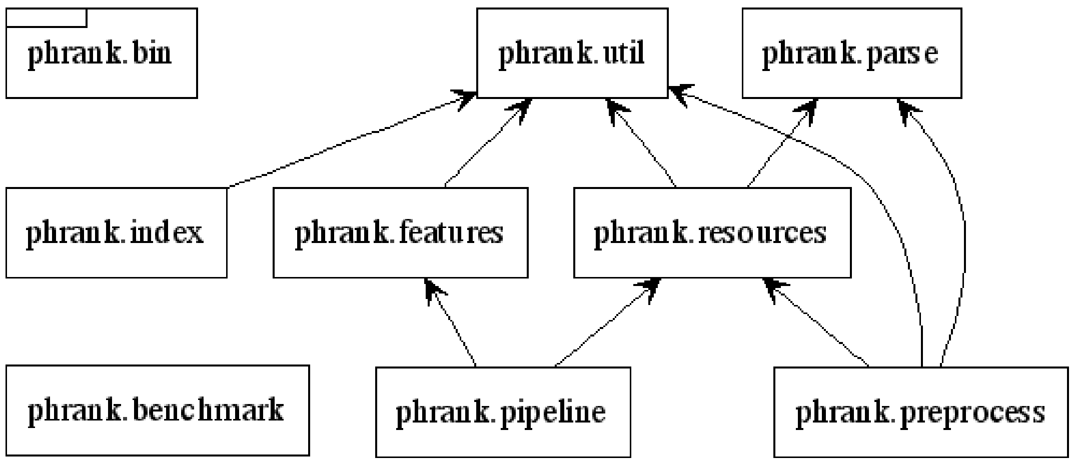


Figure 3.4: Python modules architecture.

3.1.2 Learning Resources

The performance of supervised machine learning techniques is highly dependent on the quality of training data. In order to generate a suitable model that achieves good results we ideally need annotated corpora of superior quality – what is usually called a gold standard. There have been multiple community challenges and shared tasks over the past years focused on text mining problems in the biomedical domain. These initiatives usually result in the development of carefully curated resources that are then publicly released and promote advances in the field. Notable examples include the TREC Genomics track [53], JNLPBA [7], LLL [19], BioCreative [8], [9], [49] and BioNLP [24], [25]. While the first two addressed information retrieval and named entity recognition in the biomedical field, the last three focused on information extraction, specifically bio-events extraction, co-references and relation extraction.

In spite of these community efforts that generated valuable resources, there is still no manually annotated gold standard corpus concerning biomedical facts. We can, however, try to adapt some high quality corpora produced for these tasks and use them for our intended purposes. The main tasks of the BioNLP 2011 Shared Task (BioNLP-ST hereafter), which built upon and extended the tasks from the 2009 edition, focused on event extraction from biomedical texts, extending the annotated resources with full-texts on top of the previously available abstracts. Several bio-molecular events were considered, such as gene expression, transcription, protein catabolism, phosphorylation, localization, binding, positive and negative regulation. These are some of the event types that have been attracting more attention from the research community over the last years. We can assume that a sentence that mentions a bio-event is informative in some way. Furthermore, if the sentence describes the bio-event, stating its arguments and the event trigger, which is often the case, it will most likely be a key sentence in the scientific article.

A series of related bio-events can often describe a high-level biological process (bioprocess hereafter). Bioprocesses occur in living organisms and their regulation is essential in the control of the organisms' life cycles. Wang et al. investigated methods for automatically finding bioprocess terms and events in text and, to facilitate the study, they built a manually annotated gold standard corpus with terms and bio-events related to angiogenesis [54].

This section describes the corpora that we pre-processed and adapted in order to generate training data suitable for the implementation of our proposed classification methods.

3.1.2.1 BioNLP-ST 2011 Datasets

The main tasks of BioNLP-ST 2011 were GENIA (GE), Epigenetics and Post-translational Modifications (EPI) and Infectious Diseases (ID). There was also a bacteria track, with two tasks, Bacteria Biotopes (BB) and Bacteria Interactions (BI); however, the supporting resources for this track focus specifically on bacteria localization events from textbook documents found on relevant public websites and on genetic processes mentioned in scientific texts concerning the bacterium Bacillus subtilis. For this reason, we consider that these corpora from the bacteria track are not appropriate for the development of classification methods that should be general enough to be usable on other types of text.

The GE task focuses on extraction of bio-events related to genes or gene products, without distinguishing between genes and proteins [55]. The dataset supporting this task is based on the publicly available portion of the GENIA corpus [56]. This dataset consists of full-text papers and abstracts in plain text with annotations of named entities and bio-events provided in a standoff format. Figure 3.5 illustrates a sentence annotated with bio-events from the GE task.

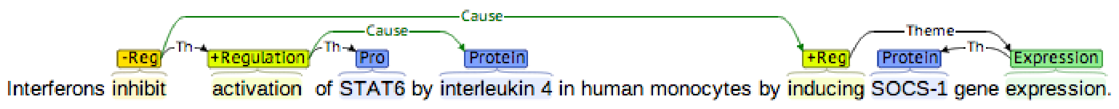

Cause
Cause
Theme
-Reg
Th
+Regulation
Th
Pro
Protein
+Reg
Protein
Th
Expression
Interferons inhibit activation of STAT6 by interleukin 4 in human monocytes by inducing SOCS-1 gene expression.

Figure 3.5: Sentence annotated with bio-events from the GE dataset. Image produced using brat tool [26].

The EPI task is centred on events relating to epigenetic change, including DNA methylation and histone modification, as well as other common post-translational protein modifications [57]. The supporting dataset was built from PubMed abstracts without subdomain restrictions and manually annotated with genes, gene products and 14 different event types, such as protein post-translational modification (PTM) events, DNA methylation events, and their reverse reactions. An annotated sentence from the EPI corpus is shown is Figure 3.6.

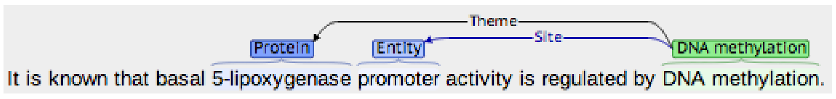
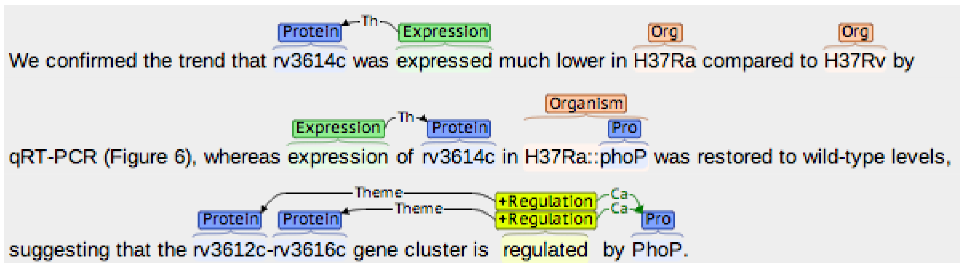

Theme
Site
Protein
Entity
DNA methylation
It is known that basal 5-lipoxygenase promoter activity is regulated by DNA methylation.

Figure 3.6: Sentence annotated with bio-events from the EPI dataset. Image produced using brat tool [26].

The ID task is an event extraction task focusing on the bio-molecular mechanisms of infectious diseases [57]. Domain experts selected representative publications on two-component regulatory systems, a prominent class of signalling system ubiquitous in bacteria. The dataset consists of full-text PMC open access documents. Those publications were manually annotated with genes and gene products, two-component systems, chemicals, organisms, regulons/operons and 10 types of bio-events involved in bioprocesses related to infectious diseases. Figure 3.7 displays a sentence annotated with named entities and bio-events from the ID task.



Protein
Th
Expression
Org
Org
We confirmed the trend that rv3614c was expressed much lower in H37Ra compared to H37Rv by
Organism
Expression
Th
Protein
Pro
qRT-PCR (Figure 6), whereas expression of rv3614c in H37Ra::phoP was restored to wild-type levels,
Theme
Theme
+Regulation
+Regulation
Ca
Ca
Protein
Protein
Pro
suggesting that the rv3612c-rv3616c gene cluster is regulated by PhoP.

Figure 3.7: Sentence annotated with bio-events from the ID dataset. Image produced using brat tool [26].

3.1.2.2 Angiogenesis Corpus

The angiogenesis gold standard corpus was constructed from MEDLINE abstracts randomly selected from a pool of documents mentioning angiogenesis events. Documents were then manually annotated by domain experts with terms and bio-events related to angiogenesis [54]. A sample annotated sentence from this corpus is illustrated in Figure 3.8.

The angiogenesis corpus is freely available in a format similar to the BioNLP-ST 2011 datasets, with abstracts delivered in plain text files and annotations provided separately as standoff files, facilitating its use as another good training and testing resource for the proposed classification methods.

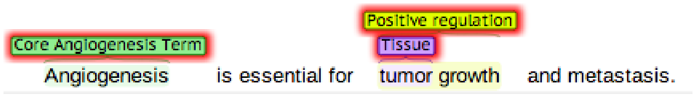
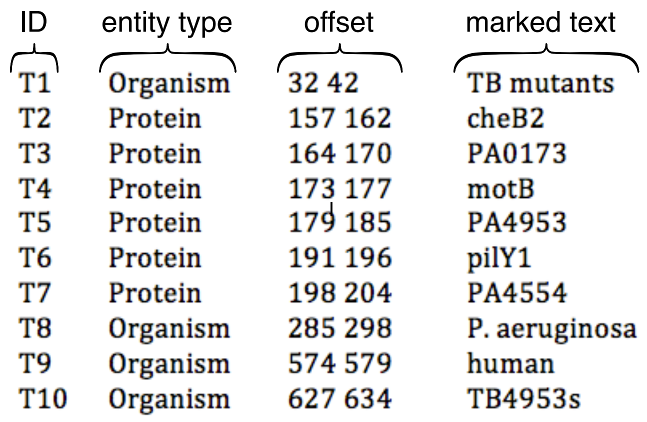


Figure 3.8: Sentence annotated with bio-events from the Angiogenesis dataset. Image produced using brat tool [26].

3.1.2.3 Original Data Format

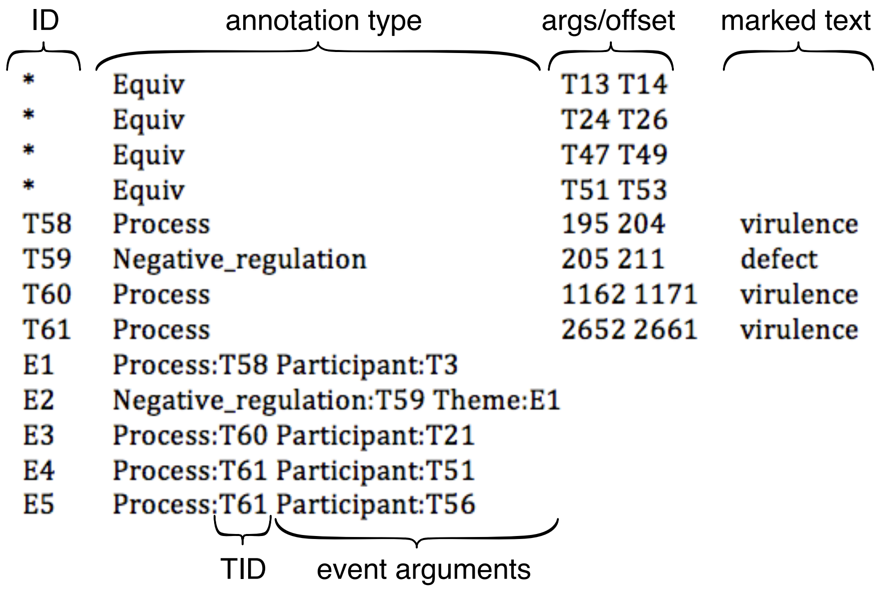
All datasets are provided in plain text files with standoff annotations available in two separate files per document. Named entities are identified in tab-separated A1 files, containing one annotation per line. Each line consists of the entity identifier, entity type, offsets of the entities in the corresponding source text file and the text that was annotated (Figure 3.9). Bio-events annotations are delivered in A2 files, following a tab-separated format similar to the previously mentioned A1. Each line of an A2 file contains the event ID, annotation type and possibly event arguments, offsets and the marked text (Figure 3.10).

For each BioNLP-ST 2001 main task, datasets are distributed in three separate collections aimed at development of automatic event extraction methods, training and then testing. Both the development and training dataset subsets of each task contain named entities and bio-events annotations (A1 and A2 standoff files), yet the test subsets come only with named entities annotations (A1 files), given that the purpose of the tasks was the automatic extraction of bio-events, resulting in the generation of the A2 files. For this reason, we used only the development and training subsets of these datasets in the development of our classification methods.



ID	entity type	offset	marked text
T1	Organism	32 42	TB mutants
T2	Protein	157 162	cheB2
T3	Protein	164 170	PA0173
T4	Protein	173 177	motB
T5	Protein	179 185	PA4953
T6	Protein	191 196	pilY1
T7	Protein	198 204	PA4554
T8	Organism	285 298	P. aeruginosa
T9	Organism	574 579	human
T10	Organism	627 634	TB4953s

Figure 3.9: Illustration of A1 file format for named entities annotations.



ID	annotation type	args/offset	marked text
*	Equiv	T13 T14	
*	Equiv	T24 T26	
*	Equiv	T47 T49	
*	Equiv	T51 T53	
T58	Process	195 204	virulence
T59	Negative_regulation	205 211	defect
T60	Process	1162 1171	virulence
T61	Process	2652 2661	virulence
E1	Process:T58 Participant:T3		
E2	Negative_regulation:T59 Theme:E1		
E3	Process:T60 Participant:T21		
E4	Process:T61 Participant:T51		
E5	Process:T61 Participant:T56		

Figure 3.10: Illustration of A2 file format for bio-events annotations.

3.1.2.4 Data Preparation

The datasets presented above are rich collections of scientific articles and abstracts, manually curated by domain experts with gold standard annotations of biologically relevant named entities, bio-events and bioprocesses. Potential key sentences in those texts are not explicitly marked as being informative, however, considering that a sentence describing any bio-event conveys information, we can process these resources and automatically tag sentences mentioning bio-events as being informative. This technique allows us to produce datasets usable for training of supervised learning methods.

While we can safely assume that, using this method, all sentences tagged as being informative will in fact be informative in some way, it is possible that some of the sentences where no bio-events are mentioned also express some type of information.

The four datasets were pre-processed using Neji [50]. Results of sentence splitting and concept identification were exported and saved to JSON [58] files, while part-of-speech tagging and dependency parsing data were stored in CoNLL [59] files.

After applying sentence splitting to extract individual sentences from the 3,256 abstracts and 87 full-text documents, a total of 25,235 sentences were identified. Table 3.1 contains statistics about each separate dataset. Documents were then processed using the algorithm described in Listing 1 in order to identify and tag informative sentences. The simple algorithm considers a sentence informative if it mentions a bio-event and contains all its arguments, which usually include the trigger and any related concepts. If an event is scattered through more than one sentence then it is not considered for tagging. Sentence labels were saved to standoff tab-separated files containing a sentence identifier and the label (0 or 1) per line.

Pre-processing and identification of informative sentences where bio-events are mentioned resulted in the tagging of 8,665 sentences as being informative, corresponding to 34.34% of the complete sentence collection. Detailed statistics about the informative sentences marked in each dataset are presented in Table 3.2.

Even though the labelled datasets collection generated from the aforementioned corpora can not be considered of gold standard quality, given that it was produced automatically and no manual curation by domain experts was performed, it is the best resource available for the proposed task and should aid in the development of classification methods that can later be trained on gold standard datasets, when they are available. The next section describes the features that were extracted from these "silver standard" datasets to build a classification model for informative sentences in the biomedical domain.

Table 3.1: Datasets statistics after sentence splitting.

Dataset	**Full-texts**	**Abstracts**	**Sentences**
GE	10	950	11579
EPI	0	800	7688
ID	77	0	3184
Angiogenesis	0	280	2784
Total	**87**	**3256**	**25235**

```
for document in dataset:
    for event in document.annotated_events:
        # find the sentence where the event was annotated
        sentence = get_sentence_annotated_with(event, document)

        # check if all event arguments (trigger, related concepts,
        #  etc.) are mentioned in the same sentence
        try:
            for event_argument in event.arguments:
                if not is_mentioned_in(event_argument, sentence):
                    raise EventSpreadThroughMultipleSentences
            # tag this sentence as informative because it mentions
            #  a bio-event
            sentence.informative = True
        except EventSpreadThroughMultipleSentences:
            # skip and ignore this event, since it is spread
            #  across multiple sentences
            pass
```

Listing 1: Algorithm used to identify and tag informative sentences.

Table 3.2: Datasets statistics after pre-processing and tagging of informative sentences.

Dataset	Sentences	Informative		Non-Informative	
GE	11579	4785	41.32%	6794	58.68%
EPI	7688	1394	18.13%	6294	81.87%
ID	3184	1226	38.51%	1958	61.49%
Angiogenesis	2784	1260	45.30%	1524	54.70%
Total	**25235**	**8665**	**34.34%**	**16570**	**65.66%**

3.1.3 Features

Language models for text classification can be based on various types of features. Typical language features can range from lexical to syntactic, grammatical, and semantic characteristics of text. The nature of each feature allows us to characterize natural language at different levels and model the information contained in a piece of text, extracting its meaning.

In an attempt to identify linguistic characteristics of informative sentences, multiple features were extracted and tested, both individually and in various combinations, in order to assess the ones that worked best for our specific test cases. This section describes the features that were considered, and results are analysed and discussed in section 3.1.6.

3.1.3.1 Bags-of-words

One of the most simple and commonly used models for text classification tasks is the bag-of-words. It is a simplifying representation of text in which a sentence or a document is represented as an unordered collection of words, disregarding grammar and word order [27]. Binary bags-of-words simply account for the presence of words in text. Frequency based models count the occurrences of each word in each document (a sentence, in our case) and use those as features for a classifier.

In the feature extraction step, words are isolated and a dictionary is constructed mapping each unique word to an integer value. Each sentence is then represented as a vector with size equal to the dictionary size, where each position of the vector contains the frequency of the word associated with that position.

Consider the following two sentences as documents of an example corpus:

- *"Alzheimer disease is the most common type of dementia."*
- *"Most studies exclude the undiagnosed dementia sufferers in the population."*

A basic bag-of-words model of our corpus would construct a dictionary with 15 unique words (Listing 2) and represent the sentences as the 15-entry vectors illustrated in Listing 3. We can then improve this basic model by removing words that occur in too many documents (stop words[11]) and words that occur only in a few documents, since they don't aid in the classification process. We used the recommended list of stop words from PubMed [60] and also filtered words occurring in only one sentence.

Bag-of-words models allow capturing specific vocabulary commonly used in certain types of sentences. However, it has the drawback of being easy to over-fit a model to a particular lexicon if training data centres around certain type of language.

```
{
    'alzheimer': 0,
    'common': 1,
    'dementia': 2,
    'disease': 3,
    'exclude': 4,
    'in': 5,
    'is': 6,
    'most': 7,
    'of': 8,
    'population': 9,
    'studies': 10,
    'sufferers': 11,
    'the': 12,
    'type': 13,
    'undiagnosed': 14
}
```

Listing 2: Bag-of-words dictionary.

```
[1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0]
[0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 2, 0, 1]
```

Listing 3: Bag-of-words sentences representation.

3.1.3.2 N-grams

N-grams can be seen as an extension of the simple bag-of-words model. An n-gram is a contiguous sequence of n items from a given sequence of text. The items can be phonemes, syllables, letters or words, depending on the purpose and application. These types of models are extensively used in statistical natural language processing tasks, given their ability to easily model and predict, to some extent, natural language construction [27].

[11]Stop words are very common words that occur in many documents and are not relevant for document discrimination, so we filter them prior to further text processing.

Recalling the example corpus of two sentences mentioned in section 3.1.3.1, an n-gram model considering word n-grams from single words (n = 1) to trigrams (n = 3) would build a dictionary similar to the one presented in Listing 2 containing also bigrams and trigrams, as illustrated in Table 3.3.

Choosing the right n for the n-gram is one of the main challenges of using this type of model, since this controls the trade-off between appropriateness of the model and the possibility of bias against a specific corpus (over-fitting the model). We evaluated n-grams from single words (merely bag-of-words) to 4-grams and found trigrams to be the best option for our data. Detailed results are presented in section 3.1.6.2.

Table 3.3: N-grams vocabulary model

1-grams (words)	**2-grams (bigrams)**	**3-grams (trigrams)**
alzheimer	alzheimer disease	alzheimer disease is
common	common type	common type of
dementia	dementia sufferers	dementia sufferers in
disease	disease is	disease is the
exclude	exclude the	exclude the undiagnosed
in	in the	in the population
is	is the	is the most
most	most common; most studies	most common type; most studies exclude
of	of dementia	-
population	-	-
studies	studies exclude	studies exclude the
sufferers	sufferers in	sufferers in the
the	the most; the population; the undiagnosed	the most common; the undiagnosed dementia
type	type of	type of dementia
undiagnosed	undiagnosed dementia	undiagnosed dementia sufferers

3.1.3.3 TF-IDF Vectors of N-grams

Term Frequency-Inverse Document Frequency (TF-IDF) is a measure that reflects how important a word or n-gram is to a document in a collection or corpus and is frequently used as a weighting factor in information retrieval and text mining applications [61]. The TF-IDF value increases proportionally to the number of times a term (a word or n-gram) appears in the document, but is offset by the frequency of the term in the corpus, which helps to control for the fact that some words are generally more common than others.

The term frequency value (TF) is the number of times a word (or n-gram) appears in a document (a sentence in our case). The inverse document frequency (IDF) is calculated by dividing the total number of documents by the number of document containing the term and then taking the logarithm of that quotient. The resulting TF-IDF value can then be scaled and/or normalized using various strategies. We tested multiple combinations of different methods of calculating TF-IDF, such as no normalization, L1 (manhattan distance) and L2 (euclidean distance) norms, simple TF versus sub-linear TF[12], and found sub-linear TF plus L2 norms to yield the best results for trigrams.

3.1.3.4 TF-IDF Vectors of POS tags

Part-of-speech tagging (POS tagging), also known as grammatical tagging or word-category disambiguation, is the process of labelling a word as corresponding to a particular part of speech, considering both its definition and the grammatical context in which it appears. It can be simply seen as the identification of words as nouns, verbs, adjectives, adverbs, pronouns, and so on.

In addition to TF-IDF vectors of n-grams we also tested using POS tags instead of words as features for informative sentences classification.

[12]Sub-linear TF is a scaling technique that replaces the raw TF value with $1 + \log(TF)$.

3.1.3.5 TF-IDF Vectors of POS, Chunk tags Pairs

Chunking, also called shallow parsing, performs an analysis of a sentence with the intent of identifying its constituent parts, such as noun groups, verbs, verb groups, and other grammatical structures without specifying the overall internal sentence structure, nor the role of the parts in the main sentence [62].

By pairing POS and chunk tags for each sentence token, we hope to be able to model the grammatical structure of a sentence without tying the model to specific lexical characteristics of the sentence. We tested the use of TF-IDF vectors of POS, chunk tags n-grams for sentence classification.

3.1.3.6 Biologically Relevant Verbs

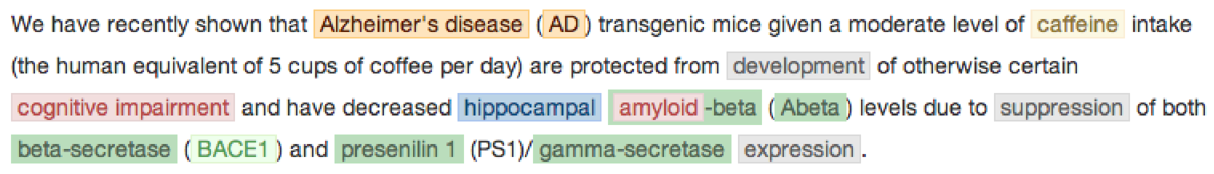
Sentences that describe some type of biological interaction or event often use a specific trigger verb to connect biological concepts. For this reason, the presence of biologically relevant verbs in a sentence can indicate a sentence conveys important information. BioLexicon is a terminology resource tailored for the biological domain that contains, among other terms, a set of terminological verbs and their derived forms [63]. We used 3,044 verbs and derivations from this lexicon as features to identify mentions of biologically relevant verbs in sentences. Each feature considered is binary, representing the occurrence of a given verb in a sentence.

3.1.3.7 Biological Concepts

An informative sentence always describes, mentions, defines or explains a given biological concept or the interaction between various concepts. Hence, the mere presence of a relevant named entity in a sentence can help assert the informative value of a sentence.

As previously mentioned, we pre-processed all datasets using Neji, which provides automatic identification of biological named entities that are used as features for the classification model. Each recognized entity is labelled with its corresponding entity type. Figure 3.11 illustrates a sentence annotated by Neji with multiple concepts.

Even though we could have used the unique identifiers of each concept as features, that would most likely bias the model towards the identification of informative sentences mentioning only the concepts present in the training datasets. For this reason, we chose to use three different types of binary features – a measure of the number of concepts mentioned in a sentence, a measure representing the different types of concepts present in a sentence and the presence of concepts of a given type. Table 3.4 describes the binary features used in the model construction.



We have recently shown that Alzheimer's disease (AD) transgenic mice given a moderate level of caffeine intake (the human equivalent of 5 cups of coffee per day) are protected from development of otherwise certain cognitive impairment and have decreased hippocampal amyloid -beta (Abeta) levels due to suppression of both beta-secretase (BACE1) and presenilin 1 (PS1)/ gamma-secretase expression .

Figure 3.11: Sentence annotated with biological concepts.

Table 3.4: Biological concepts features used for sentence classification

Feature	**Description**
concept_count(1)	Sentence mentions one biological concept
concept_count(2)	Sentence mentions two biological concepts
concept_count(3)	Sentence mentions three biological concepts
concept_count(4+)	Sentence mentions four or more biological concepts
concept_types_count(1)	Sentence mentions only one type of biological concepts
concept_types_count(2)	Sentence mentions two types of different biological concepts
concept_types_count(3)	Sentence mentions three types of different biological concepts
concept_types_count(4+)	Sentence mentions four or more types of different biological concepts
has_concept(SPEC)	Sentence mentions at least one species
has_concept(ANAT)	Sentence mentions at least one anatomy part
has_concept(DISO)	Sentence mentions at least one disorder
has_concept(PATH)	Sentence mentions at least one metabolic pathway
has_concept(CHED)	Sentence mentions at least one chemical or drug
has_concept(ENZY)	Sentence mentions at least one enzyme
has_concept(MRNA)	Sentence mentions at least one microRNA entity
has_concept(PRGE)	Sentence mentions at least one gene or protein
has_concept(COMP)	Sentence mentions at least one cellular component
has_concept(FUNC)	Sentence mentions at least one molecular function
has_concept(PROC)	Sentence mentions at least one biological process

3.1.3.8 Syntactic Arcs

Sentences from the development datasets were pre-processed using Neji in order to generate dependency graphs representing syntactic dependencies between terms. Neji uses an external tool to perform dependency parsing, a customized version of GDep, which is a dependency parser for biomedical text based on LR models and parser ensembles [64].

There are a multitude of features from dependency parses we could consider for inclusion in our model. For the purpose of this thesis, we experimented with using the number of arcs of a certain distance (in number of hops) between recognized concepts and biologically relevant verbs.

Figure 3.12 shows a graphical representation of a syntactic dependency tree generated by GDep for the sentence *"Promoter methylation of CpG target sites or direct deletions/insertions of genes are mechanisms of a reversible or permanent silencing of gene expression, respectively."*. *"Methylation"* was annotated by Neji as a biological process and *"silencing"* is a biologically relevant verb present in BioLexicon. For this sentence, the arc of 4 hops distance between *"methylation"* and *"silencing"* would be considered a discrete feature named *syntactic_arcs*(4) with value equal to 1.

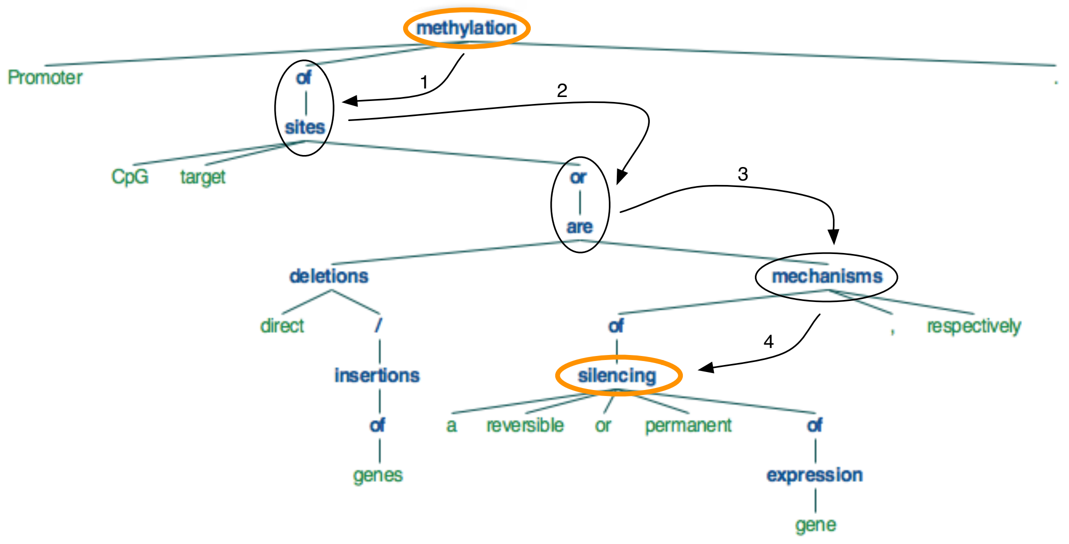


Figure 3.12: Syntactic dependency tree illustrating number of hops between recognized concept and biologically relevant verb.

3.1.3.9 Feature Representation

Supervised classifiers are mathematical instruments based on probabilistic models that analyse input values in order to produce an output value – the class that best represents the input data. Therefore, we need to represent our sentences as vectors of feature values in which classification algorithms can operate. The length of each vector is equal to the total number of features considered by the classification model and the absence of a feature from a sentence is represented by the value 0 in that specific vector position. Depending on the type of each feature, its value can be binary (0 or 1), integer or real.

Considering the case of using the four aggregated datasets as training data and a set of features before applying any feature selection consisting of TF-IDF vectors of n-grams (1-3), TF-IDF vectors of POS, Chunk n-gram tags pairs (1-3), biologically relevant verbs, biological concepts and syntactic arcs, the input data for our learning functions would be a matrix of approximately 25,000 rows (samples) by 120,000 columns (features), resulting in a total of 3×10^9 elements. In Python, a double precision floating point number occupies 8 bytes of memory, so holding a standard matrix of 3×10^9 floating point numbers in memory using a typical representation would require more than 22 gigabytes of RAM, more than what it usually available in modern personal computers.

Given the iterative and repetitive nature of the exploratory process of experimenting with different feature combinations, normalization techniques, classifier regulation parameters and evaluation strategies, demanding the load of such amount of data to memory on a regular basis can affect development agility. Hence, less resource intensive data representations present various advantages when working with large datasets. Matrices can be represented using sparse data structures that are less memory hungry, due to the fact that only values different from zero are stored, along with metadata used to identify their positions.

NumPy [13] is a Python library providing multi-dimensional array objects with sparse implementations that are ideal for highly dimensional feature matrices representation. We used these data structures to represent our feature matrices and tackle the memory issues at the cost of not being able to use some implementation of feature selection, normalization and classification techniques that require dense data representations.

[13]`http://www.numpy.org/`

3.1.3.10 Feature Standardization

The range of raw feature values collected after the feature extraction step can vary widely, depending on the datasets and features considered for classification. If one of the features has a broad range of values, this particular feature can dominate the feature space, which can cause a classifier to be unable to correctly learn from other relevant features, hence negatively affecting classification performance. Consequently, if can be beneficial to normalize the range of all features so that each feature contributes correctly classification.

Standardization of a dataset is a common necessity for some ML algorithms, since they might behave badly if individual features do not look more or less like standard normally distributed data. Two common operations usually performed on feature values prior to classification model induction are scaling and normalization. Scaling is the process of standardizing features (matrix columns) by removing the mean (centering) and scaling to unit variance. Normalization deals with rescaling each training sample (a row of the data matrix) independently of other samples so that its norm (e.g. L1 or L2) equals one.

We compared the use of raw feature values versus unit variance scaled and L2-norm normalized data for evaluating the performance of various classifiers. Results are presented in section 3.1.5.

3.1.3.11 Feature Selection

Feature Selection (FS) is the process of reducing the dimensionality of feature sets by selecting a subset of relevant features for use in model construction. This can both decrease the time it takes to train and apply a classifier, and, in some cases, improve the classification accuracy. Common FS techniques include univariate feature selection, multivariate feature selection, Recursive Feature Elimination (RFE) , tree-based FS, Mutual Information (MI) and Information Gain (IG). For TF-IDF feature vectors, minimum and maximum DF thresholds, and limiting the number of top terms used can also be applied for reducing vocabulary size [65].

Univariate feature selection works by selecting the best features based on univariate statistical tests. A commonly used technique for univariate feature selection is the chi-square (X^2) test [66]. Statistics from chi-square tests for each feature/class combination can then be used to select the features with higher test scores. We tried to use chi-square based FS to reduce the dimensionality of our datasets, but all attempts resulted in significantly reduced classification scores, leading us to abandon this approach.

Given a classifier that assigns weights to features (e.g., the coefficients of a linear model), RFE selects features by recursively considering smaller sets of features. Initially the classifier is trained on the complete set of features and weights are assigned to each one of them. Then, features with the smallest absolute weights are pruned from the current feature set. This procedure is recursively repeated on the pruned set until the desired number of features is eventually reached. We tried using a linear Supervised Vector Machine (SVM) on a cross-validation pipeline for RFE, however, we believe that the high number of features of our data caused this method to be inefficient for successful feature elimination.

Tree-based classifier ensembles, also called randomized forests can also be used for feature selection because decision tree classifiers calculate feature importance values. Unfortunately, the implementation of decision trees of sklearn requires dense data (demanding huge amounts of memory) to construct a model, making them unusable in our tests.

Minimum and maximum document frequency thresholds combined with limiting the maximum number of features used in TF-IDF vectors were the only FS techniques that allowed us to considerably reduce the dimensionality of our sample sets without significant performance losses. We performed a grid-search to find the best combination of parameters for the TF-IDF feature extractor and found the best compromise between classification performance and feature vectors dimensionality reduction to be when using the top 25,000 n-grams after using a minimum DF threshold of 2 sentences and a maximum threshold of 90%, meaning that only terms that occur in at least 2 sentences and at most in 90% of training instances are considered.

3.1.4 Evaluation Strategy

Various methods have been proposed and widely used for the evaluation of classifiers, such as the holdout method, k-fold cross-validation, leave-one-out and bootstrap.

The holdout method, also called test sample estimation, partitions the data randomly into two mutually exclusive subsets called a training set and a test set (or holdout set, hence the name). It is typical to choose 2/3 of the data as the training set and the remaining 1/3 as the test set. A classification model is then induced on the training set data and the generated classifier is tested on the test set. The holdout method is a pessimistic estimator due to the fact that only a fraction of the data is given to the model inducer for training. A common variation of the holdout strategy is random subsampling, where the holdout method is repeated k times and accuracy is then averaged from all the accuracies obtained.

In k-fold cross-validation, sometimes called rotation estimation, the complete dataset is randomly split into k mutually exclusive subsets of approximately equal size, the folds. The classification model is then trained k times, each time on $k - 1$ folds and tested on the other. The performance measure reported by k-fold cross-validation is then the average of the measures calculated for each individual test. In stratified k-fold cross-validation the folds are stratified so that they contain approximately same proportion of entities in each class as in the complete dataset. Previous studies have show that stratification is generally a better scheme both in terms of bias and variance when compared to regular cross-validation [67].

The leave-one-out method is a specific type of cross-validation where k is equal to the number of samples in the dataset. The model is then induced on all samples except one, which is used for testing. This method is commonly used on the presence of very small datasets, however, while leave-one-out strategies are almost unbiased, they present high variance that can lead to unreliable estimates [68].

Bootstrapping is a general statistical technique that iterates the computation of an estimator on a resampled dataset. Given a dataset of size n, a bootstrap sample is created by sampling n instances uniformly from the data (with replacement) [69]. Contrary to other cross-validation strategies, bootstrapping will allow some samples to occur several times in each data split.

Stratified k-fold cross-validation with $k = 10$ has been shown to be one of the most reliable classifier evaluation strategies, usually yielding low variation and bias [67]. For this reason, we chose to use this type of cross-validation to evaluate the performance of our classification methods on our complete dataset, except for the initial comparison of multiple classification algorithms, in which we used stratified k-fold cross-validation with $k = 5$, to reduce the time required to test the various classifiers.

Since our complete aggregated dataset is composed by four datasets containing different types of informative sentences, we also performed stratified k-fold cross-validation ($k = 10$) individually on each of the datasets. These additional evaluations give us insight into the appropriateness of our classification methods on more specific types of text and can help us identify types of features that do not yield good results for certain types of data.

Even though we are evaluating the performance of our classification model using our development datasets, the developed methods should be general enough to work on different types of biomedical sentences. One way to test if our classifier can produce good results on different types of sentences is to train it on datasets containing types of sentences (and possibly features) different from the ones used to test its performance. This strategy can be described as a type of leave-one-out cross-validation, where instead of leaving one sample out, we leave a whole dataset out in each of the evaluations and then average results across all runs.

The results presented in the next sections are labelled with the type of evaluation that was performed on the induced classifier and should be interpreted as following:

- **CV[BioNLP+Angio]**: Stratified 10-fold cross-validation on the complete aggregated dataset, composed by the three BioNLP datasets (GE, EPI and ID) and the Angiogenesis dataset;
- **CV[GE]**: Stratified 10-fold cross-validation on the BioNLP GE dataset;
- **CV[EPI]**: Stratified 10-fold cross-validation on the BioNLP EPI dataset;
- **CV[ID]**: Stratified 10-fold cross-validation on the BioNLP ID dataset;
- **CV[Angio]**: Stratified 10-fold cross-validation on the Angiogenesis dataset;
- **TT[BioNLP/Angio]**: Classification model trained on the three BioNLP datasets (GE, EPI and ID) and tested on the Angiogenesis dataset;

- **TT[GE+EPI+Angio/ID]**: Classification model trained on the GE, EPI and Angiogenesis datasets and tested on the ID dataset;
- **TT[GE+ID+Angio/EPI]**: Classification model trained on the GE, ID and Angiogenesis datasets and tested on the EPI dataset;
- **TT[EPI+ID+Angio/GE]**: Classification model trained on the EPI, ID and Angiogenesis datasets and tested on the GE dataset;
- **TT[Average]**: Average of the four previous evaluations (TT[BioNLP/Angio], TT[GE+EPI+Angio/ID], TT[GE+ID+Angio/EPI] and TT[EPI+ID+Angio/GE]).

For each evaluation run we calculated multiple performance metrics, such as the F1-measure (F-score), precision (Prec.), recall, average precision (Av.Pr.), the Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) and accuracy.

3.1.5 Classification Algorithms Evaluation

Estimating the performance of a classifier induced by supervised learning algorithms is important not only to predict its future accuracy but also for choosing a classification algorithm from a given set (model selection) or for combining various classifiers [70]. Various supervised learning algorithms are commonly used for text classification tasks with different degrees of success, but no single algorithm can uniformly outperform other algorithms over all datasets.

The nature of the text being classified, the features extracted and the purpose of the classification application influence the classifier choice and only practical tests and evaluation of various algorithms on the feature set that induces the prediction model can guide the choice of the best algorithm for a given problem. The key question when dealing with ML classification is not whether a learning algorithm is superior to others, but under which conditions a particular method can significantly outperform others on a given application problem [36]. When faced with the question of which learning algorithm will perform best on our classification problem, the simplest approach is to estimate the accuracy of several candidate algorithms on the problem and select the one that appears to be most accurate.

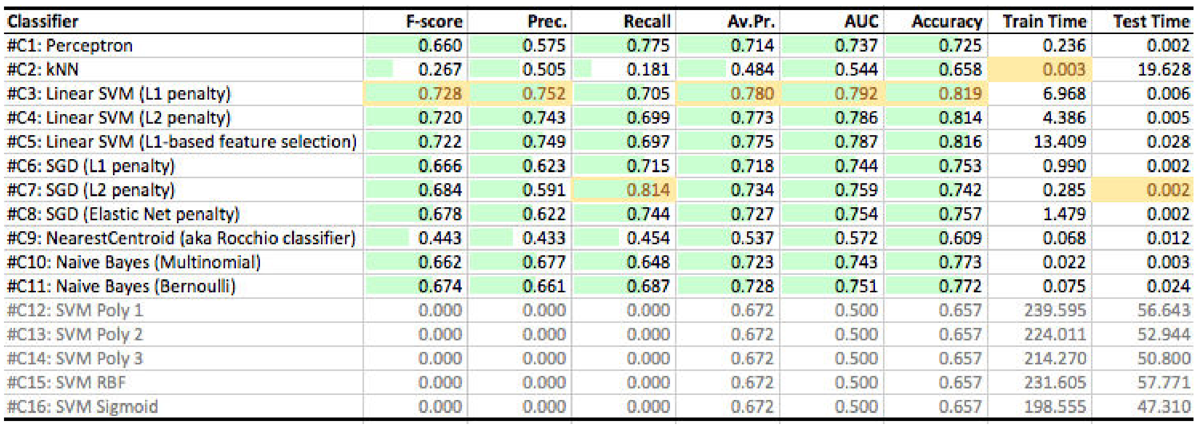
We selected various candidate algorithms for our classification task, which were all implemented by our machine learning framework of choice, scikit-learn. Since classifier parameters affect the behaviour of the model inducer, we also experimented with a few parameter variations, such as regularization parameters of linear models and kernel choices for SVMs. Table B.1 enumerates the classifiers we tested and the chosen configuration parameters. It should be noted that even though decision tree based classifiers are often used for text classification tasks, we were not able to apply those to our problem due to the use of sparse matrices for feature representation and the requirement of dense data by the scikit-learn implementations of decision tree model inducers.

An initial baseline feature set consisting of a simple bag-of-words model of our data was established and was iteratively extended with more features while evaluating the impact of each additional feature on the performance of 16 candidate classifiers listed in Table B.1. In this section we present final results for the evaluation of those classifiers on the extended feature set that we selected for our classification task. Results for the evaluation of the elected classifier using multiple validation strategies on various subsets of features are later presented in section 3.1.6.

The 16 classifiers were tested on the extended feature set of TF-IDF vectors of 1, 2, and 3-grams TF-IDF vectors of POS tags trigrams, TF-IDF vectors of POS-Chunk tag pairs trigrams, biologically relevant verbs, biological concepts and syntactic arcs after applying feature selection through TF-IDF thresholds. Stratified 5-fold cross-validation on the complete dataset (BioNLP+Angio) was used for the evaluation, which resulted in training sets of 20,193 sentences and testing sets of 5,049 sentences. The feature set consisted of approximately 44,000 features per evaluation run.

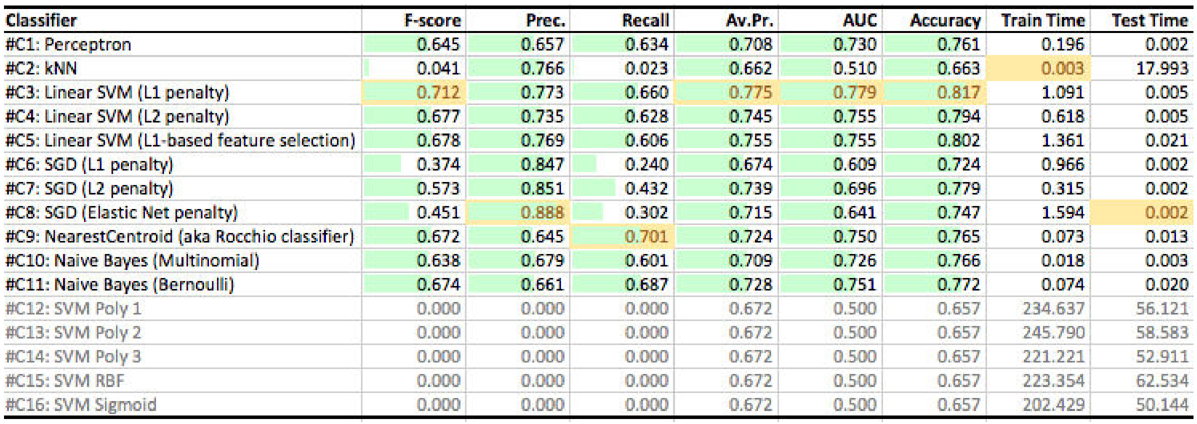
We first induced classification models from the raw feature values and then tested inducing the same models after applying scaling and normalization, since some models based on distances are quite sensitive to feature range variations. Table 3.5 presents results for evaluation using raw feature values and results for scaled and normalized feature values are shown in Table 3.6. The last two columns of each table present training and testing times averages for each evaluation run, in seconds. The best value achieved for each performance metric is highlighted in yellow.

Table 3.5: Classifiers comparison using large feature set with raw feature values using CV[BioNLP+Angio] evaluation.



Classifier	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy	Train Time	Test Time
#C1: Perceptron	0.660	0.575	0.775	0.714	0.737	0.725	0.236	0.002
#C2: kNN	0.267	0.505	0.181	0.484	0.544	0.658	0.003	19.628
#C3: Linear SVM (L1 penalty)	0.728	0.752	0.705	0.780	0.792	0.819	6.968	0.006
#C4: Linear SVM (L2 penalty)	0.720	0.743	0.699	0.773	0.786	0.814	4.386	0.005
#C5: Linear SVM (L1-based feature selection)	0.722	0.749	0.697	0.775	0.787	0.816	13.409	0.028
#C6: SGD (L1 penalty)	0.666	0.623	0.715	0.718	0.744	0.753	0.990	0.002
#C7: SGD (L2 penalty)	0.684	0.591	0.814	0.734	0.759	0.742	0.285	0.002
#C8: SGD (Elastic Net penalty)	0.678	0.622	0.744	0.727	0.754	0.757	1.479	0.002
#C9: NearestCentroid (aka Rocchio classifier)	0.443	0.433	0.454	0.537	0.572	0.609	0.068	0.012
#C10: Naive Bayes (Multinomial)	0.662	0.677	0.648	0.723	0.743	0.773	0.022	0.003
#C11: Naive Bayes (Bernoulli)	0.674	0.661	0.687	0.728	0.751	0.772	0.075	0.024
#C12: SVM Poly 1	0.000	0.000	0.000	0.672	0.500	0.657	239.595	56.643
#C13: SVM Poly 2	0.000	0.000	0.000	0.672	0.500	0.657	224.011	52.944
#C14: SVM Poly 3	0.000	0.000	0.000	0.672	0.500	0.657	214.270	50.800
#C15: SVM RBF	0.000	0.000	0.000	0.672	0.500	0.657	231.605	57.771
#C16: SVM Sigmoid	0.000	0.000	0.000	0.672	0.500	0.657	198.555	47.310

Table 3.6: Classifiers comparison using large feature set with scaled and normalized feature values using CV[BioNLP+Angio] evaluation.



Classifier	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy	Train Time	Test Time
#C1: Perceptron	0.645	0.657	0.634	0.708	0.730	0.761	0.196	0.002
#C2: kNN	0.041	0.766	0.023	0.662	0.510	0.663	0.003	17.993
#C3: Linear SVM (L1 penalty)	0.712	0.773	0.660	0.775	0.779	0.817	1.091	0.005
#C4: Linear SVM (L2 penalty)	0.677	0.735	0.628	0.745	0.755	0.794	0.618	0.005
#C5: Linear SVM (L1-based feature selection)	0.678	0.769	0.606	0.755	0.755	0.802	1.361	0.021
#C6: SGD (L1 penalty)	0.374	0.847	0.240	0.674	0.609	0.724	0.966	0.002
#C7: SGD (L2 penalty)	0.573	0.851	0.432	0.739	0.696	0.779	0.315	0.002
#C8: SGD (Elastic Net penalty)	0.451	0.888	0.302	0.715	0.641	0.747	1.594	0.002
#C9: NearestCentroid (aka Rocchio classifier)	0.672	0.645	0.701	0.724	0.750	0.765	0.073	0.013
#C10: Naive Bayes (Multinomial)	0.638	0.679	0.601	0.709	0.726	0.766	0.018	0.003
#C11: Naive Bayes (Bernoulli)	0.674	0.661	0.687	0.728	0.751	0.772	0.074	0.020
#C12: SVM Poly 1	0.000	0.000	0.000	0.672	0.500	0.657	234.637	56.121
#C13: SVM Poly 2	0.000	0.000	0.000	0.672	0.500	0.657	245.790	58.583
#C14: SVM Poly 3	0.000	0.000	0.000	0.672	0.500	0.657	221.221	52.911
#C15: SVM RBF	0.000	0.000	0.000	0.672	0.500	0.657	223.354	62.534
#C16: SVM Sigmoid	0.000	0.000	0.000	0.672	0.500	0.657	202.429	50.144

Five of the classifiers tested – all the SVMs with non-linear kernels (#C12-16) – tried to separate the feature space and induce a classification model for considerable longer time than the other linear models, but apparently failed to do so. Analysis of confusion matrices for those cases revealed that the generated models classified all instances as non-informative. This resulted in zero precision, recall and consequently f-measure scores. The remaining performance metrics are then irrelevant and their values simply reflect the ratio of non-informative sentences in the testing sets.

It is noted in the scikit-learn documentation that when the number of features is much greater than the number of samples, SVMs with non-linear kernels are likely to give poor performances[14]. Their implementation of non-linear kernels is based on libsvm[15], hence the fit time complexity is more than quadratic with the number of samples, making it hard to scale for datasets with more than a couple of 10,000 samples, which is the case of our data. Linear kernel implementations in the scikit are based on liblinear[16], a much more efficient implementation than libsvm that can scale almost linearly to millions of samples and/or features[17].

[14]`http://scikit-learn.org/0.12/modules/svm.html#support-vector-machines`
[15]`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`
[16]`http://www.csie.ntu.edu.tw/~cjlin/liblinear/`
[17]`http://scikit-learn.org/0.12/modules/svm.html#complexity`

We tried to validate the hypothesis that the failure of these SVMs in our data was due to its high dimensionality and discard possible implementation issues in our code or its supporting libraries. For this, we evaluated the same classifiers on a smaller feature set of only biological concepts, biologically relevant verbs and syntactic arcs using stratified 5-fold cross-validation on our smallest dataset, the angiogenesis collection. This resulted in training/testing splits of 2,228 and 556 sentences, respectively, and approximately 670 features considered for classification. Results for classification using non-scaled nor normalized feature values are presented in Table B.2.

Using a number of features lower than the number of samples and a reduced number of samples, the first and second degree polynomial kernels and the RBF kernel were able to find a hyper-plane to separate the data, albeit with poor performance. However, the third degree polynomial and sigmoid kernels continued to fail to classify any instance as positive. Possible causes might be the algorithms complexity, libsvm based implementation limitations or the fact that we did not experiment with thorough variations of configuration parameters[18], especially the tolerance for the stopping criterion and the size of the kernel cache.

On Figure 3.13, which compares F1-measure results for each classifier using both raw and scaled/normalized features, we can clearly observe that all classifiers with the exception of the Nearest Centroid (Rocchio) achieved higher or similar scores when feature values were not scaled and normalized. This is also true for recall (Figure B.2), average precision (with the additional exception of K-Nearest Neighbors (kNN) , the worst performing classifier, Figure B.3) and Area Under the ROC Curve (AUC) (Figure B.4) scores. Classification accuracy was not considerably affected by scaling and normalization (Figure B.5), but all algorithms, with the exception of linear SVMs and Naïve Bayes, achieved significantly higher precision when features were scaled and normalized (Figure B.1), at the expense of recall.

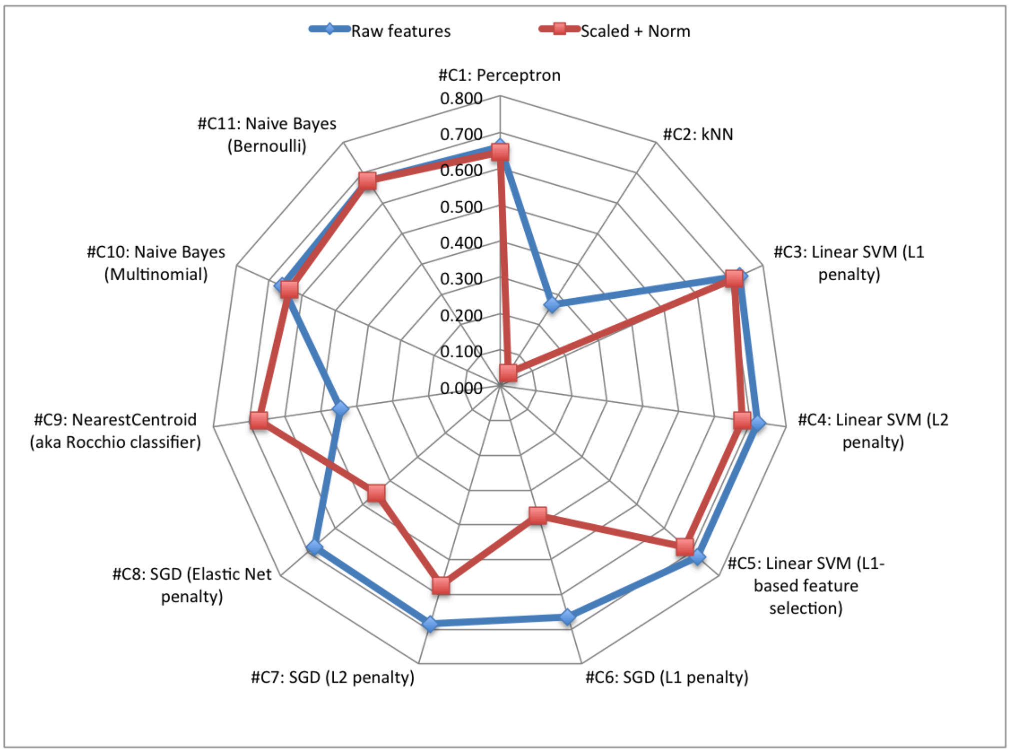


Figure 3.13: F1-measure scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

[18]http://scikit-learn.org/0.12/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC

In order to determine how feature scaling and normalization affected the trade-off between precision and recall of the various candidate algorithms, we plotted precision and recall values for each classifier and observed three different trends (Figure 3.14). Online learning algorithms like the Perceptron and Stochastic Gradient Descent (SGD) based methods were the only ones in which scaling and normalization inverted the relationship between precision and recall, causing precision to increase more than 20% at the expense of recall dropping more than 40% in the case of some SGDs. Linear SVMs, kNN and Naïve Bayes with multinomial event model achieved higher precision than recall with both raw features and normalized values, while the Rocchio and Naïve Bayes considering Bernoulli distribution showed higher recall than precision in both cases.

Linear SVMs proved to perform consistently better than other classification algorithms using raw feature values, presenting only lower recall than SGD methods. When models were induced after scaling and normalization, SVMs also achieved the best F1-measure, accuracy, AUC score and average precision, being only surpassed in precision by SGDs and in recall by the Rocchio classifier.

Since our classification method will be used to rank sentences in an information retrieval system, one important metric we aim to optimize is the AUC value, which estimates the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one [71]. After assessing the performance of all classifiers evaluated and considering the purpose of our application, we selected the Linear SVM with L1-norm based penalization (#C3) as our classification algorithm.

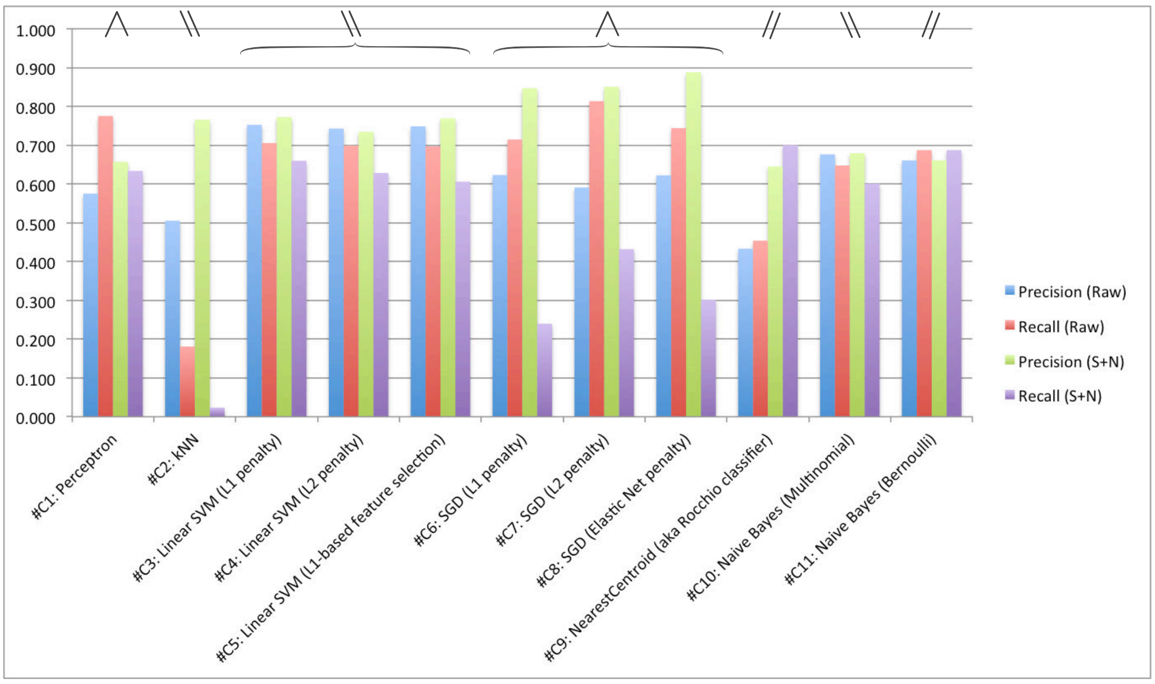


Figure 3.14: Precision and recall comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

Analysis of the effects of scaling and normalization on the performance of our chosen classifier revealed that feature normalization increases precision by 2.1% while decreasing recall by 4.5%. This is reflected on the F-measure and AUC scores by losses of 1.6% and 1.3%, respectively (Figure 3.15). Since we value AUC more than precision and the two percentual points gain are not sufficient to compensate the 4.5% recall loss, we chose to not apply scaling and normalization to feature values.

Linear SVMs performance is highly dependent on the chosen operational parameters, specifically the C penalization parameter of the error term. We tried to optimize the AUC score of the elected SVM by testing multiple values for the C parameter, spaced exponentially far apart from 0.000,001 to 1,000,000. Results showed that the default value of 1.0 provided best performance for the AUC metric (Figure B.6).

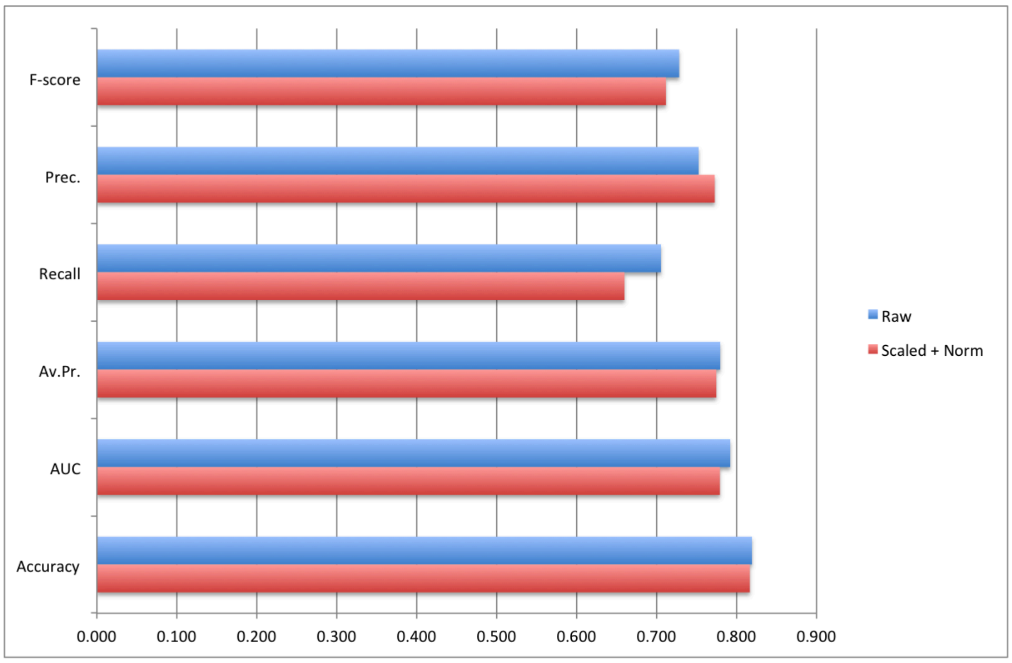


Figure 3.15: Performance of Linear SVM classifier with L1 penalty (#C3) on large feature set using CV[BioNLP+Angio] evaluation.

3.1.6 Results

The previous section (3.1.5) described experiments conducted to choose the best classification algorithm for our data and feature set, which turned out to be a Linear SVM with L1 penalization. This section presents results we obtained for the selected classifier on various sets of features using the multiple evaluation strategies described in section 3.1.4.

3.1.6.1 Baseline

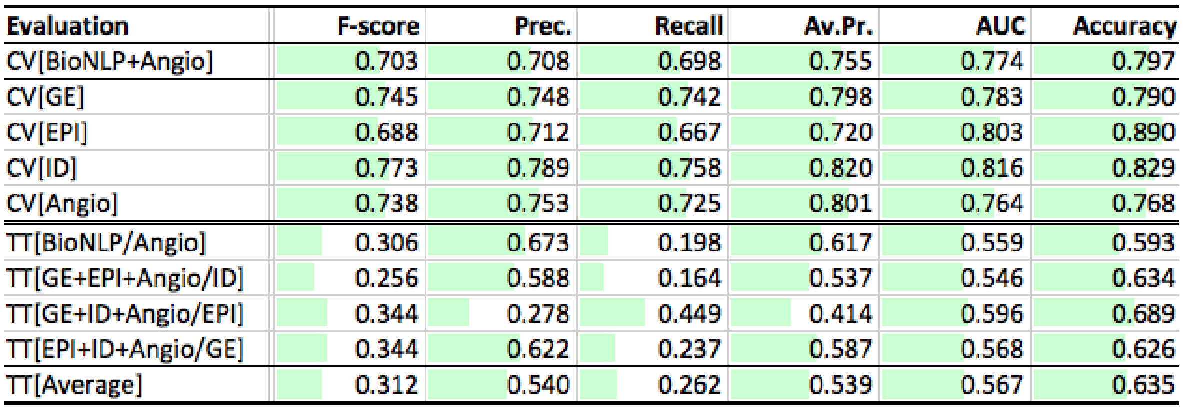
In order to set a starting point for feature exploration and reference values to try to improve the classification model, we established a baseline consisting of one of the simplest possible models for text classification, the bag-of-words (described in section 3.1.3.1). Our goal is to try to identify features of informative sentences that improve this basic model, which can easily cause over-fitting to training data.

Classification results for the Linear SVM trained on bag-of-words features are presented in Table 3.7. Cross-validation results on the whole dataset showed reasonable performance, achieving accuracy of almost 80%, f-measure of 70%, average precision of 75.5% and AUC score of 77.4%. Separate cross-validation evaluations on each dataset suggest the classification method is relatively stable, with performance variances bellow 10%. Best results were attained on the ID dataset, with accuracy, average precision and AUC around 82%, f-measure of 77.3% and precision and recall of 78.9% and 75.8%, respectively.

Leave-one-dataset-out evaluations (labelled TT – different Train/Test collections – in the results table) showed poor overall performance of the bag-of-words model when training and testing on distinct types of sentences that mention different kinds of biological events. Variance of accuracy, f-score, average precision and AUC score are relatively low (around 10%), but precision suffers a variance of almost 40% using this type of evaluation of the bag-of-words model. The main reason for this variance is the TT[GE+ID+Angio/EPI] evaluation, where the model is evaluated on sentences from the EPI dataset. Even though it was the evaluation that showed lowest precision, it achieved significantly higher recall than the other TT evaluations and exhibited the best AUC, accuracy and f-score of the leave-one-dataset-out runs.

The results obtained using the different evaluation strategies we adopted clearly reflect the main limitation of bag-of-words models. Since BoW features are the words mentioned in sentences, this model is highly dependent on the vocabulary used for inducing the classifier. For this reason, it was expected that the performance demonstrated by leave-one-dataset-out evaluations, where the classifiers were tested on text from a different biomedical sub-domain than the ones used for training, would be significantly lower than the cross-validation results measured on more homogeneous data.

Table 3.7: Classification performance results for the baseline (simple bag-of-words).



Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	0.703	0.708	0.698	0.755	0.774	0.797
CV[GE]	0.745	0.748	0.742	0.798	0.783	0.790
CV[EPI]	0.688	0.712	0.667	0.720	0.803	0.890
CV[ID]	0.773	0.789	0.758	0.820	0.816	0.829
CV[Angio]	0.738	0.753	0.725	0.801	0.764	0.768
TT[BioNLP/Angio]	0.306	0.673	0.198	0.617	0.559	0.593
TT[GE+EPI+Angio/ID]	0.256	0.588	0.164	0.537	0.546	0.634
TT[GE+ID+Angio/EPI]	0.344	0.278	0.449	0.414	0.596	0.689
TT[EPI+ID+Angio/GE]	0.344	0.622	0.237	0.587	0.568	0.626
TT[Average]	0.312	0.540	0.262	0.539	0.567	0.635

3.1.6.2 N-Grams

The first step we performed to improve our feature set was extending the basic bag-of-words model with n-grams (detailed in section 3.1.3.2). We tested the use of bigrams (plus words) (Table B.3), trigrams (plus words and bigrams) (Table B.4) and 4-grams (plus words, bigrams and trigrams) (Table B.5). Icons in tables are visual indicators to help compare performance values to the baseline results. A green upward arrow indicates a performance measure of a given evaluation yielded results better than the corresponding baseline value, shown on Table 3.7. A red downward arrow indicates that value is lower than the baseline.

Cross-validation evaluation results displayed improvements across all measures, with the exception of recall reductions between 1 and 3% for cross-validation runs on the EPI and ID datasets. Figure 3.16 compares the performance of the various n-grams combinations using cross-validation on the whole dataset. Trigrams proved to be the best n-gram combination, showing results consistently better than the baseline (BoW on the figure) and other tested n-gram combinations, with the exception of bigrams, which present better recall at the expense of precision. The use of trigrams provided a 6% precision boost, 2% AUC increase and improvement of 2.5% on both accuracy and f-measure.

Leave-one-dataset-out evaluation results (Figure 3.17) show that the use of n-grams significantly increases prediction precision, average precision and accuracy at the expense of recall loss, resulting in reduced F1 and AUC scores.

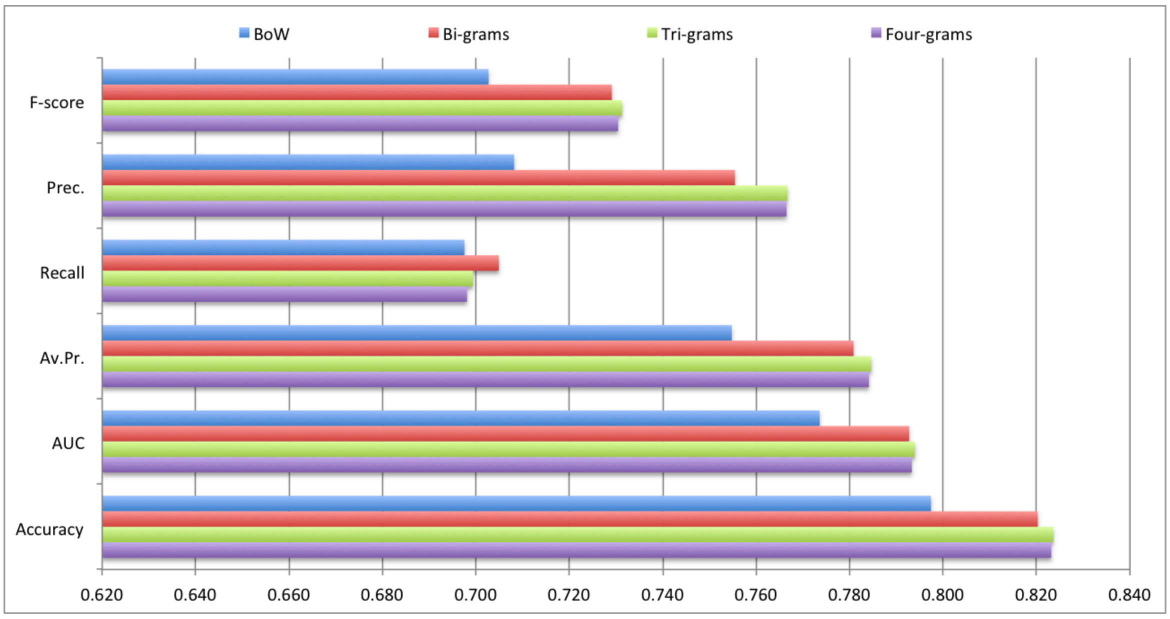


Figure 3.16: Classification performance comparison for n-grams using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

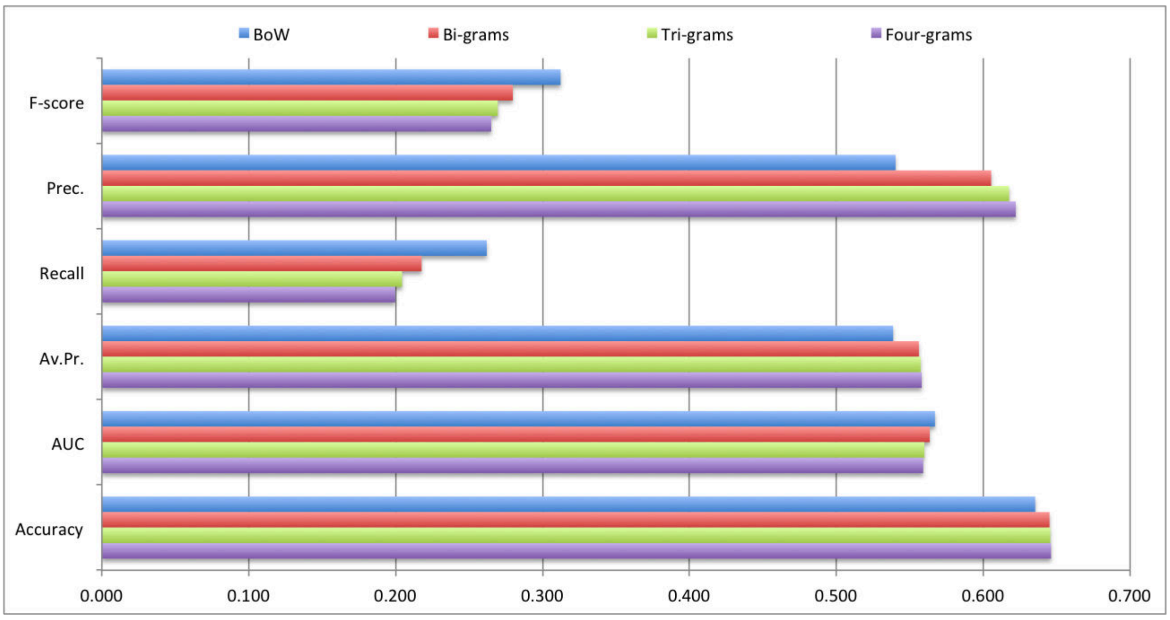


Figure 3.17: Classification performance comparison for n-grams using leave-one-dataset-out evaluation averages (TT[Average]).

3.1.6.3 TF-IDF Vectors of Trigrams

TF-IDF is a form of normalization of absolute term counts such as the ones used in bag-of-words or n-grams models, which takes into account the relative occurrence of certain terms across all documents of a corpus. If is commonly used in IR and text classification applications to improve both retrieval and classification results.

We tested normalizing our trigrams classification model using TF-IDF values to try to improve prediction results. Detailed results for TF-IDF vectors of simple bags-of-words and bags-of-trigrams are displayed in Tables B.6 and B.7, respectively.

Prediction performance of our Linear SVM classifier for simple bag-of-words (BoW) versus TF-IDF normalized BoW, bag-of-trigrams and TF-IDF normalized bag-of-trigrams is plotted in Figure 3.18. Scores were calculated using 10-fold cross-validation on the complete dataset. Results for the leave-one-dataset-out evaluation strategy are presented in Figure 3.19.

The use of TF-IDF scores, both for the simple BoW and bag-of-trigrams models, significantly improved precision of the estimators using both evaluation strategies, with increases over the BoW baseline of 7% for the CV evaluation and 8% for the TT evaluation. Cross-validation results showed consistent improvements in all evaluation metrics, with increases over the previous trigram model in term of precision and AUC (+1.2%), recall (+1.8%) and f-score (1.5%). Leave-one-dataset-out performance estimations suffered on recalling relevant sentences, causing the f-measure and AUC scores to decrease slightly (-0.8% recall, -0.4% AUC and -1.3% F1).

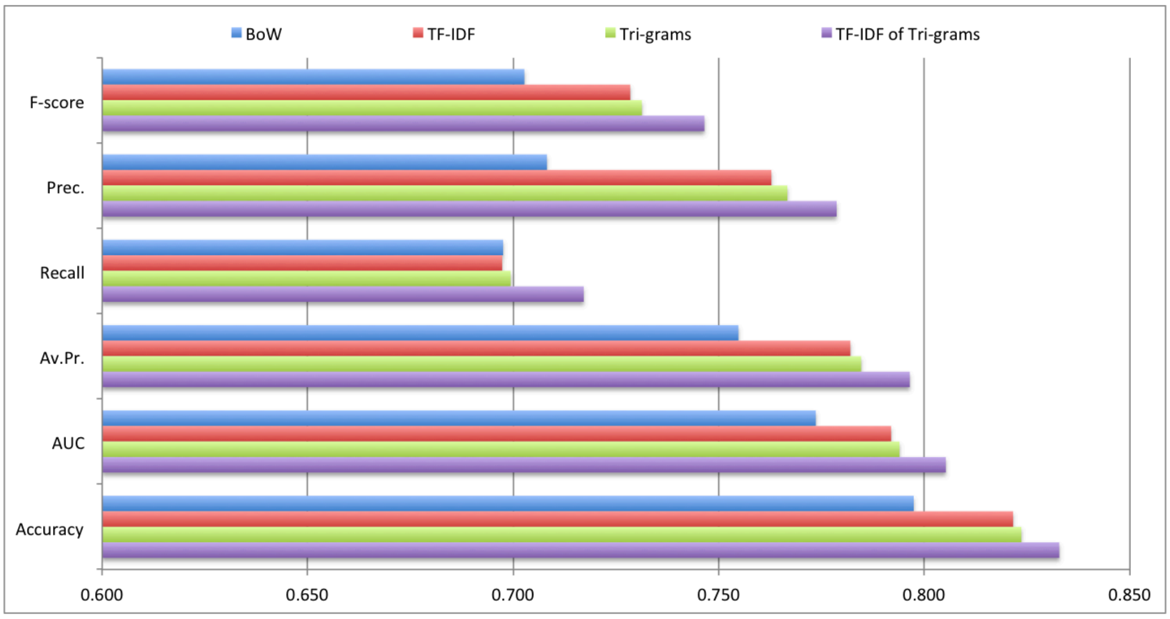


Figure 3.18: Classification performance comparison for TF-IDF vectors of n-grams using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

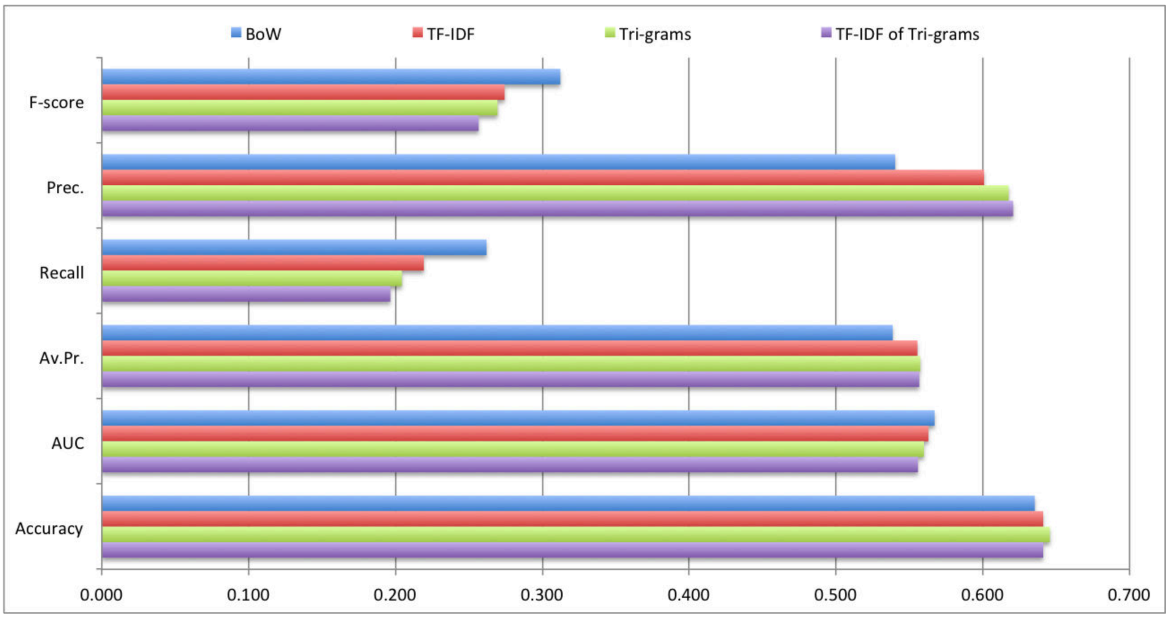


Figure 3.19: Classification performance comparison for TF-IDF vectors of n-grams using leave-one-dataset-out evaluation averages (TT[Average]).

3.1.6.4 TF-IDF Vectors of Trigrams, POS and Chunk Tags Combinations

In addition to TF-IDF vectors of word trigrams, we tested the value of also using TF-IDF scaled vectors of trigrams of POS and chunk tags (see sections 3.1.3.4 and 3.1.3.5). These types of linguistic features model sentence structure at the grammatical level without tying the model to specific lexicons and should improve generalization to different kinds of sentences.

Classification results for TF-IDF vectors of trigrams of POS tags are presented in Table B.8. Table B.9 contains results for a model considering TF-IDF trigrams pairing POS and chunk tags. We also tested the combination of these two features (Table B.10) and their addition to the previously evaluated feature set of TF-IDF vectors of word trigrams (Table B.11). Since we're trying to improve performance over the current best model – the TF-IDF trigrams model – these results are shown in comparison to the results achieved with that model.

Both the POS, the POS-Chunk pairs and the combination of both models perform considerably worst than the TF-IDF trigrams model when evaluated using our cross-validation strategies (Figure 3.20), suggesting that when training and testing on datasets dominated by the same lexicon, a trigram model of words can better classify text. The addition of these grammatical features to the TF-IDF trigram model also caused a slight decrease in performance, since these features augment dimensionality considerably and can add redundancy and noise to the lexical feature set.

Leave-one-dataset-out evaluations (Figure 3.21) showed that even though these models are less precise than lexical models, like the bag-of-words and word trigrams, they are better at recalling positive instances and classify them higher than negative ones, as proved by increased recall, AUC and F-score.

We can also conclude that pairing POS tags with chunk tags yields a better classification model than using only POS tags. Combining trigrams of POS tags with trigrams of POS-Chunk tags pairs seems to generate less precise classification models but with higher recall.

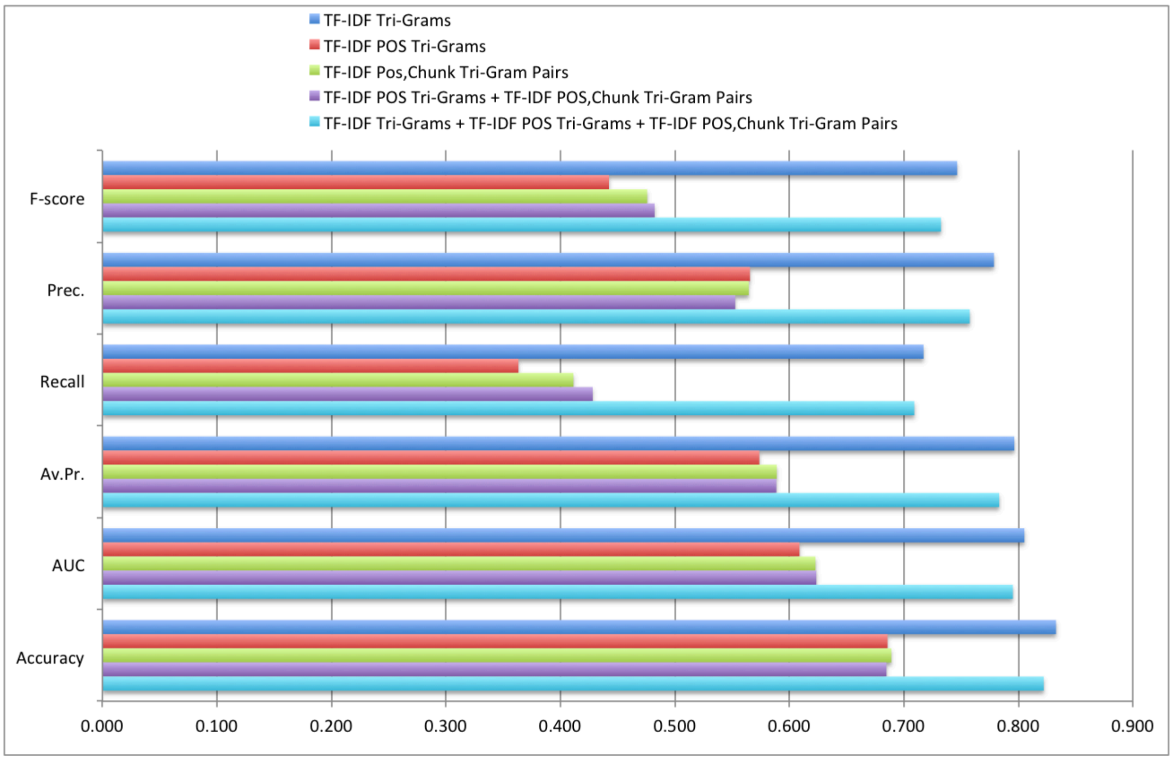


Figure 3.20: Classification performance comparison for various combinations of TF-IDF vectors of features using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

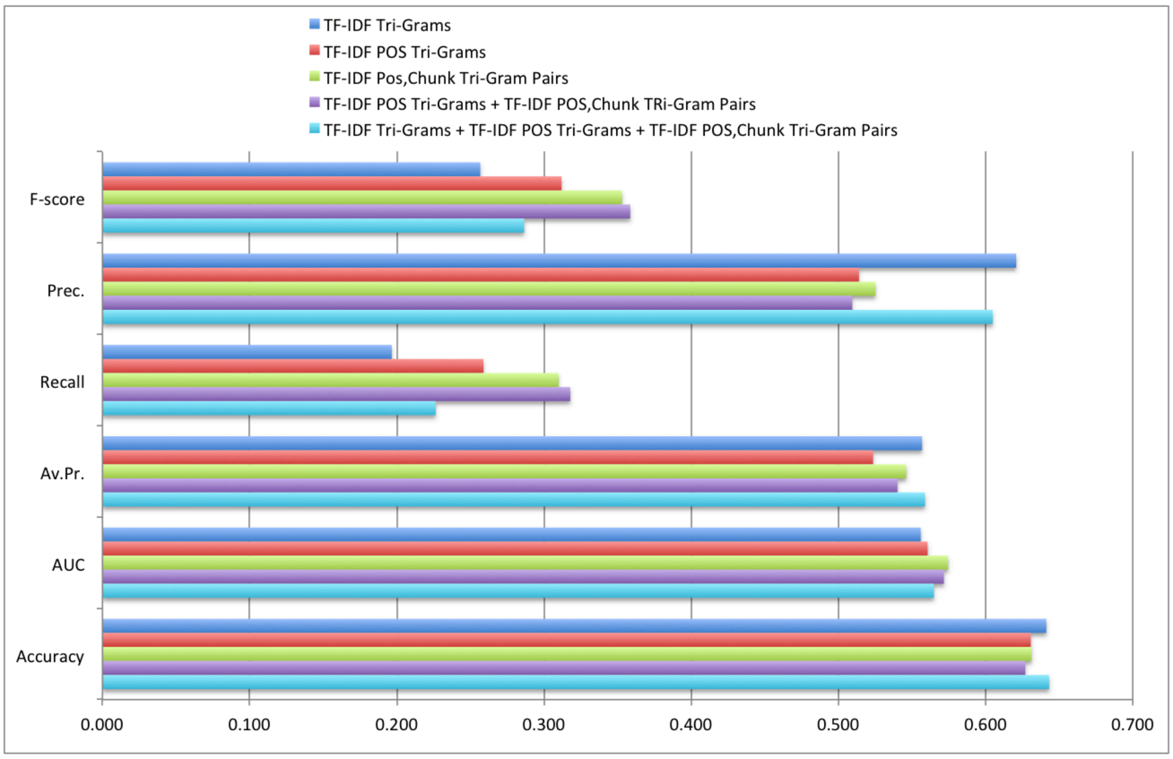


Figure 3.21: Classification performance comparison for various combinations of TF-IDF vectors of features using leave-one-dataset-out evaluation averages (TT[Average]).

3.1.6.5 Verbs, Concepts and Syntactic Arcs

After verifying that trigrams of POS and chunk tags pairs can be helpful in recalling and ranking informative sentences relatively different from the ones the model was trained on, albeit with a minor precision loss, we extended the TF-IDF vectors model with more lexical features, such as the presence of biologically relevant verbs in sentences (previously mentioned in section 3.1.3.6), semantic features related to biological concepts (described in section 3.1.3.7) and syntactic features related to the number of arcs of a certain distance (in number of hops) between recognized concepts and biologically relevant verbs (section 3.1.3.8).

Comprehensive results using our considered evaluation strategies for verbs, concepts and syntactic arcs are presented in Tables B.12, B.13 and B.14, respectively. Table B.15 shows results for the combination of these three types of features and Table B.16 details results from the combination of the previously evaluated TF-IDF vectors model with these new features.

As we can visualize in Figure 3.22, none of these new features were able to outperform the previous TF-IDF classification model when evaluating performance using our cross-validation strategy. The features related to syntactic arcs presented the poorest performance, classifying less than half of instances correctly and failing to recall more than 5% of informative sentences. Biological concepts alone also proved to be insufficient for proper sentence discrimination, followed by biologically relevant verbs, which were the best of these new three sets of features but also achieved considerably lower results than our baseline. Combining these feature sets improved performance across all metrics, but was still not sufficient to surpass the previous models.

Model evaluation using the leave-one-dataset-out strategies revealed the interesting finding that while none of the features alone achieved brilliant results, the combination of these three sets of features outperformed previous strategies in terms of f-measure, recall, accuracy and AUC score with only marginal precision loss (Figure 3.23). We can conclude that this combination of features has good potential at correctly recalling and ranking informative sentences from sentences lexically different from the ones used in the training data. Additionally, even though the extension of the combined TF-IDF model considered before with these features did not improve prediction performance, the SVM classifier was not significantly affected by the feature set expansion.

TF-IDF models present a considerably higher dimensionality than using the verbs, concepts and syntactic arcs combined. Feature values are also of different types and ranges. TF-IDF scores are real, verbs and concept features are represented as binary values and syntactic arcs present discrete feature values. For this reasons, it was challenging to combine all the features together without decreasing prediction accuracy. We tried applying scaling and normalization (individually and combined) aiming to improve classification performance, but our efforts proved unsuccessful. Attempts to reduce data dimensionality through feature selection techniques such as **pca!** (**pca!**) and chi-squared analysis also failed to produce satisfactory results.

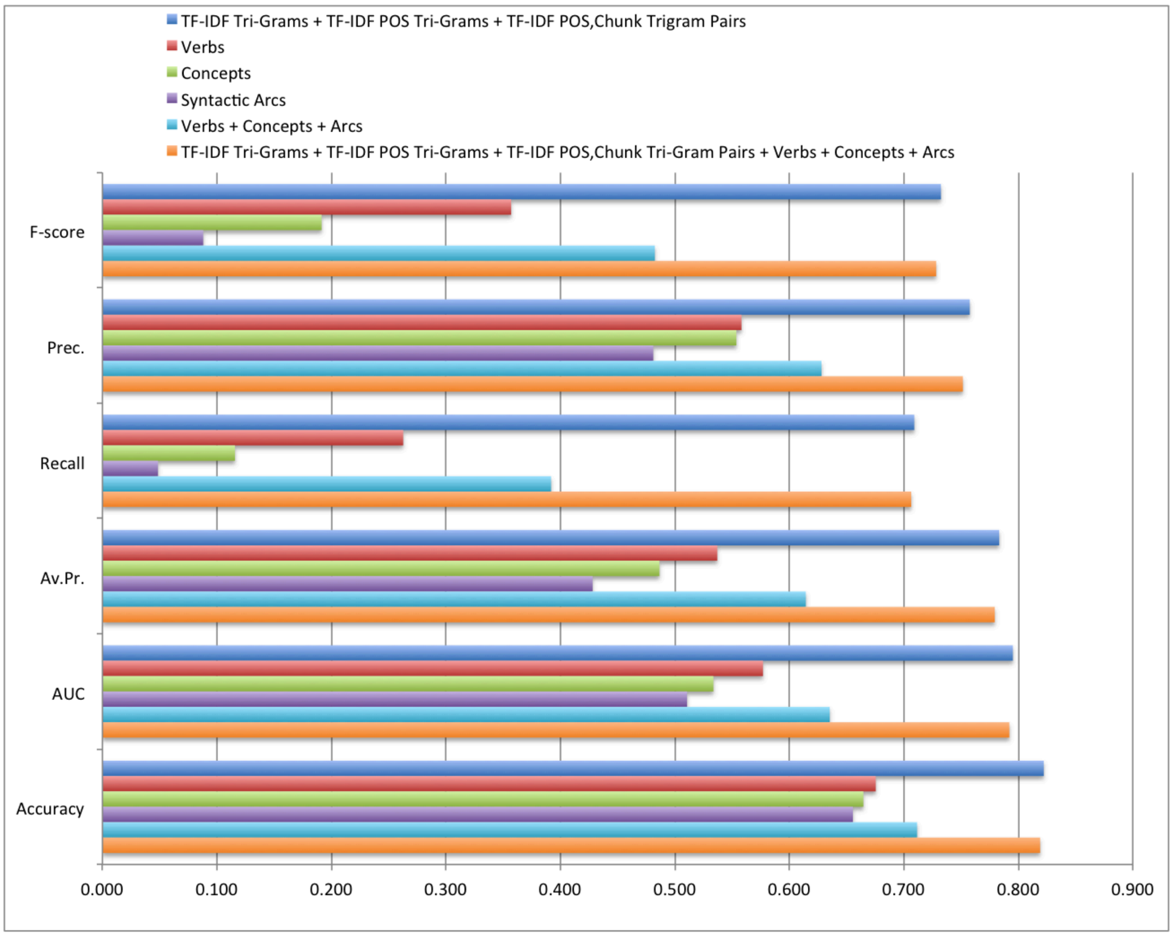


Figure 3.22: Classification performance comparison for verbs, concepts and syntactic arcs features using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

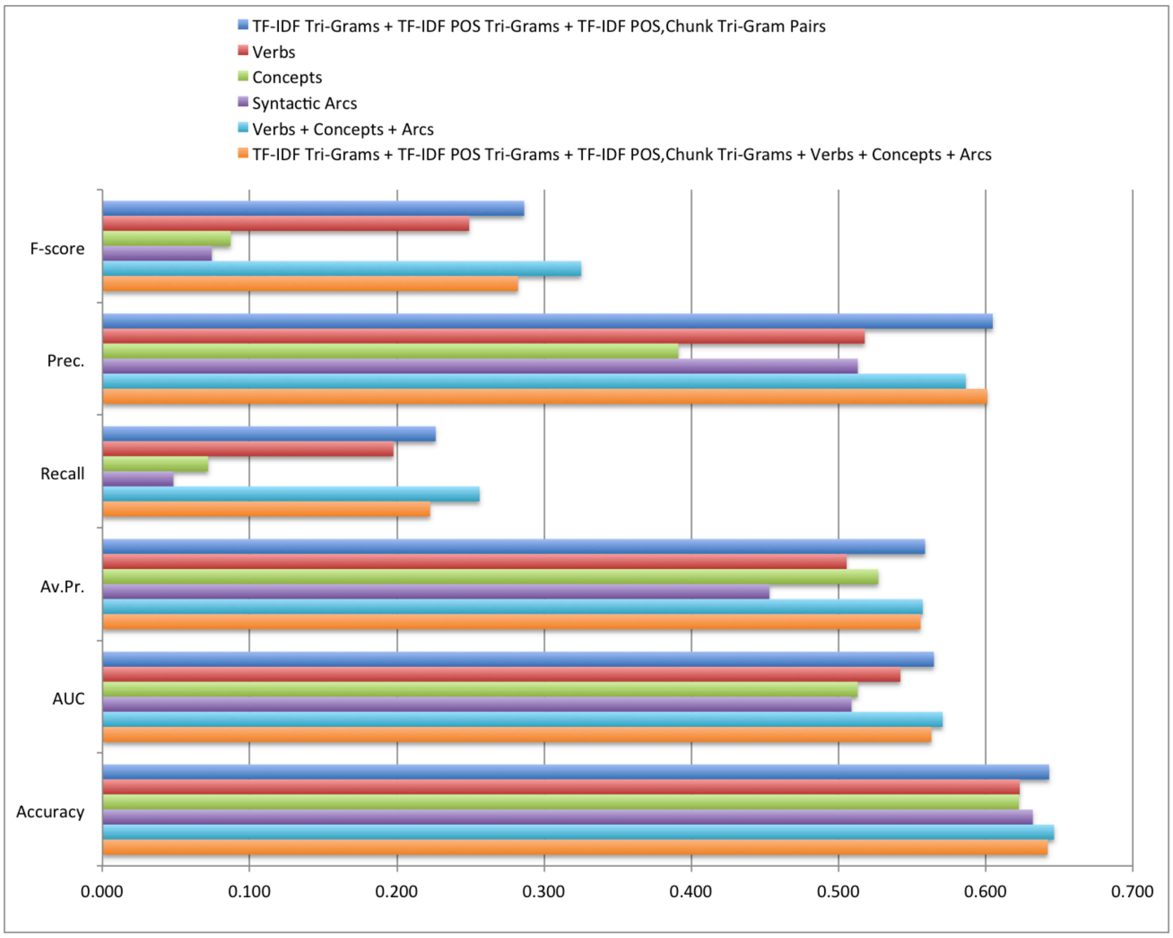


Figure 3.23: Classification performance comparison for verbs, concepts and syntactic arcs features using leave-one-dataset-out evaluation averages (TT[Average]).

3.1.6.6 Overall Features Comparison

The previous sections presented results for various classification models considering different sets of features, both in isolation and used in multiple combinations. Here we compare all the models that were evaluated and analyse their performance in regard to each of the evaluation strategies.

Results for the stratified 10-fold cross-validation strategy on the complete aggregated dataset collection are plotted as a bar graph similar to the ones used previously for feature sets comparison in Figure B.7. To facilitate visual comparison of each model we plotted the same data as a line graph (Figure 3.24), where we can clearly observe that only four of the proposed models achieved higher performance results than our baseline model (the bag-of-words model, Table 3.7) when considering the cross-validated evaluation.

TF-IDF trigrams (Table B.7) achieved best performance across all metrics, followed closely by the TF-IDF bag-of-words (Table B.6); the combined model of TF-IDF trigrams, TF-IDF vectors of POS tags trigrams and TF-IDF vectors of POS-Chunk tag pairs trigrams (Table B.11); and finally the complete model combining all the features considered, TF-IDF normalized trigrams, TF-IDF vectors of POS tags trigrams, TF-IDF vectors of POS-Chunk tag pairs trigrams, verbs, concepts and distance of syntactic arcs between verbs and concepts in the same sentence (Table B.16). Performance differences between these top performing methods are marginal, with variance under 3%. The top performing classifier achieved improvements over the baseline of 7% precision and 2% recall, reflected in the f-measure by an increase of 4.4%. AUC was improved by 3.2%, average precision by 4.2% and overall accuracy by 3.5%.

When considering the leave-one-dataset-out evaluations averages (bar plot in Figure B.8 and line graph in Figure 3.25), none of the classification models was consistently better over all performance measures. The TF-IDF trigrams model that proved to be the best using the cross-validation evaluation achieved best precision using the TT strategy, followed closely by the complete TF-IDF vectors model and the model using the whole feature set.

The model induced on our complete feature set achieved higher recall (+2.6%), f-score (+2.6%) and AUC (+0.7%) than the TF-IDF trigram model when considering the leave-one-dataset-out evaluation, only suffering a 2% precision hit. Since the classification method will be used to identify and rank informative sentences on an information retrieval system, the use of the extended feature set can be beneficial for the correct discrimination and ranking of sentences lexically different from the sample data used to induce the classification model. For this reason, we chose the model aggregating TF-IDF vectors of trigrams, TF-IDF vectors of POS tags trigrams, TF-IDF vectors of POS-Chunk tag pairs trigrams, biologically relevant verbs, biological concepts and distance of syntactic arcs between verbs and concepts in the same sentence for integration in our sentence retrieval solution.

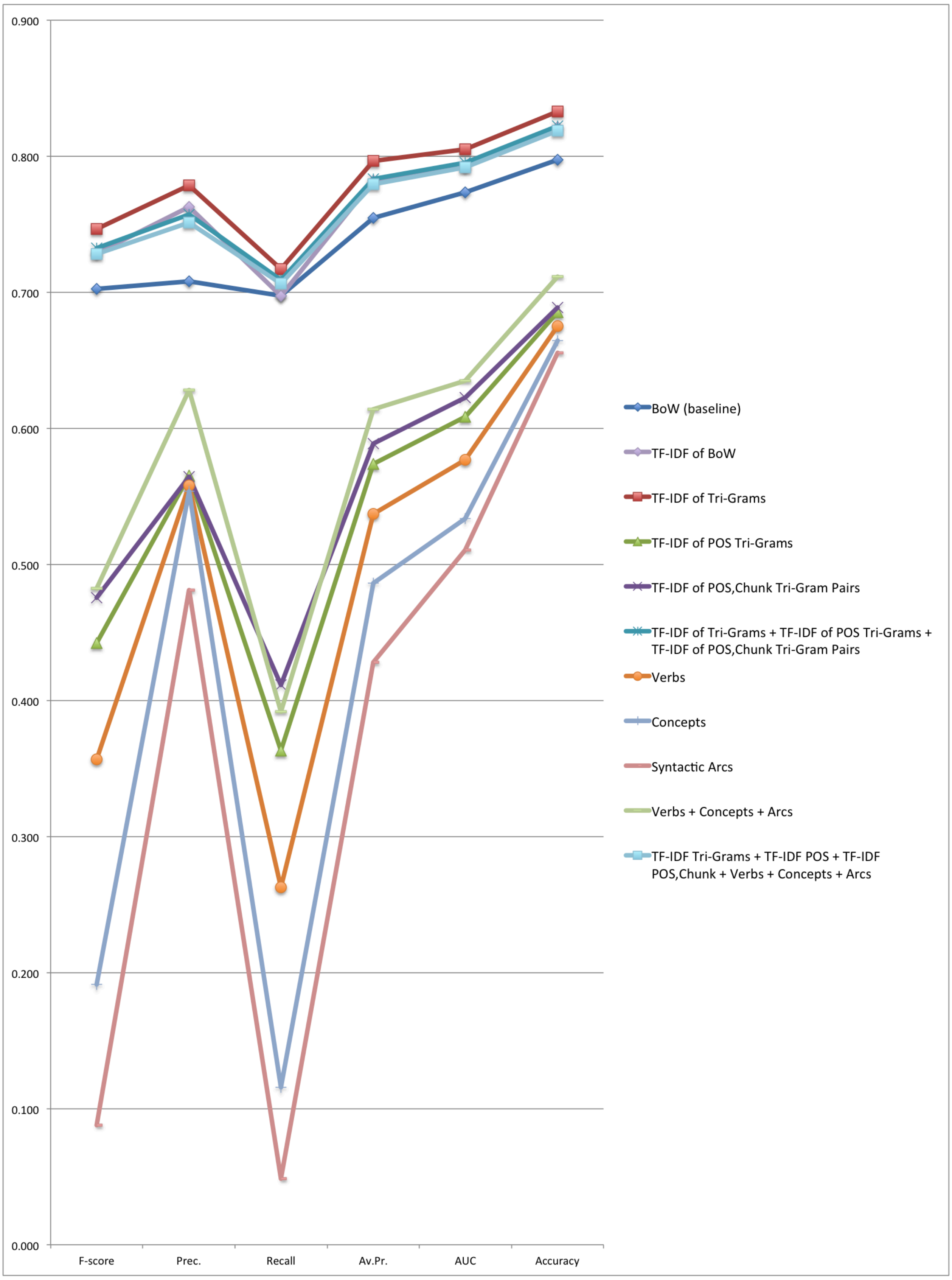


Figure 3.24: Classification performance results for different types of features using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

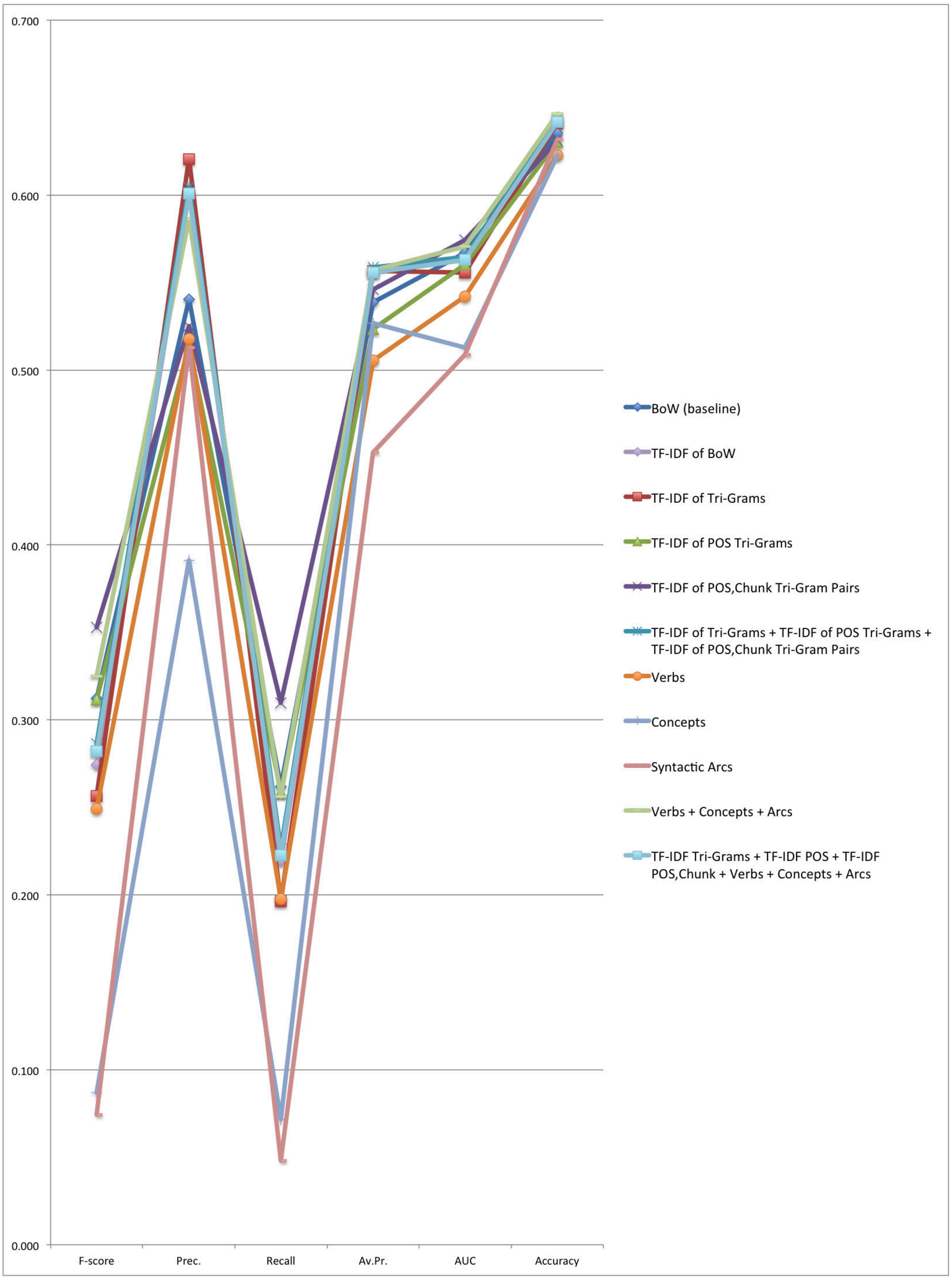


Figure 3.25: Classification performance results for different types of features using leave-one-dataset-out evaluation averages (TT[Average]) plotted.

3.1.7 Summary

In the first part of this chapter we presented the work conducted for the development of a classification model for informative sentences in the biomedical domain.

We used supervised learning techniques (section 3.1.1) to induce prediction models from a set of resources where key sentences were tagged as being informative. Since no gold standard corpora exist for biomedical key sentence extraction, we gathered four freely available datasets where sentences were annotated with relevant bio-events and tagged sentences mentioning or describing the events as being informative, in order to produce training data (section 3.1.2).

A set of lexical, grammatical, syntactic and semantic features were considered for model construction, such as simple bags-of-words, n-grams, POS and chunking tags, the presence of biologically relevant verbs in sentences, as well as mentions of biological concepts, and existence of syntactic dependencies between relevant verbs and concepts (section 3.1.3). We performed experiments with various techniques for feature selection and normalization, but only the use of TF-IDF vectorization parameters showed beneficial results for our data and methods.

Multiple evaluation strategies were adopted to guide the development of our classification model (section 3.1.4). After estimating the performance of several classification algorithms in our data and feature set, we selected a linear SVM for model induction, since it proved to be the best classifier of all the methods we tested (section 3.1.5).

We established a baseline feature set of a simple bag-of-words model that was iteratively extended with additional features. Exploratory work was then conducted to evaluate the impact that each additional feature had on our classification model considering various methods of performance evaluation (section 3.1.6). Our results suggest that bag-of-words and n-grams models are capable of accurately identifying informative sentences from biomedical texts and the use of TF-IDF weighting techniques proved to be beneficial in increasing precision of those models.

Constructing a classification model that combines all the features we considered for informative sentence discrimination posed challenges that were not easily tackled by common feature selection and data normalization techniques. Leave-one-dataset-out evaluation strategies suggest that the features we used to extend the basic BoW model can aid in better recalling and ranking informative sentences from sub-domains distinct from the ones the model was trained on.

A possible interesting approach for future work regarding the improvement of the developed classification model could focus on the use of ensemble methods that combine various classifiers induced from different feature sets.

3.2 Sentence Retrieval

The first part of this chapter detailed the methodology we used to build and evaluate classification models that can be applied for the identification and ranking of informative sentences in the biomedical domain. In this section we describe an information retrieval approach that integrates these models and facilitates fast retrieval and ranking of informative sentences.

Classical information retrieval systems return entire documents, which can be long and require considerable reading time to find the specific information that users seek, frequently present in a small passage of text. Moreover, IR can fail to find the most relevant documents to answer a certain information need, because the words used to express a query are lexically different from the words used in relevant sources.

We propose a sentence-based indexing and retrieval method that, for a given user query, allows seeking relevant sentences from the scientific literature that answer the user information need. The proposed system – named *PhraseRank (Phrank)* – supports free-text and concept-based queries, enriches search results with relevant concept annotations and ranks sentences using various customizable strategies.

Human users can interact with the system using a web-based User Interface (UI) . This UI is powered by an HTTP Representational State Transfer (REST) API that enables easy integration of the application search, ranking and concept discovery features by third-party tools.

3.2.1 Architecture

Web-based systems lower the adoption barrier faced by users in regard to initial setup, on-going system maintenance and minimum computational system requirements, as they live in the cloud and, from a user point of view, require no installation, no maintenance, and place the toll of resource consumption on machines owned and managed by service providers. Moreover, these types of applications allow service providers to perform system upgrades, like bug fixes, addition of new features and data in a seamless way to end-users.

Information retrieval systems have the intrinsic necessity of requiring access to indexed data collections, which are usually very large and would need plenty of time to download. Additionally, it is not always in the best interest of service providers to distribute these data collections in their complete form, as there might be both licensing issues and requirements for protection of intellectual property. For these reasons, contemporary applications, especially IR systems, are usually made available in the form of web applications powered by a set of web services that users can access using a web browser.

PhraseRank architecture, illustrated in Figure 3.26, is based on the widely used client-server model [72]. First, a user user contacts the Phrank application server through a browser and requests the download of the client-side components of the system, comprised of HyperText Markup Language (HTML) , Cascading Style Sheets (CSS) and JavaScript (JS) resources. Second, the web browser renders the web application and waits till the user begins typing in a query. Third, the browser executes asynchronous HTTP requests using Asynchronous JavaScript and XML (AJAX) technologies to communicate with web services in Phrank application server for multiple purposes, such as obtaining query auto-complete suggestions, submitting search requests and re-ranking results. These requests prompt the application server to retrieve query suggestions or query results from the sentences index server. Whenever metadata about biological concepts is required, the application server contacts a concept data store to acquire the required information before, at last, sending a response back to the user web browser. The document-oriented nature of index servers (described later in section 3.2.4) and our requirement to store additional structured data about biological concepts, such as preferred names and external database references, triggered the need to use a second data store, in addition to the sentences index.

All application state is managed client-side and the server provides a pure stateless web services layer, which only serves static resources and exposes the system API. The application client is structured around the Model-View-Presenter (MVP) pattern [73] and uses an event-driven messaging architecture to achieve decoupled communication across modules.

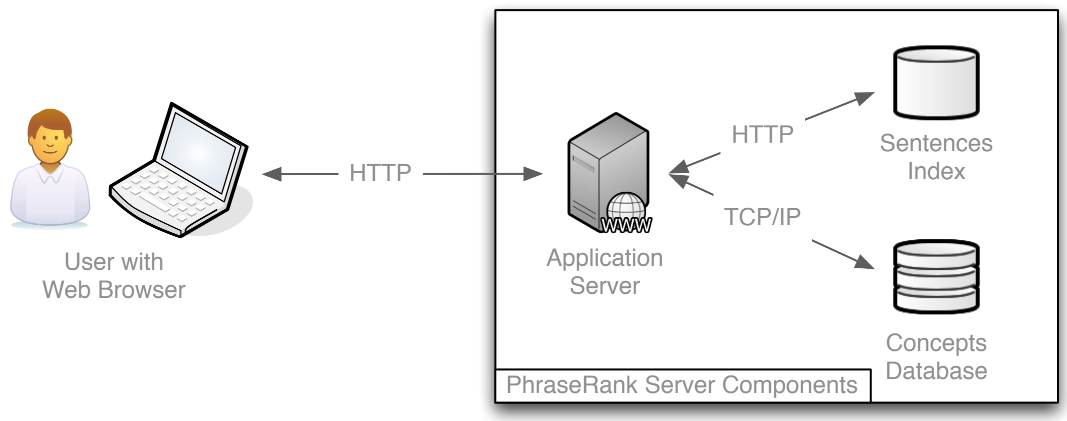


Figure 3.26: PhraseRank System Architecture Overview.

3.2.2 Technology Stack

The choice of technologies used for the development of the proposed system, driven by our architectural requirements, led to the use of the heterogeneous software stack illustrated in Figure 3.27.

The client user interface application, with which users interact, was built using JavaScript (JS) and makes use of a collection of supporting libraries that provide common functionality required by most web applications; and frameworks that aid in code modularization, while offering boilerplate structures for common architectural patterns. The most relevant client dependencies are:

- **Require.js**[19]: JavaScript file and module loader. Allows asynchronous loading of application modules during development and facilitates *"compilation"* and *minification* of the whole application code into a single file, for use in production;
- **Backbone.js**[20]: Minimal set of data-structuring (models and collections) and user interface (views and URLs) primitives that are generally useful when building web applications with JavaScript;
- **Backbone.Marionette**[21]: Application framework for Backbone.js that offers a collection of common design and implementation patterns inspired by composite application architectures, event-driven architectures, messaging architectures, and more;
- **Underscore.js**[22]: Utility-belt library for JavaScript that provides functional programming support methods, such as `map`, `reduce`, `filter`, `forEach`, `invoke`, `bind`, and others;
- **Handlebars**[23]: Dynamic semantic templates library for JavaScript;
- **jQuery**[24]: Library for cross-browser HTML Document Object Model (DOM) manipulation, event handling, animation and AJAX;
- **Flatstrap**[25]: HTML and CSS framework based on Bootstrap[26] for consistent and responsive application layout construction.

[19]`http://requirejs.org/`
[20]`http://backbonejs.org/`
[21]`http://marionettejs.com/`
[22]`http://underscorejs.org/`
[23]`http://handlebarsjs.com/`
[24]`http://jquery.com/`
[25]`http://flatstrap.org/`
[26]`http://getbootstrap.com/`

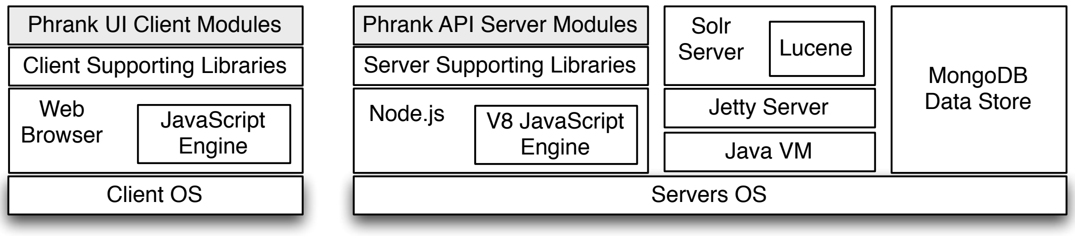


Figure 3.27: PhraseRank System Technology Stack.

As mentioned when describing the system client-server architecture, the server-side is composed by three main top-level components – an HTTP application server that implements the system API, a search server that provides full-text and field-based search capabilities, and a data repository used to store metadata about biological concepts. We selected node.js[27] as our application server, Apache Solr[28] as the search server solution and MongoDB[29] for the concepts repository. The remainder of this section discusses these technology choices.

There are a multitude of options to choose from when considering *open-source* and *freely available* HTTP application servers and the programming languages supported by them. Among the list of popular choices are Apache Tomcat[30], Jetty[31], GlassFish[32] and WildFly (formerly JBoss)[33] for Java applications; the Apache[34] and nginx[35] web servers that support various programming languages, such as Python, PHP, Perl and Ruby; and the previously mentioned node.js platform for JavaScript applications.

Since the application client, targeted at web browsers' JS run-times, was programmed in JavaScript, we choose to use the same language in the server-side, with the intent of reducing code-duplication and redundancy between the client and server layers. This way we were able to easily share some data structures, such as ranking strategies configuration, URL query parameters and index field names between client and server code.

Phrank server-side API modules run on top of the node.js platform. Node.js is a platform built on Google Chrome's V8 JavaScript engine[36] for developing fast and scalable network applications. It uses an event-driven, non-blocking I/O model that allows handling thousands of concurrent requests more efficiently than with the more common concurrency model where OS threads are employed [74]. For this reason, we consider it a suitable platform to implement a thin service layer with the main purpose of interacting with external servers. Server modules make use of the following third-party libraries:

- **Express**[37]: A minimal web application framework for node.js;
- **node-solr**[38]: Node.js client for the Apache Solr server;
- **node-mongodb-native**[39]: Native node.js client for the MongoDB data store;

[27]http://nodejs.org/
[28]https://lucene.apache.org/solr/
[29]http://www.mongodb.org/
[30]https://tomcat.apache.org/
[31]http://www.eclipse.org/jetty/
[32]https://glassfish.java.net/
[33]http://www.wildfly.org/
[34]https://httpd.apache.org/
[35]http://nginx.org/
[36]https://code.google.com/p/v8/
[37]http://expressjs.com/
[38]https://github.com/gsf/node-solr
[39]https://github.com/mongodb/node-mongodb-native

Search server and concept repository choices were driven by the University of Aveiro Bioinformatics Group strategy and current text mining framework. Past and on-going research projects at the group already make fruitful use of Apache Solr for full-text indexing and search and MongoDB for flexible storage of structured concepts metadata. Thus, aiming to align the Phrank application with the existing ecosystem, these platforms were the evident choices for the implementation of our system.

Apache Solr is an open-source search platform based on Apache Lucene [40], a text search engine library written in Java. It supports and extends all Lucene features, such as full-text search, ranked searching, pluggable ranking models and configurable text analysis; and exposes search capabilities through an HTTP REST-like API. It also supports index schema configuration using XML files, allowing fine-grained customization of text analysis methods used on a field-by-field basis.

MongoDB is an open-source document-oriented non-relational database and *arguably* the leading NoSQL [75] database[41]. It stores JSON-style documents with dynamic schemas, facilitating agile application development, where data structures evolve quickly over consecutive iterations. Indexing is supported on any attribute, allowing fast queries by any indexed field. Additionaly, it supports map/reduce methods for result aggregation and data processing and auto-sharding for seamless horizontal scaling.

Potential alternatives for the search component could be Elasticsearch[42], which is, like Solr, also based on Apache Lucene; or Sphinx[43], another open-source search server, written in C++. Any open-source relational database, like MySQL[44], MariaDB[45] (a MySQL fork) or PostgreSQL[46]; or popular NoSQL document-oriented databases like CouchDB[47], RethinkDB[48] or RavenDB[49] could be considered to hold the concepts repository. However, for the purpose of this thesis, we did not consider nor compared any of those systems.

3.2.3 Validation Corpus

The proposed system architecture is intended to support indexing and searching of large corpora of sentences from biomedical research articles. In the future, we plan to classify, index and provide searching and ranking facilities over the complete MEDLINE collection, which already contains over 22 million citations and is growing at a rate of approximately 4% per year [2]. However, pre-processing, classifying and indexing a great amount of text is an extremely time-consuming task and it would be impractical to embark in such an endeavour without first testing and tuning our proposed approach on a smaller dataset. Therefore, for the purpose of testing and validating the methods developed in this thesis, we selected a small subset of the whole MEDLINE abstracts collection.

Since one of the current areas of interest and active research of the University of Aveiro Bioinformatics Group is neurodegeneration, including the study of neurodegenerative processes that provoke diseases such as Alzheimer's, Parkinson's and Huntigton's, we compiled a corpus comprised of MEDLINE abstracts tagged with MeSH terms related to neurodegenerative diseases. Using the PubMed API – Entrez Programming Utilities [76] – we fetched all the English citations with abstract that are identified with neurodegeneration MeSH terms. Listing 4 describes the query used to obtain the publications. This resulted in a corpus of 136,985 abstracts with corresponding metadata formatted according to the MEDLINE XML format. After applying sentence splitting, we extracted 1,207,952 sentences from the abstracts, including sentences from publication titles.

```
"Neurodegenerative␣Diseases"[MeSH Terms] OR
"Heredodegenerative␣Disorders,␣Nervous␣System"[MeSH Terms]) AND
```

[40]https://lucene.apache.org/core/
[41]http://www.mongodb.com/leading-nosql-database
[42]http://www.elasticsearch.org/
[43]http://sphinxsearch.com/
[44]https://www.mysql.com/
[45]https://mariadb.org/
[46]http://www.postgresql.org/
[47]https://couchdb.apache.org/
[48]http://www.rethinkdb.com/
[49]http://ravendb.net/

```
English[Language] AND hasabstract[text]
```

Listing 4: PubMed query used to fetch neurodegenerative diseases corpus.

3.2.4 Indexing Model

The indexing model we devised for PhraseRank was conditioned by the overall goal of the system and its intended capabilities. It was built to address the following requirements, which were translated into system features:

1. full-text search;
2. concept-based search;
3. ranking of sentences according to various customizable strategies, including classification scores (information content predictions) and estimated semantic richness (based on concept diversity);
4. query suggestions for concepts (also known as automatic completion of query terms);
5. visual identification of recognized concepts in search results;
6. referencing to source publications (including journal and publication date) from where sentences were extracted;
7. linking to external reference databases for biological concepts identified in the texts.

Figure 3.28 illustrates the overall indexing workflow. First, publications are pre-processed using Neji [50], which uses dictionaries of concept names and gene/protein recognition models to annotate source MEDLINE XML files. Annotated files are saved in IeXML format [77] and dependency parses of sentences are saved to CoNLL files [59]. Second, annotated XML and CoNLL files are parsed and features are extracted from sentences, along with metadata present in the XML files, such as publication identifiers (PMIDs), journal name and publication date details. Third, sentences are classified using several customizable classification models and confidence scores for each model are calculated. Last, sentences are indexed, along with classification scores, recognized concepts and relevant metadata.

It should be noted that, even though for the purpose of this thesis we classified and indexed sentences from MEDLINE publications in XML format, our workflow is applicable to text in any format, as long as a parser is implemented to read the source files.

Sentence data are indexed and stored in our index server – Apache Solr – according to the simplified schema described in Table 3.8. Field names are shown in the *Field* column; the *Type* column designates field data types (influencing the type of analysis performed on data, if any); the *Indexed* and *Stored* columns indicate whether fields are indexed and stored, respectively; the *Multi-valued* column specifies if multiple field values are allowed for each document; and finally, the *Dynamic* column denotes fields that are not statically configured, but can instead be defined dynamically at index time.

In Solr, indexed fields are available for searching and, if additionally, they are not multi-valued nor tokenized (or are tokenized into a single token), we can also sort results based on their value. Tokenization is configured by the type of analysis performed on each field, which is determined by its type. Stored fields can be retrieved in result sets. If a field is indexed, but not stored, we can perform searches and result ranking based on its value, but cannot recover or present the value.

The purpose of each field and its relation to the system requirements mentioned previously are the following:

- The **id** field is the unique identifier for each sentence and results from the concatenation of the publication PMID and the sentence position in the publication title and abstract (e.g. "`16437381-3`");
- The **pmid** field holds the identifier of the publication from which the sentence was extracted (e.g. "`16437381`"). It is used to satisfy requirement 6;

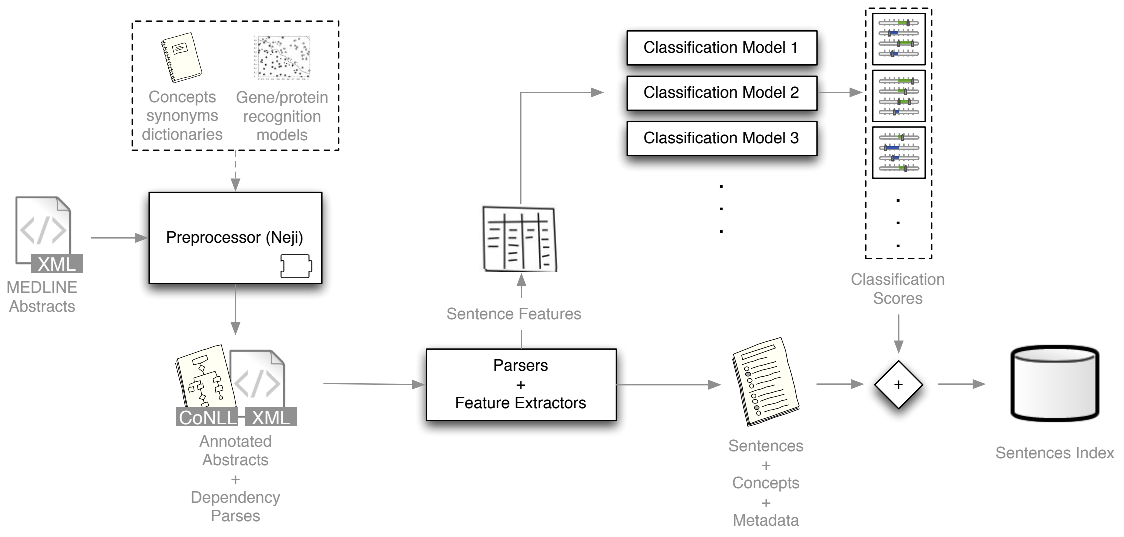
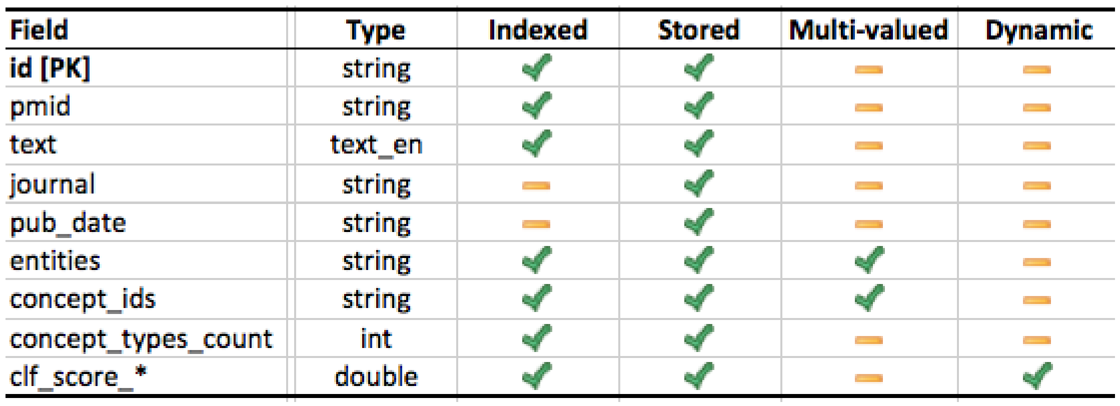


Figure 3.28: Sentence indexing workflow.

Table 3.8: Solr index schema for sentences.



Field	Type	Indexed	Stored	Multi-valued	Dynamic
id [PK]	string	✓	✓	–	–
pmid	string	✓	✓	–	–
text	text_en	✓	✓	–	–
journal	string	–	✓	–	–
pub_date	string	–	✓	–	–
entities	string	✓	✓	✓	–
concept_ids	string	✓	✓	✓	–
concept_types_count	int	✓	✓	–	–
clf_score_*	double	✓	✓	–	✓

- The **text** field contains the sentence text. It allows the use of full-text queries, in line with requirement 1. It is the only field that is analysed, by the `text_en` analyser defined in Solr. This analyser performs tokenization, removes stop words, changes all text to lower case, and finally applies Porter's stemming, the de facto stemming algorithm for English text [78]. Tokenization treats whitespace and punctuation as delimiters, discarding delimiter characters;
- The **journal** and **pub_date** fields store the name of the journal where the article was published and its publication date, respectively. Both these fields address requirement 6;
- The multi-valued **entities** field stores all the named entities with which the sentence was annotated. For each entity, it saves the entity term(s), associated concept identifier(s) and the position in the sentence where the entity occurs. This field serves two purposes, hence, it is indexed to satisfy requirement 4 and stored to serve requirement 5. First, in line with requirement 4, it is used to provide query suggestions for concepts (using the Solr Terms Component[50]). Since it holds named entities annotated with concept identifiers, we can use it to auto-complete queries with concept names. Suggestions are sorted by document frequency, meaning that concepts that occur in more sentences will appear first in the suggestions list. Second, it is used to satisfy

[50]`https://cwiki.apache.org/confluence/display/solr/The+Terms+Component`

requirement 5 by encoding the position of entities identified in sentences, along with concept identifiers. Entities are structured as "`entity terms|ENTITY_ID1;ENTITY_ID2|pos`" (e.g. "`Parkinson disease|UMLS:C0030567:T047:DISO|51`");

- The multi-valued **concept_ids** field indexes the unique concept identifiers of named entities tagged in each sentence. It supports requirement 2, allowing concept-based search. Each identifier encodes the data source from which the concept was collected, the concept identifier on that data source, the concept semantic sub-group, and semantic group (e.g. "`UMLS:C0030567:T047:DISO`");
- The **concept_types_count** field contains the number of different concept types present in each sentence. It used to satisfy requirement 3, by facilitating sentence ranking by concept diversity;
- The **clf_score_*** field is a dynamic field[51] that can be expanded into multiple fields at index time. Using this *meta-field* definition, we can index and store results of multiple classification models for each sentence, without hard-coding all of them in the schema beforehand. This way, in line with requirement 3, we gain flexibility to rank sentences using various strategies, implemented by each of the classification models we decide to integrate in the system.

Document indexing servers like Apache Solr are, as the name implies, document-oriented and geared towards efficient retrieval of documents based on field values. Our requirement 7 is intended to facilitate concept exploration, by providing users with reference information about biological concepts in external databases. This prompted the use of an additional repository to store concept metadata in a structured way.

We had previously compiled a database of concepts for use in another text mining project at the University of Aveiro Bioinformatics Group – BeCAS [6]. We used multiple meta-sources, including UMLS [79], LexEBI [63], Jochem [80], and NCBI BioSystems [81]. Data from these meta-sources was also used to generate the concept synonyms dictionaries employed for concept annotation in our indexing pipeline.

For this thesis, we reused the database of concepts compiled for the BeCAS application [6], stored in a MongoDB server. This database was compiled from multiple meta-sources, including UMLS [79], LexEBI [63], Jochem [80], and NCBI BioSystems [81]. Data from these meta-sources was also used to generate the concept synonyms dictionaries employed for concept annotation in our indexing pipeline. MongoDB is a *schema-less* database, which means that it does not enforce a strict pre-defined structure on documents, as relational databases do. Nonetheless, it stores records (in JSON-like document format) in a structured way. Figure 3.29 describes the schema according to which concept metadata is stored.

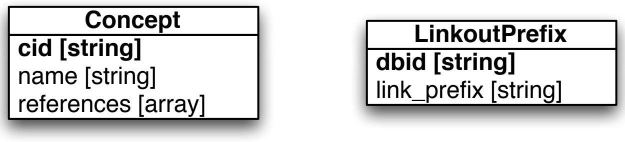


Figure 3.29: MongoDB data store schema for concepts and references.

Concepts are stored in a *concepts* collection, which contains the concept identifier for each concept, *cid* (e.g. "`UMLS:C0030567:T047:DISO`"), its preferred name (e.g. "Parkinson's Disease") and a list of references to external sources (e.g. `["NCI:C26845", "SNOMEDCT:49049000", ...]`). In order to provide hyperlinks to external sources, linkout prefixes are available in a *linkouts* collection. Each linkout prefix maps a reference source identifier, *dbid* (e.g. "`NCI`"), to the corresponding prefix used to construct hyperlinks to external databases (e.g.

[51]`https://cwiki.apache.org/confluence/display/solr/Dynamic+Fields`

http://ncit.nci.nih.gov/ncitbrowser/...&code=). This strategy facilitates updating the hyperlinks in case of changes introduced by the external sources. Collection indexes were defined on the *cid* and *dbid* fields to allow fast retrieval of records based on these unique identifiers.

Requirement 7 is not yet implemented in the current version of the PhraseRank application user interface. However, the system API already supports this requirement, providing metadata about biological concepts in a programmatic way.

3.2.5 Ranking Methods

The indexing model adopted for the proposed system was designed with the intent of integrating various sentence-ranking models, while facilitating flexible addition of new ones in a pluggable way. PhraseRank allows ranking of search results according to either one of various sentence classification scores, or based on feature values, such as measures of concept diversity.

Results ranking is enabled by Apache Solr's sorting functionality[52]. It allows ordering of result sets according to either values of indexed fields or custom functions, known as function queries[53]. Function queries based ranking can make use of a variety of mathematical, relevance and logical functions that operate on indexed values to compute sorting scores.

3.2.5.1 Classification Scores based Ranking

In section 3.1.2 we described various datasets of biomedical texts that we adapted to induce multiple classification models of informative sentences. This allowed us to train models that are more tailored to the discrimination and ranking of different types of informative sentences, based on distinct sub-domains of biomedicine or associated to particular biological events or processes.

The datasets considered were:

- **BioNLP GE**: focused on biological events occurring upon genes and gene products;
- **BioNLP EPI**: targeting events relating to epigenetic change and post-translational protein modification (PTM) events;
- **BioNLP ID**: focused on the bio-molecular mechanisms of infectious diseases;
- **Angiogenesis**: focused on biological processes related to angiogenesis.

Since we indexed a corpus of sentences related to neurodegenerative diseases, not all of the training datasets, when used individually, can be considered suitable to produce classification models applicable to our corpus. For instance, our corpus does not include sentences related to infectious diseases, so we did not use that data alone to induce a classification model usable as a ranking strategy.

After selecting the features we considered more appropriate for our task (discussed in section 3.1.6), we trained the elected classifier – a Linear SVM – on various datasets using that feature set. A model was trained on the whole aggregated data – the four datasets – and used as the default ranking strategy of the system. We then induced three additional models on each of the datasets (with the exception of ID), and integrated them in the system as separate ranking strategies. The default ranking approach – which we labelled as "automatic ranking" – should be the more appropriate for general-purpose ordering of informative sentences, without focusing on any specific type of sentences. The three additional ranking methods allow users interested in particular types of bio-events to re-rank the sentences retrieved for their query. For example, a researcher studying post-translational protein modifications could use the PTM ranking strategy (induced on the EPI dataset) to cause sentences mentioning PTM events to rank higher, thus appearing first on the results list.

We take the classification scores as measures of the informative value of each sentence, according to each model, and use them for ranking the sentences. In practice, they are the predicted confidence scores assigned to each sentence. A confidence score for a sentence is the signed distance of the sample representing the features of that sentence to the hyperplane drawn by the SVM classifier to separate sentence classes (informative vs. non-informative). This causes sentences considered informative to

[52]https://wiki.apache.org/solr/CommonQueryParameters#sort
[53]https://wiki.apache.org/solr/FunctionQuery

present positive scores, while other sentences have negative score values. The higher the *absolute* value of the classification score, the higher the probability of a sentence being correctly classified.

Classification scores assigned by the various models to each sentence are indexed and stored by Solr using the dynamic `clf_score_*` field definition, mentioned in Table 3.8. In the current version of the system, this dynamic field is materialized into 4 fields – `clf_score_all`, `clf_score_genes`, `clf_score_ptm` and `clf_score_angio`, corresponding to each of the previously mentioned classification models induced on different datasets. However, the dynamic nature of the indexing model allows addition of new ranking methods in a flexible way, without the need to update the index schema.

3.2.5.2 Feature Values based Ranking

We can argue that the semantic richness of a sentence can be considered a good measure of its informative value. Moreover, if we consider concepts to add semantic value to a sentence, one way to quantify this semantic richness is by counting the number of concepts of different types that are mentioned in the same sentence.

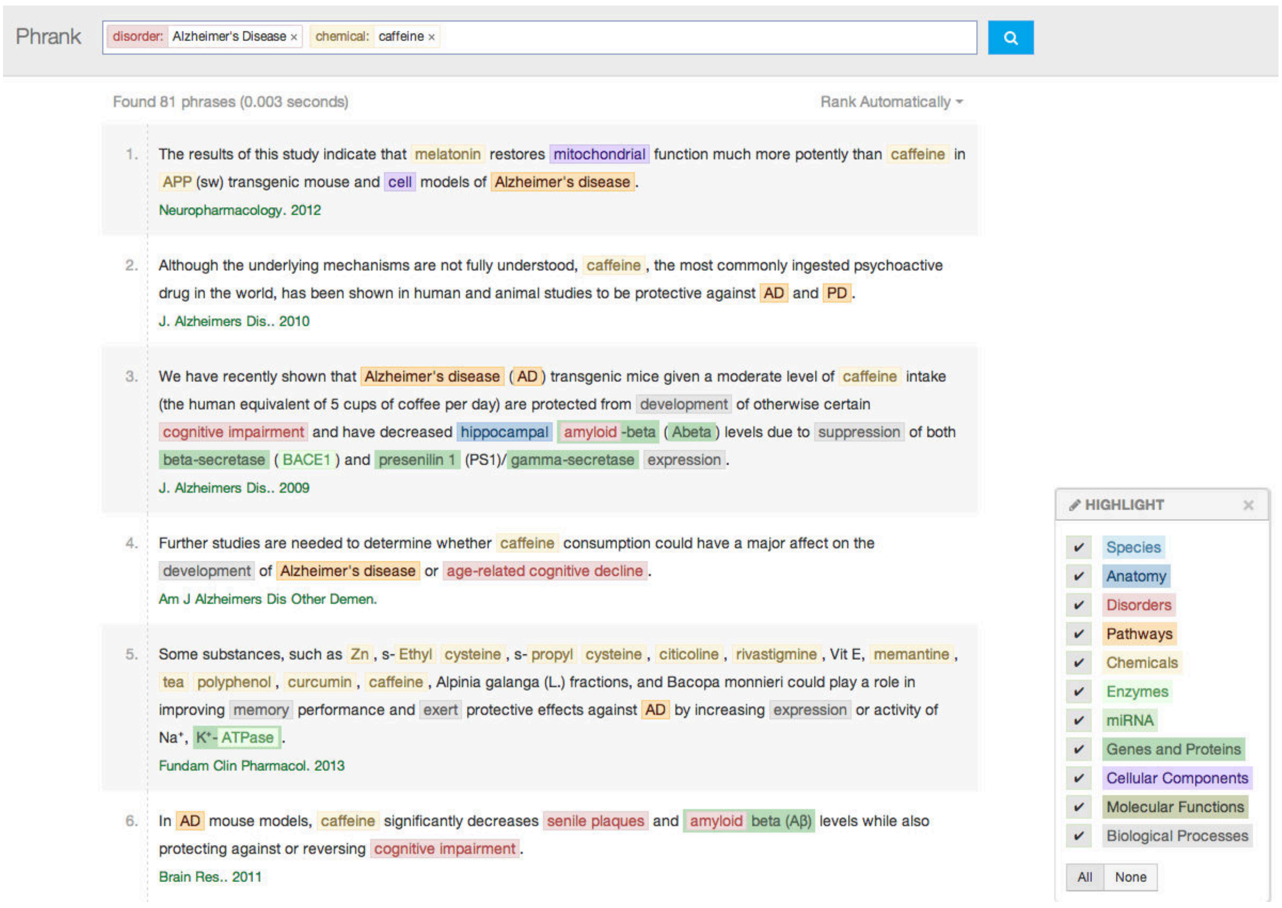
Typical user information needs are related to either the lack of knowledge about a certain concept, or the correlation between two or more concepts. Thus, ranking sentences that mention different types of concepts higher can possibly aid in the discovery of previously unknown relations between concepts.

These hypotheses prompted us to support ranking strategies based on feature values, such as the number of concepts of different types mentioned in the same sentence. The current version of PhraseRank uses the value of the `concept_types_count` field mentioned in Table 3.8 to rank sentences by a method we labelled "concept diversity".

For this thesis we only contemplated the use of the mentioned concept diversity measure as a feature value based ranking strategy. In the future, we might consider testing different feature values as ranking strategies. To facilitate this, it could be beneficial to adapt the indexing schema to use more dynamic fields of different types, instead of explicitly defining feature values. A possible approach could be the removal of the `concept_types_count` field and the addition of more dynamic fields, such as `feature_int_*`, `feature_double_*`, and maybe others. This would allow definition of several feature values of different types at index time, without the need for schema changes.

3.2.6 Search Interface

Users can interact with the PhraseRank (Phrank) system through a web-based interface, illustrated in Figure 3.30. It supports both concept-based and full-text based queries, auto-completion for concept names, multiple result ranking strategies, selective highlighting/muting of concepts per concept type, addition of concepts mentioned in result sentences to the initial query, and incremental continuous loading of results (removing the need for typical pagination strategies).


Phrank

disorder: Alzheimer's Disease × chemical: caffeine ×

Found 81 phrases (0.003 seconds)

Rank Automatically

1. The results of this study indicate that melatonin restores mitochondrial function much more potently than caffeine in APP (sw) transgenic mouse and cell models of Alzheimer's disease.
Neuropharmacology. 2012

2. Although the underlying mechanisms are not fully understood, caffeine, the most commonly ingested psychoactive drug in the world, has been shown in human and animal studies to be protective against AD and PD.
J. Alzheimers Dis.. 2010

3. We have recently shown that Alzheimer's disease (AD) transgenic mice given a moderate level of caffeine intake (the human equivalent of 5 cups of coffee per day) are protected from development of otherwise certain cognitive impairment and have decreased hippocampal amyloid -beta (Abeta) levels due to suppression of both beta-secretase (BACE1) and presenilin 1 (PS1)/gamma-secretase expression.
J. Alzheimers Dis.. 2009

4. Further studies are needed to determine whether caffeine consumption could have a major affect on the development of Alzheimer's disease or age-related cognitive decline.
Am J Alzheimers Dis Other Demen.

5. Some substances, such as Zn, s- Ethyl cysteine, s- propyl cysteine, citicoline, rivastigmine, Vit E, memantine, tea polyphenol, curcumin, caffeine, Alpinia galanga (L.) fractions, and Bacopa monnieri could play a role in improving memory performance and exert protective effects against AD by increasing expression or activity of Na⁺, K⁺- ATPase.
Fundam Clin Pharmacol. 2013

6. In AD mouse models, caffeine significantly decreases senile plaques and amyloid beta (Aβ) levels while also protecting against or reversing cognitive impairment.
Brain Res.. 2011

HIGHLIGHT

- ✓ Species
- ✓ Anatomy
- ✓ Disorders
- ✓ Pathways
- ✓ Chemicals
- ✓ Enzymes
- ✓ miRNA
- ✓ Genes and Proteins
- ✓ Cellular Components
- ✓ Molecular Functions
- ✓ Biological Processes

All None

Figure 3.30: Search interface overview.

3.2.6.1 *Query Protocol*

The system supports a query protocol that allows a combination of concept-based and free-text queries (Figure 3.31). To assist users in typing queries, suggestions are offered for multiple concepts that match a given term prefix (Figure 3.32). Suggestions are sorted by the frequency of occurrence of each matching concept in the indexed corpus, hence, more “popular” concepts are featured first in the suggested concepts list.

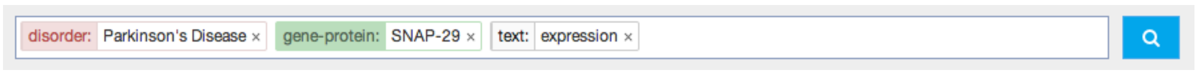

disorder: Parkinson's Disease × gene-protein: SNAP-29 × text: expression ×

Figure 3.31: Query protocol supporting a combination of concept-based and free-text queries.

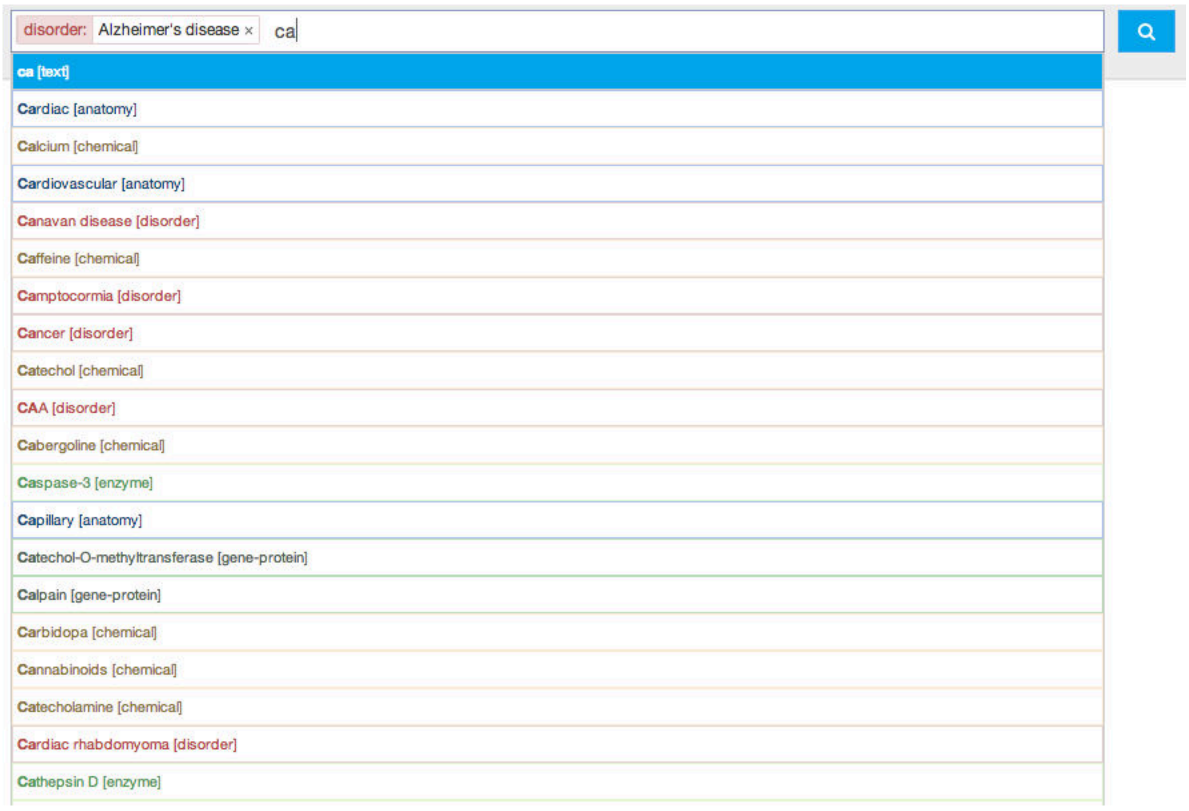


Figure 3.32: Suggestion-assisted query completion for biological concepts.

3.2.6.2 Results Ranking

Sentences retrieved for a given query can be ordered according to the various ranking strategies described in section 3.2.5. Users can select one of the available ranking methods by using a dropdown list (Figure 3.33) and results are immediately reordered.

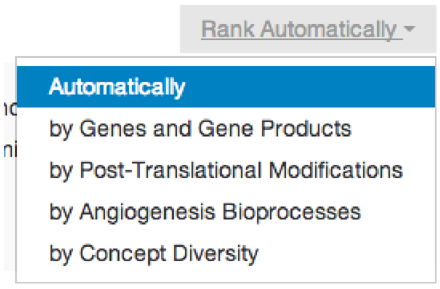
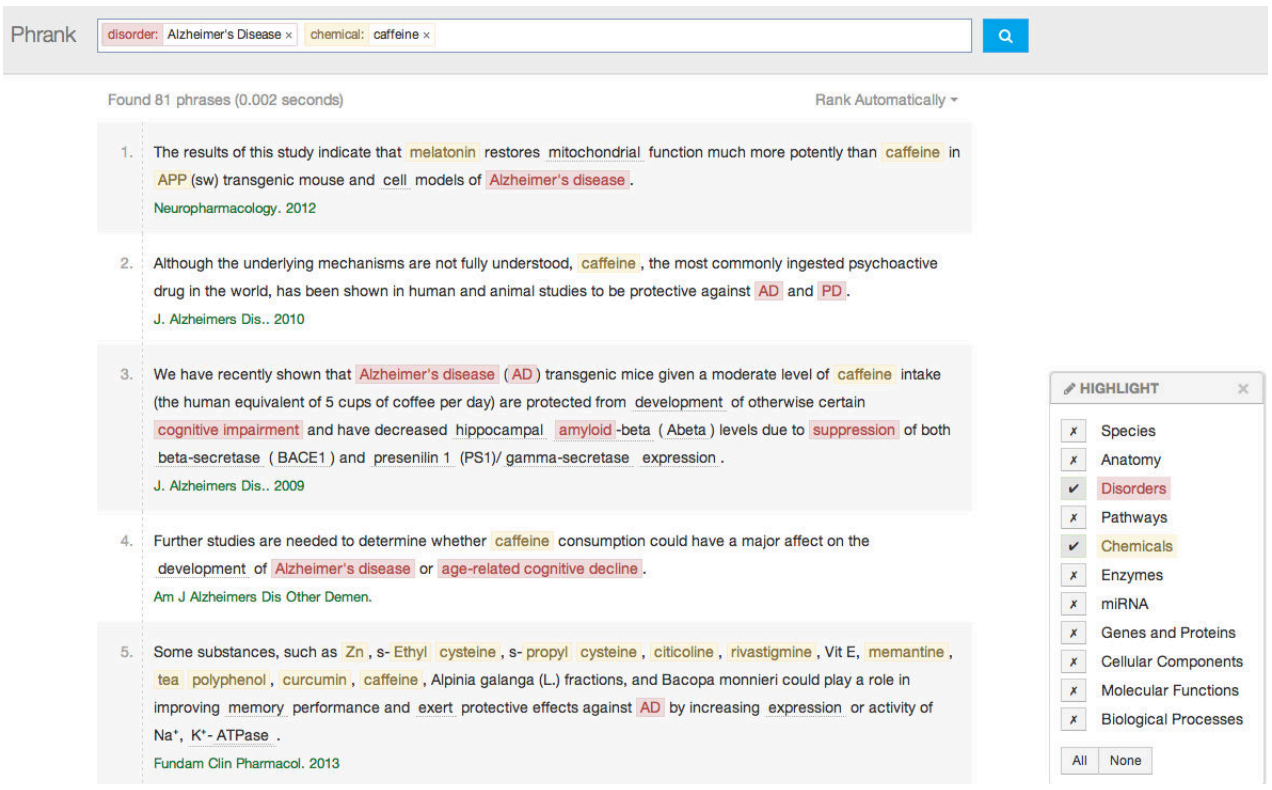

Rank Automatically
Automatically
by Genes and Gene Products
by Post-Translational Modifications
by Angiogenesis Bioprocesses
by Concept Diversity

Figure 3.33: Result ranking method selection list.

3.2.6.3 Concept Highlighting

To facilitate visual discrimination of different concepts, annotated entities are highlighted with distinct colours, according to their type (as shown in Figure 3.30). Using a set of checkboxes, users can selectively highlight and mute certain types of concepts, facilitating focusing only on concepts of a certain type (Figure 3.34). Muted concepts are underlined with a subtle dotted line, to provide a visual indication of other identified concepts. Clicking on any annotated concept allows addition of that concept to the initial query, which facilitates iterative filtering of search results by augmenting the query with more concepts.



1. The results of this study indicate that melatonin restores mitochondrial function much more potently than caffeine in APP (sw) transgenic mouse and cell models of Alzheimer's disease.
Neuropharmacology. 2012

2. Although the underlying mechanisms are not fully understood, caffeine, the most commonly ingested psychoactive drug in the world, has been shown in human and animal studies to be protective against AD and PD.
J. Alzheimers Dis.. 2010

3. We have recently shown that Alzheimer's disease (AD) transgenic mice given a moderate level of caffeine intake (the human equivalent of 5 cups of coffee per day) are protected from development of otherwise certain cognitive impairment and have decreased hippocampal amyloid-beta (Abeta) levels due to suppression of both beta-secretase (BACE1) and presenilin 1 (PS1)/ gamma-secretase expression.
J. Alzheimers Dis.. 2009

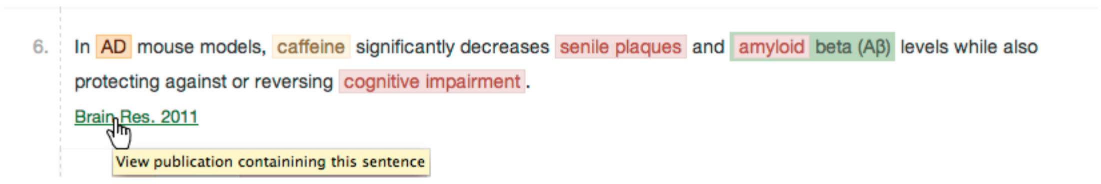
4. Further studies are needed to determine whether caffeine consumption could have a major affect on the development of Alzheimer's disease or age-related cognitive decline.
Am J Alzheimers Dis Other Demen.

5. Some substances, such as Zn, s- Ethyl cysteine, s- propyl cysteine, citicoline, rivastigmine, Vit E, memantine, tea polyphenol, curcumin, caffeine, Alpinia galanga (L.) fractions, and Bacopa monnieri could play a role in improving memory performance and exert protective effects against AD by increasing expression or activity of Na^+, K^+- ATPase.
Fundam Clin Pharmacol. 2013

Figure 3.34: Selective concept highlighting and muting per concept type.

3.2.6.4 References to Source Publications

All sentences presented in search results include a reference to the journal in which the corresponding article was published, along with the publication year, if available. By clicking on that reference, the user is redirected to a web application (BeCAS[54]) where he can read the whole abstract, also annotated with biological concepts (Figure 3.35).



6. In AD mouse models, caffeine significantly decreases senile plaques and amyloid beta (Aβ) levels while also protecting against or reversing cognitive impairment.
Brain Res. 2011

Figure 3.35: References to source publications in search results.

[54]http://bioinformatics.ua.pt/becas/pmid/21907331

3.2.7 Search API

As mentioned when describing the overall system architecture (section 3.2.1), the system features are exposed through an HTTP REST API, facilitating integration in third-party tools. The API offers methods to perform queries (with integrated ranking), request concept suggestions and resolve concept references. All the API endpoints described in the following sections respond to the HTTP GET verb, with parameters specified as URL arguments.

3.2.7.1 Querying Sentences

In order to perform queries, API clients, such as the application user interface, execute HTTP GET requests to the search endpoint, exposed at the address `http://`*`phrank`*`/api/query/sentences`.

Table 3.9 describes the parameters accepted by the search endpoint. The query string is specified using the `q` parameter as a comma delimited list of query tokens. Each token is composed by a prefix, followed by a colon and finally the search terms, which can be either concept identifiers or words. Free-text tokens take the form "`text:query terms`". Concept-based tokens follow the format "`id:CONCEPT_IDENTIFIER`". Multiple tokens of either type can be combined in the same query, using commas as delimiters.

The optional parameters `start` and `limit` are used to provide pagination over query results. The ranking strategy used to sort sentences is specified using the optional `s` parameter. Its value should be one of "`auto`", "`genes`", "`ptm`", "`angio`" or "`cdiversity`", depending on the desired ranking method (described in section 3.2.5).

Search results are returned in JSON format. The results are based on the response returned by the Apache Solr server and augmented with additional parameters, such as concepts metadata and details about the request. A sample response for a concept query about Alzheimer's disease is included in Listing 5 (Appendix C).

Table 3.9: Search API Parameters

Parameter	**Type**	**Default**	**Description**
q	string	-	Query string
limit	int	10	Maximum number of sentences to retrieve
start	int	0	Offset from the first result
s	string	auto	Results ranking strategy

3.2.7.2 Concept Suggestions

The query auto-complete feature implemented in the user interface is assisted by an API method for concept suggestions. It is available at an endpoint of the form `http://`*`phrank`*`/api/suggest/concepts`. API clients specify a concept prefix with a minimum of two characters using the `q` parameter and a list of matching concepts is returned (sample in Listing 6, Appendix C). The maximum number of suggestions can be limited using the `limit` parameter.

Table 3.10: Concept Suggestions API Parameters

Parameter	**Type**	**Default**	**Description**
q	string	-	Query string
limit	int	20	Maximum number of suggestions to retrieve

3.2.7.3 Resolving Concept References

Search results returned by the API can contain external references (described in section 3.2.4) for known biomedical concepts identified in sentences. These references can be resolved by executing an HTTP GET request to the address `http://`*`phrank`*`/api/concept-reference/redirect/`*`REF`*, where *`REF`* is the concept reference. This API method then responds with an HTTP redirect with status code 303, containing the resolved address of the concept external reference in the `Location` header of the HTTP response.

3.2.8 Summary

In the second part of this chapter we described how the sentence classification methods proposed in section 3.1 were used for sentence ranking in an information retrieval system for biomedical sentences.

The use of a flexible sentence-indexing model (section 3.2.4) allows us to integrate various ranking strategies for query results, based on information content predictions made by pluggable text classification methods. This permits tailoring of result sets to match expectations of users with different information needs.

Five different ranking strategies were integrated in the sentence-based information retrieval system (section 3.2.5). Two general-purpose methods, aiming to discriminate informative sentences from biomedical research articles regardless of sub-domain, and three focused strategies, which target specific types of informative sentences. One of the general-purpose strategies and the three focused ones result from automatic text classification methods. The other general-purpose ranking approach is based on the notion of concept diversity.

The proposed information retrieval system leverages the use of state-of-the-art concept identification tools to semantically index sentences with associated metadata about biological concepts. This allows concept-based queries – assisted by auto-complete suggestions – and visual tagging of named entities in the search interface (section 3.2.6).

Sentence retrieval and ranking functionalities, as well as concept suggestions and external referencing features, are exposed programmatically as an HTTP API (section 3.2.7).

Empiric system evaluation showed that the different ranking methods seem to produce expected results at correctly discriminating different types of informative sentences. However, it would be desirable to submit the system to thorough validation by domain experts, to accurately assess the quality of the proposed retrieval and ranking methods.

Chapter Four

Conclusions

"If we knew what we were doing it wouldn't be called research, would it?"
— Albert Einstein

The main targets of research for this thesis were the identification of features of informative sentences in the biomedical domain, and the development of an information retrieval approach that allows seeking relevant sentences from the literature, while facilitating results ranking based on user information needs.

We first conducted exploratory work for the identification and evaluation of various lexical, syntactic, grammatical and semantic features of text that could potentially be used for discrimination of informative sentences. We concluded that one of the simplest models for text classification – the bag-of-words – can be used to identify informative sentences with satisfactory results. Since this model is based on lexical features of sentences (words), it works best when training data has a vocabulary aligned with the target data for classification. Our results suggest that the use of n-grams yields better results than plain bags-of-words. We found trigrams to produce best results for our data. The use of TF-IDF weighting techniques on top of bag-of-words and n-grams improves classification precision, but can have a negative effect on recall.

Grammatical features such as part-of-speech (POS) and chunk tags seem to be good at producing classification models that better generalize to types of text distinct from the text used to induce the models. Namely, they improve recall and AUC scores, meaning they are good choices for use in ranking models, which value correct classification of positive instances higher than negative instances. However, these models present lower precision than pure lexical models, such as the previously mentioned bags-of-words and n-grams models.

Other types of features considered, like presence of biologically relevant verbs; presence of biological concepts and concept diversity; and syntactic features related to existence of dependencies between verbs and concepts, failed to produce satisfactory results when used in isolation, but showed performance similar or better than bags-of-words for model generalization.

One of the challenges we had to tackle was the inexistence of a gold-standard corpus for key sentence or fact extraction in the biomedical domain. For this, we adapted annotated corpora produced for event-extraction tasks in order to support the use of supervised learning techniques. Even though this data cannot be considered of excellent quality, it allowed us to conduct our task and is now available for use in future research. Curation of this automatically produced resource by domain experts would certainly improve its quality and consequently our results.

We developed an automated framework for the evaluation of multiple classification models on arbitrary corpora that respects a certain data format. Hence, future availability of higher quality annotated resources could immediately allow us to produce better models with minimal work.

From several algorithms tested for text classification, we found that SVMs with linear kernels produced better results for our task, followed closely by SGD based methods. Feature standardization techniques like scaling and normalization reduced training times for all classifiers. Additionally, they significantly improved precision of all methods, except linear SVMs and Naïve Bayes algorithms, albeit at the expense of considerable recall losses. Since the effectiveness of the elected linear SVM was negatively affected by feature standardization we opted to not apply scaling nor normalization to feature values.

Users with different backgrounds and distinct research interests seek different types of information and have disparate notions of what they consider an informative sentence. The use of a flexible sentence indexing model allowed us to implement an information retrieval approach that supports ranking of sentences according to custom user requirements. We adapted various collections of text focusing on different sub-domains of biomedicine and molecular biology and produced several classification models that try to address different information needs. Additionally, we used the aggregated corpora to infer a general-purpose ranking method with the intent of classifying any sentence from a biomedical article as being informative or not, while attempting to quantify its information content.

Sentences that mention different types of concepts usually describe some known or suspected relationship between these concepts. By leveraging methods for the automatic identification of biological concepts in text, we can find sentences that are semantically rich, in the sense that they imply correlations among concepts.

The sentence classification models and indexing workflow devised for our proposed system resulted in a final application capable of assisting users in biomedical fact discovery and concept exploration. Empiric system evaluation suggests that top-ranking sentences for various exemplary queries are indeed informative and can serve as a starting point for deeper literature reviewing. Concept-based queries usually produce results that can act as summaries of information about the related concepts. However, the system was not evaluated by domain experts, and so it would be desirable to conduct thorough user tests by specialists for proper system validation.

We used several freely available open-source libraries to support the research methods presented in this thesis. Since we found software bugs and improved the performance of a third-party component, we shared our contributions with the community, namely to the scikit-learn machine learning library[1][2].

4.1 Future Directions

It was challenging to combine multiple features of different natures in a single classification model with improved performance, but some features proved to work better than others for generalization purposes. For this reason, a possible interesting approach for future work could be the use of ensemble methods that combine various classifiers induced on different feature sets. Each individual classification score could then be weighted into a final combined score.

Given that we developed an automated pipeline for evaluation of multiple classification models using different feature sets, whenever gold-standard annotated corpora for biomedical fact extraction is available we can test our methods on new data, and try to generate more accurate models.

The semantic indexing model adopted indexes sentences along with concept metadata. This characteristic could be exploited to permit clustering of query results by concepts, or concept types.

Regarding the identification and evaluation of features of informative sentences, several new features could be tested in different combinations. There are also a multitude of syntactic features that could be derived from dependency parses of sentences. For instance, we could extend our syntactic arcs approach with features that model arcs between two concepts, possibly with a biologically relevant verb between them.

Some requirements were not met during the development of the application and should be the main target of attention when continuing its construction. The main requirement that lacks in the final result is the ability to visualize information about concepts annotated in sentences, such as preferred names and external references. This would allow users to perform concept exploration more easily, by providing direct ways to obtain more information about concepts from external reference databases.

The query protocol implemented in the current version of the system performs logical conjunction of query terms. This way only sentences that contain all the query terms (either as annotated concepts or text) are returned in result sets. It would be desirable to augment this query protocol with the possibility to perform full Boolean queries, allowing conjunction, disjunction and negation of composite query expressions.

[1]https://github.com/scikit-learn/scikit-learn/pull/1428
[2]https://github.com/scikit-learn/scikit-learn/pull/1429

For the purpose of this thesis we indexed a small corpus of neurodegenerative diseases, suitable for empiric evaluation of system performance. However, we intend to classify and index the whole MEDLINE collection, in order to provide searching capabilities over the largest collection of biomedical articles publicly available.

Finally, we intend to publish the outcomes of our research that resulted in the sentence classification and indexing models presented here.

Appendix A

Sentence Retrieval Systems Comparison

This appendix contains a comparison of information retrieval systems for the biomedical domain that return results at the sentence level.

Table A.1: Comparison of sentence-based information retrieval systems for the biomedical domain.

System	Domain	Query Type	Results Type	Approach to Sentence Retrieval/Ranking	Ranking Options	Query Auto-Complete	Results Export	Results Highlighting	External References	Last Content Update	Major features
iHOP	Genes/Proteins	Gene synonyms or Accession numbers (one gene only)	List of sentence; Gene profile	Experimental interaction evidence	Relevance; Date; PMID	–	–	Genes/Proteins; Chemicals; Trigger verbs; Mesh terms	PubMed; Interaction evidence DBs	Current	Network metaphor to navigate between sentences using protein hyperlinks
Chilibot	Genes/Proteins	Gene/protein synonyms (two, many, or two lists)	List of sentences; Genes/proteins relationships maps	Query term co-occurrence; rule-based sentence classification	None	–	–	Query keywords	PubMed	April 2007	Classifies relationships; Hypothetical relationship generation
MEDIE	MEDLINE	Predicate argument structures [(subject, verb, object) triplets]	List of sentences; List of abstracts; Table of predicate argument structures	Predicate argument structures	Relevance; Date	✓	–	Subject; Verb; Object; Gene; Disease	PubMed	October 2009	Extracting text fragments matching queried semantics
Textpresso	Neuroscience	Keywords + Categories	List of abstracts with matching sentences	Number of matches in the document	–	–	✓	Query keywords	PubMed	May 2009	Ontology based search
BioIE	MEDLINE	Keywords	List of abstracts from which then are extracted sentences (2-step)	Purpose built templates	–	–	–	Query keywords; Matched template keywords	–	No content. Uses MEDLINE API	Custom template extraction
PhraseRank	Currently Neurodegenerative Diseases (Aimed at MEDLINE)	Concept-based + Keywords	List of sentences	Hybrid (Automatic classification models + Sentence features based)	Estimated relevance; Genes; PTM; Angiogenesis bioprocesses; Concept diversity	✓	✓	Species; Anatomy; Disorders; Pathways; Chemicals; Enzymes; miRNA; Genes/Proteins; Cellular Components; Molecular Functions; Bioprocesses	PubMed; Annotated abstract; Concept DBs (Currently API only)	Current	Support of flexible ranking models based on pluggable scoring functions; Interactive concept highlighting

Appendix B

Classification Model Results

This appendix contains additional results from the classification model evaluation not included in section 3.1, but referenced there.

Icons in tables are visual indicators to help compare performance values to the baseline results (Table 3.7), unless explicitly noted that the comparison is against other reference scores. A green upward arrow indicates a performance measure of a given evaluation yielded results better than the corresponding baseline value. A red downward arrow indicates that value is lower than the corresponding baseline value. Cells highlighted in yellow mark that value as the best of its column. Depending on the metric of each column, the best result can be either the highest or lowest value.

B.1 Classifiers Comparison

These results are discussed in section 3.1.5.

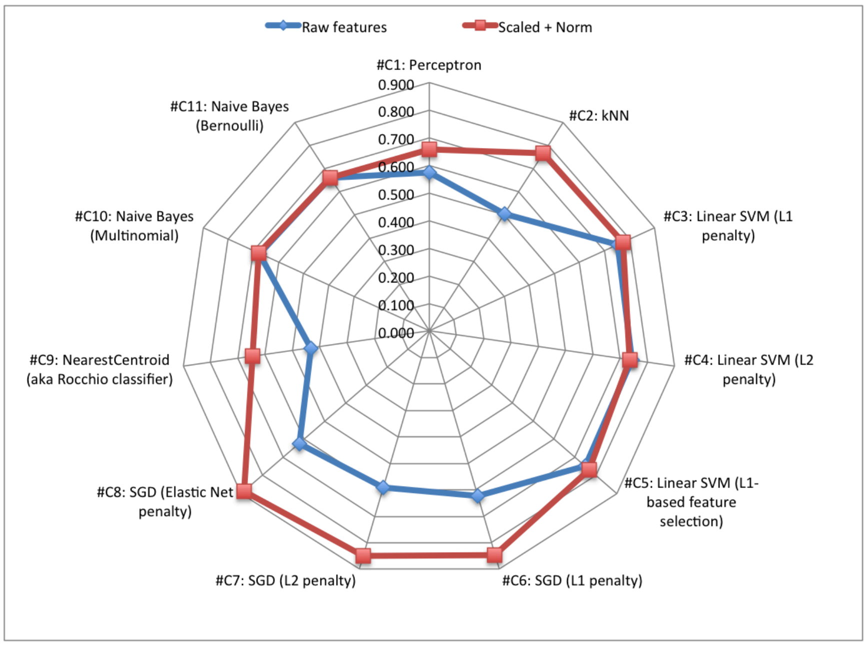
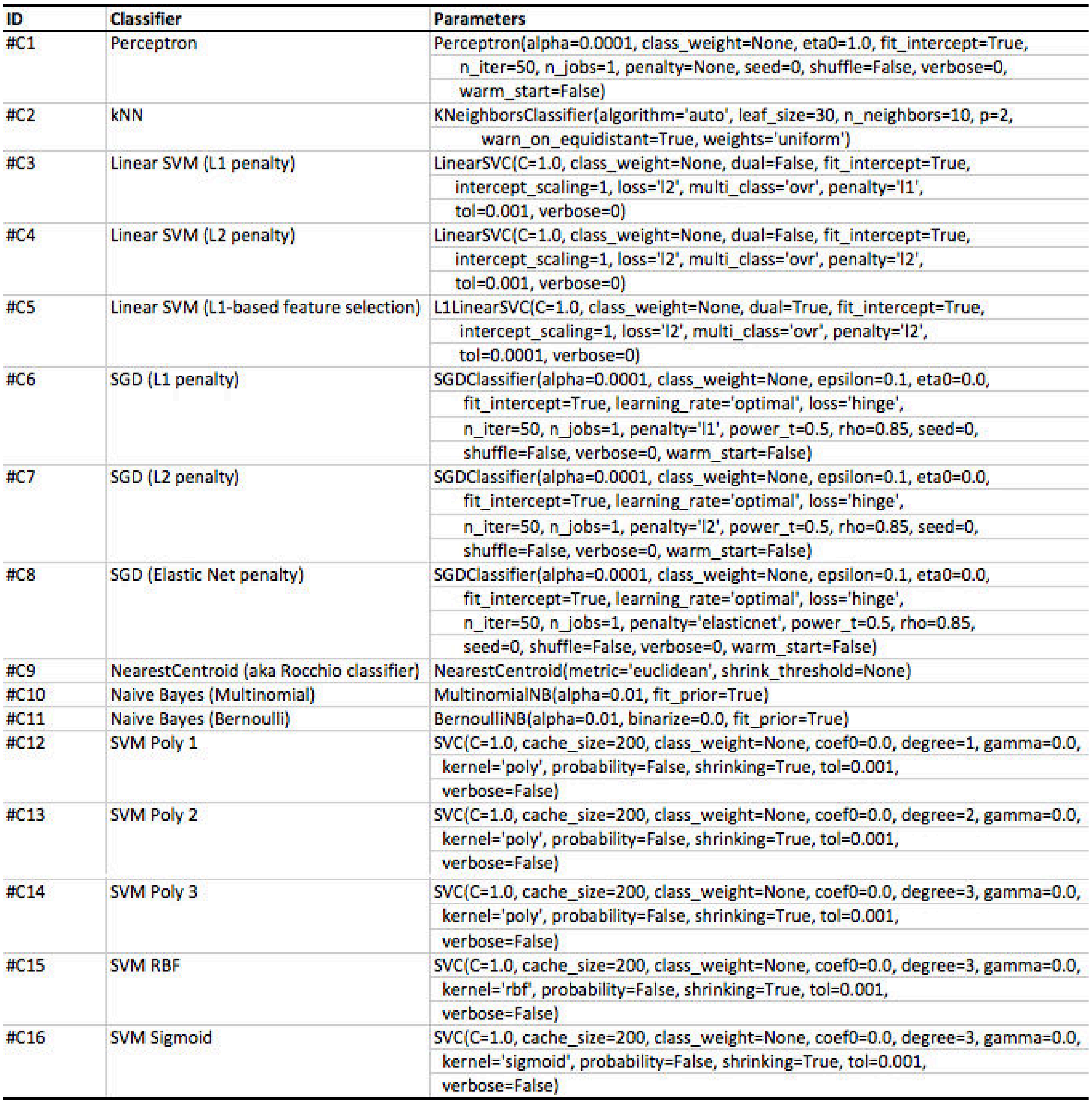


Figure B.1: Precision scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

Table B.1: Candidate classifiers evaluated. Details about each parameter can be consulted online in the scikit-learn class and function reference (`http://scikit-learn.org/0.12/modules/classes.html`).



ID	Classifier	Parameters
#C1	Perceptron	Perceptron(alpha=0.0001, class_weight=None, eta0=1.0, fit_intercept=True, n_iter=50, n_jobs=1, penalty=None, seed=0, shuffle=False, verbose=0, warm_start=False)
#C2	kNN	KNeighborsClassifier(algorithm='auto', leaf_size=30, n_neighbors=10, p=2, warn_on_equidistant=True, weights='uniform')
#C3	Linear SVM (L1 penalty)	LinearSVC(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, loss='l2', multi_class='ovr', penalty='l1', tol=0.001, verbose=0)
#C4	Linear SVM (L2 penalty)	LinearSVC(C=1.0, class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, loss='l2', multi_class='ovr', penalty='l2', tol=0.001, verbose=0)
#C5	Linear SVM (L1-based feature selection)	L1LinearSVC(C=1.0, class_weight=None, dual=True, fit_intercept=True, intercept_scaling=1, loss='l2', multi_class='ovr', penalty='l2', tol=0.0001, verbose=0)
#C6	SGD (L1 penalty)	SGDClassifier(alpha=0.0001, class_weight=None, epsilon=0.1, eta0=0.0, fit_intercept=True, learning_rate='optimal', loss='hinge', n_iter=50, n_jobs=1, penalty='l1', power_t=0.5, rho=0.85, seed=0, shuffle=False, verbose=0, warm_start=False)
#C7	SGD (L2 penalty)	SGDClassifier(alpha=0.0001, class_weight=None, epsilon=0.1, eta0=0.0, fit_intercept=True, learning_rate='optimal', loss='hinge', n_iter=50, n_jobs=1, penalty='l2', power_t=0.5, rho=0.85, seed=0, shuffle=False, verbose=0, warm_start=False)
#C8	SGD (Elastic Net penalty)	SGDClassifier(alpha=0.0001, class_weight=None, epsilon=0.1, eta0=0.0, fit_intercept=True, learning_rate='optimal', loss='hinge', n_iter=50, n_jobs=1, penalty='elasticnet', power_t=0.5, rho=0.85, seed=0, shuffle=False, verbose=0, warm_start=False)
#C9	NearestCentroid (aka Rocchio classifier)	NearestCentroid(metric='euclidean', shrink_threshold=None)
#C10	Naive Bayes (Multinomial)	MultinomialNB(alpha=0.01, fit_prior=True)
#C11	Naive Bayes (Bernoulli)	BernoulliNB(alpha=0.01, binarize=0.0, fit_prior=True)
#C12	SVM Poly 1	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=1, gamma=0.0, kernel='poly', probability=False, shrinking=True, tol=0.001, verbose=False)
#C13	SVM Poly 2	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=2, gamma=0.0, kernel='poly', probability=False, shrinking=True, tol=0.001, verbose=False)
#C14	SVM Poly 3	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=3, gamma=0.0, kernel='poly', probability=False, shrinking=True, tol=0.001, verbose=False)
#C15	SVM RBF	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=3, gamma=0.0, kernel='rbf', probability=False, shrinking=True, tol=0.001, verbose=False)
#C16	SVM Sigmoid	SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0, degree=3, gamma=0.0, kernel='sigmoid', probability=False, shrinking=True, tol=0.001, verbose=False)

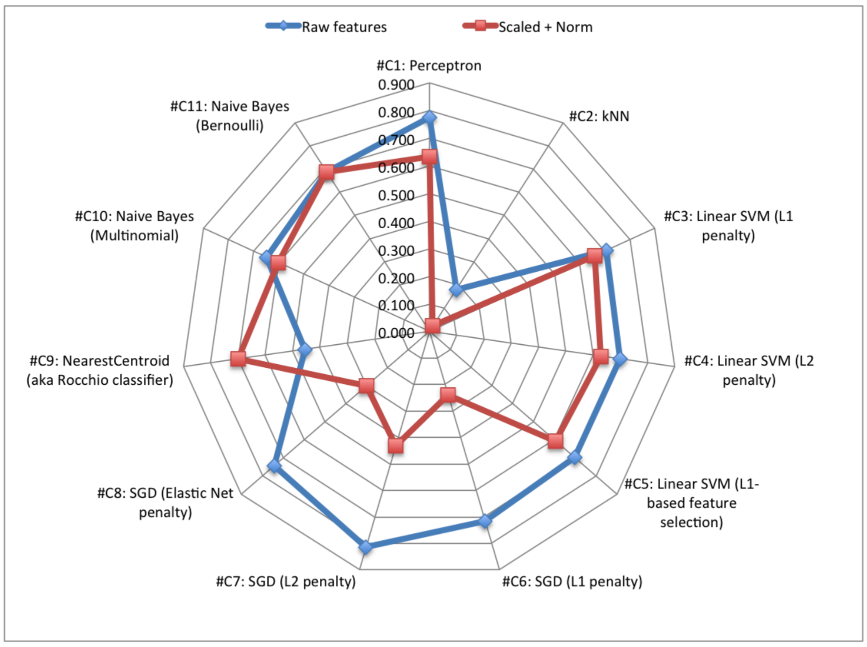


Figure B.2: Recall scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

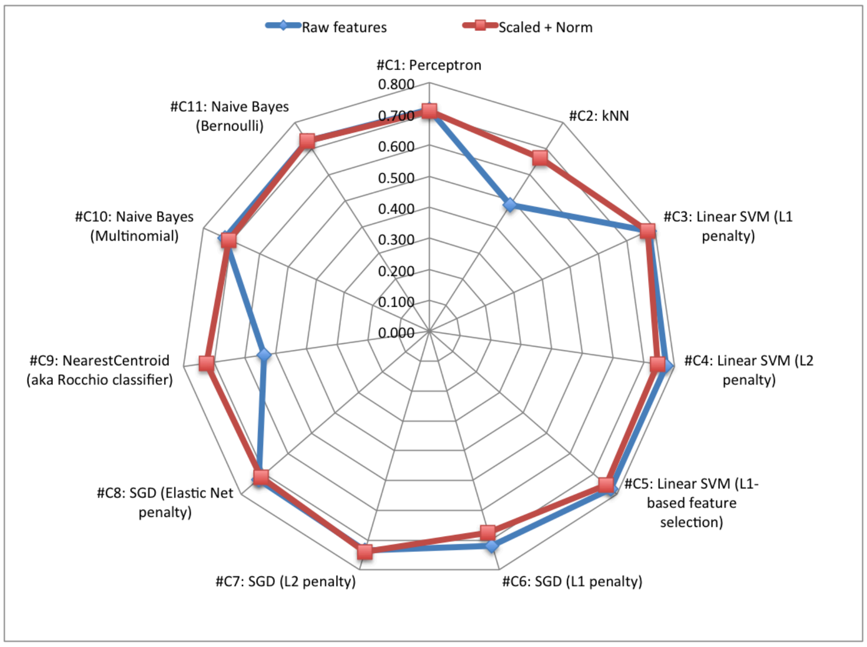


Figure B.3: Average Precision scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

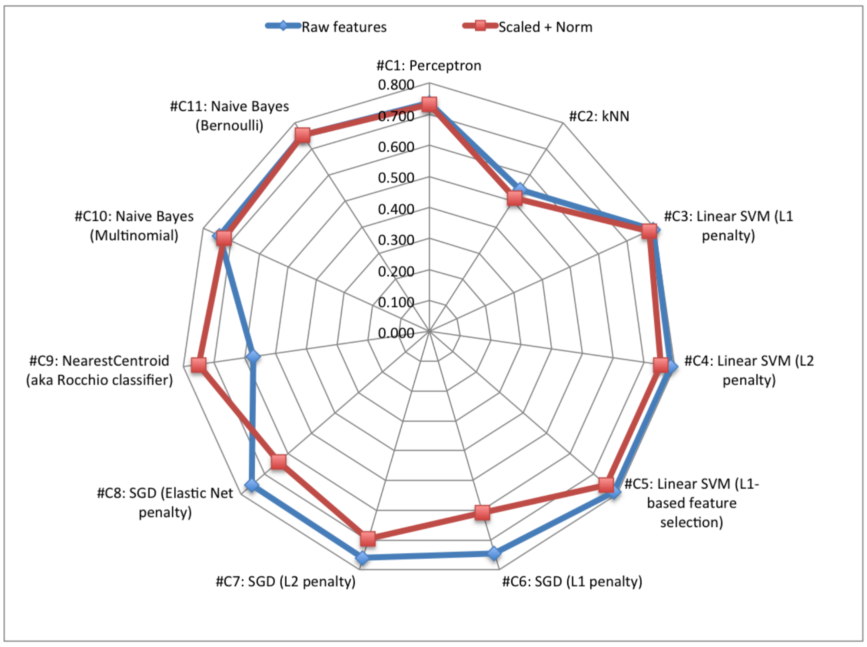


Figure B.4: AUC scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

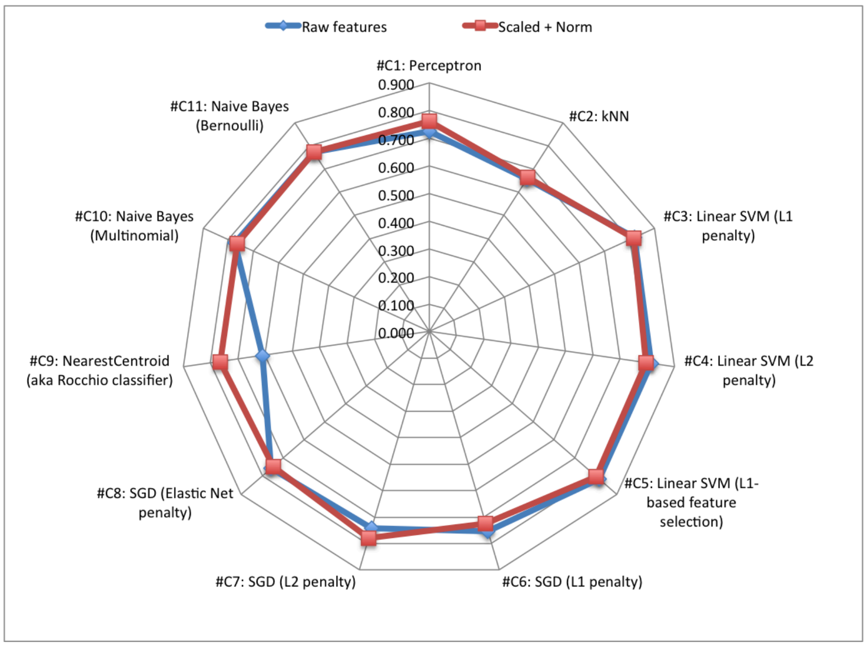
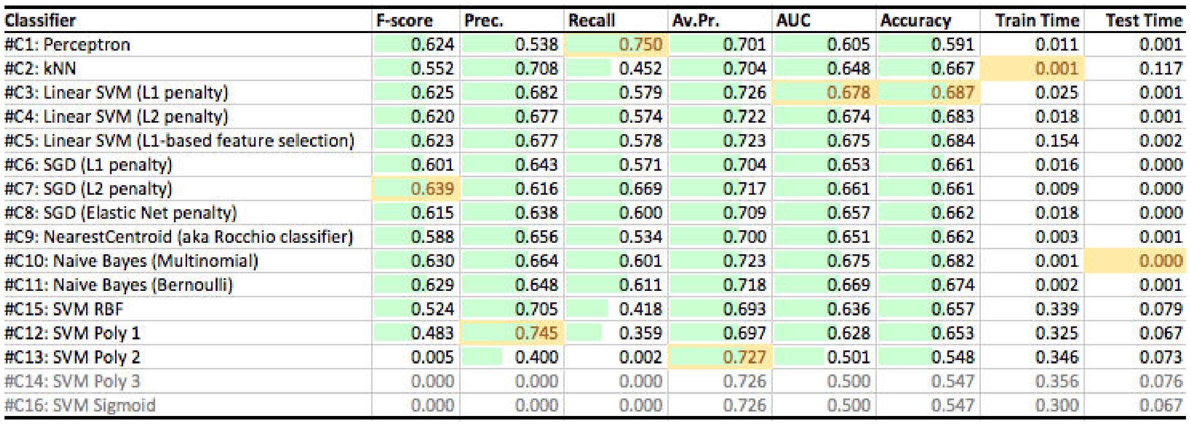


Figure B.5: Accuracy scores comparison for multiple classifiers on large feature set using CV[BioNLP+Angio] evaluation.

Table B.2: Classifiers comparison using reduced feature set of only biological concepts, relevant verbs and syntactic arcs using CV[Angio] evaluation.



Classifier	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy	Train Time	Test Time
#C1: Perceptron	0.624	0.538	0.750	0.701	0.605	0.591	0.011	0.001
#C2: kNN	0.552	0.708	0.452	0.704	0.648	0.667	0.001	0.117
#C3: Linear SVM (L1 penalty)	0.625	0.682	0.579	0.726	0.678	0.687	0.025	0.001
#C4: Linear SVM (L2 penalty)	0.620	0.677	0.574	0.722	0.674	0.683	0.018	0.001
#C5: Linear SVM (L1-based feature selection)	0.623	0.677	0.578	0.723	0.675	0.684	0.154	0.002
#C6: SGD (L1 penalty)	0.601	0.643	0.571	0.704	0.653	0.661	0.016	0.000
#C7: SGD (L2 penalty)	0.639	0.616	0.669	0.717	0.661	0.661	0.009	0.000
#C8: SGD (Elastic Net penalty)	0.615	0.638	0.600	0.709	0.657	0.662	0.018	0.000
#C9: NearestCentroid (aka Rocchio classifier)	0.588	0.656	0.534	0.700	0.651	0.662	0.003	0.001
#C10: Naive Bayes (Multinomial)	0.630	0.664	0.601	0.723	0.675	0.682	0.001	0.000
#C11: Naive Bayes (Bernoulli)	0.629	0.648	0.611	0.718	0.669	0.674	0.002	0.001
#C15: SVM RBF	0.524	0.705	0.418	0.693	0.636	0.657	0.339	0.079
#C12: SVM Poly 1	0.483	0.745	0.359	0.697	0.628	0.653	0.325	0.067
#C13: SVM Poly 2	0.005	0.400	0.002	0.727	0.501	0.548	0.346	0.073
#C14: SVM Poly 3	0.000	0.000	0.000	0.726	0.500	0.547	0.356	0.076
#C16: SVM Sigmoid	0.000	0.000	0.000	0.726	0.500	0.547	0.300	0.067

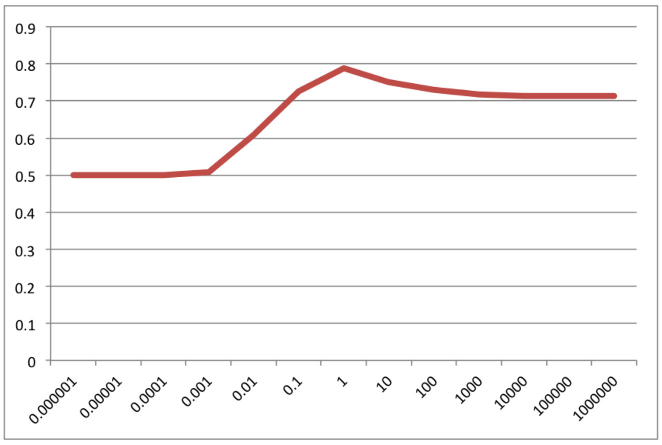


Figure B.6: Effect of C parameter on AUC score of Linear SVM with L1 penalization using CV[BioNLP+Angio] evaluation.

B.2 N-Grams

These results are discussed in section 3.1.6.2.

Table B.3: Classification performance results for bags of bigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.729	▲ 0.755	▲ 0.705	▲ 0.781	▲ 0.793	▲ 0.820
CV[GE]	▲ 0.777	▲ 0.793	▲ 0.762	▲ 0.827	▲ 0.811	▲ 0.819
CV[EPI]	▲ 0.713	▲ 0.791	▼ 0.651	▲ 0.752	▲ 0.806	▲ 0.905
CV[ID]	▲ 0.783	▲ 0.822	▼ 0.749	▲ 0.834	▲ 0.823	▲ 0.840
CV[Angio]	▲ 0.754	▲ 0.780	▲ 0.730	▲ 0.816	▲ 0.779	▲ 0.784
TT[BioNLP/Angio]	▼ 0.213	▲ 0.799	▼ 0.123	▲ 0.659	▼ 0.549	▼ 0.589
TT[GE+EPI+Angio/ID]	▲ 0.261	▲ 0.676	▼ 0.162	▲ 0.580	▲ 0.556	▲ 0.647
TT[GE+ID+Angio/EPI]	▼ 0.317	▼ 0.278	▼ 0.369	▼ 0.381	▼ 0.578	▲ 0.712
TT[EPI+ID+Angio/GE]	▼ 0.328	▲ 0.669	▼ 0.217	▲ 0.605	▲ 0.571	▲ 0.632
TT[Average]	▼ 0.280	▲ 0.605	▼ 0.218	▲ 0.556	▼ 0.564	▲ 0.645

Table B.4: Classification performance results for bags of trigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.731	▲ 0.767	▲ 0.699	▲ 0.785	▲ 0.794	▲ 0.824
CV[GE]	▲ 0.783	▲ 0.803	▲ 0.765	▲ 0.833	▲ 0.816	▲ 0.825
CV[EPI]	▲ 0.713	▲ 0.802	▼ 0.643	▲ 0.755	▲ 0.804	▲ 0.906
CV[ID]	▲ 0.779	▲ 0.818	▼ 0.744	▲ 0.830	▲ 0.820	▲ 0.838
CV[Angio]	▲ 0.761	▲ 0.790	▲ 0.737	▲ 0.823	▲ 0.787	▲ 0.791
TT[BioNLP/Angio]	▼ 0.203	▲ 0.843	▼ 0.115	▲ 0.679	▼ 0.549	▼ 0.590
TT[GE+EPI+Angio/ID]	▲ 0.263	▲ 0.696	▼ 0.162	▲ 0.590	▲ 0.559	▲ 0.650
TT[GE+ID+Angio/EPI]	▼ 0.296	▼ 0.267	▼ 0.333	▼ 0.360	▼ 0.565	▲ 0.713
TT[EPI+ID+Angio/GE]	▼ 0.316	▲ 0.664	▼ 0.207	▲ 0.600	▼ 0.567	▲ 0.629
TT[Average]	▼ 0.270	▲ 0.618	▼ 0.204	▲ 0.557	▼ 0.560	▲ 0.646

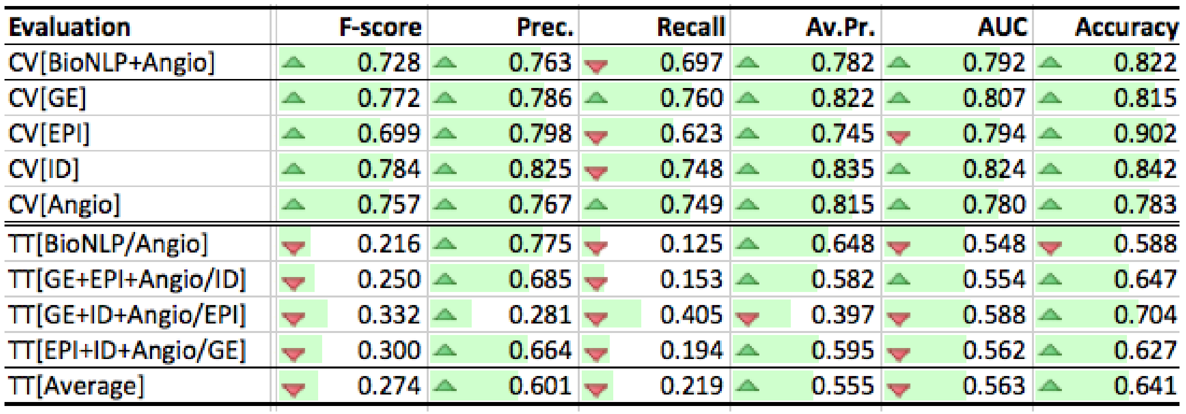
Table B.5: Classification performance results for bags of 4-grams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.730	▲ 0.766	▲ 0.698	▲ 0.784	▲ 0.793	▲ 0.823
CV[GE]	▲ 0.781	▲ 0.804	▲ 0.760	▲ 0.832	▲ 0.815	▲ 0.824
CV[EPI]	▲ 0.711	▲ 0.805	▼ 0.638	▲ 0.754	▼ 0.802	▲ 0.906
CV[ID]	▲ 0.776	▲ 0.819	▼ 0.738	▲ 0.829	▲ 0.818	▲ 0.836
CV[Angio]	▲ 0.762	▲ 0.792	▲ 0.737	▲ 0.824	▲ 0.788	▲ 0.792
TT[BioNLP/Angio]	▼ 0.200	▲ 0.841	▼ 0.113	▲ 0.678	▼ 0.548	▼ 0.589
TT[GE+EPI+Angio/ID]	▼ 0.252	▲ 0.709	▼ 0.153	▲ 0.594	▲ 0.557	▲ 0.650
TT[GE+ID+Angio/EPI]	▼ 0.295	▼ 0.269	▼ 0.328	▼ 0.359	▼ 0.565	▲ 0.716
TT[EPI+ID+Angio/GE]	▼ 0.313	▲ 0.670	▼ 0.204	▲ 0.601	▼ 0.567	▲ 0.630
TT[Average]	▼ 0.265	▲ 0.622	▼ 0.200	▲ 0.558	▼ 0.559	▲ 0.646

B.3 TF-IDF Vectors of Trigrams

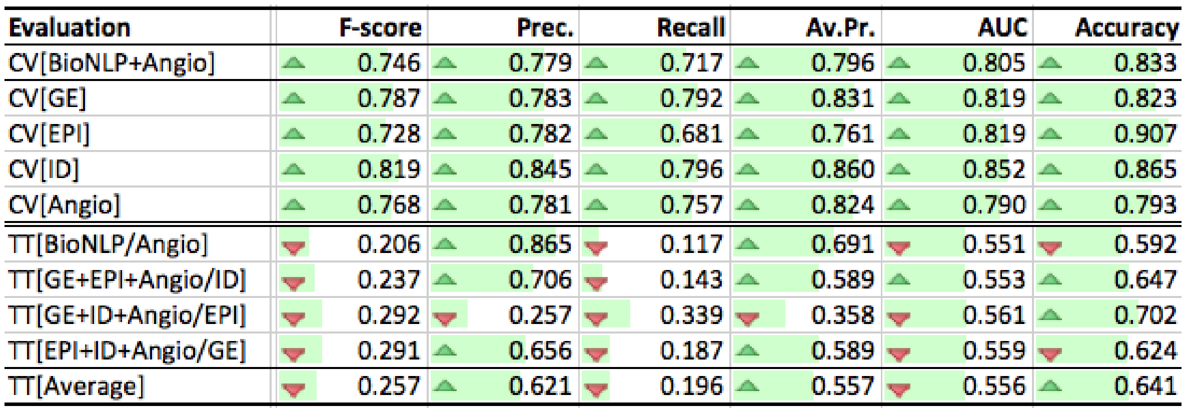
These results are discussed in section 3.1.6.3.

Table B.6: Classification performance results for TF-IDF vectors of bags-of-words.



Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.728	▲ 0.763	▼ 0.697	▲ 0.782	▲ 0.792	▲ 0.822
CV[GE]	▲ 0.772	▲ 0.786	▲ 0.760	▲ 0.822	▲ 0.807	▲ 0.815
CV[EPI]	▲ 0.699	▲ 0.798	▼ 0.623	▲ 0.745	▼ 0.794	▲ 0.902
CV[ID]	▲ 0.784	▲ 0.825	▼ 0.748	▲ 0.835	▲ 0.824	▲ 0.842
CV[Angio]	▲ 0.757	▲ 0.767	▲ 0.749	▲ 0.815	▲ 0.780	▲ 0.783
TT[BioNLP/Angio]	▼ 0.216	▲ 0.775	▼ 0.125	▲ 0.648	▼ 0.548	▼ 0.588
TT[GE+EPI+Angio/ID]	▼ 0.250	▲ 0.685	▼ 0.153	▲ 0.582	▲ 0.554	▲ 0.647
TT[GE+ID+Angio/EPI]	▼ 0.332	▲ 0.281	▼ 0.405	▼ 0.397	▼ 0.588	▲ 0.704
TT[EPI+ID+Angio/GE]	▼ 0.300	▲ 0.664	▼ 0.194	▲ 0.595	▼ 0.562	▲ 0.627
TT[Average]	▼ 0.274	▲ 0.601	▼ 0.219	▲ 0.555	▼ 0.563	▲ 0.641

Table B.7: Classification performance results for TF-IDF vectors of bags-of-trigrams.



Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.746	▲ 0.779	▲ 0.717	▲ 0.796	▲ 0.805	▲ 0.833
CV[GE]	▲ 0.787	▲ 0.783	▲ 0.792	▲ 0.831	▲ 0.819	▲ 0.823
CV[EPI]	▲ 0.728	▲ 0.782	▲ 0.681	▲ 0.761	▲ 0.819	▲ 0.907
CV[ID]	▲ 0.819	▲ 0.845	▲ 0.796	▲ 0.860	▲ 0.852	▲ 0.865
CV[Angio]	▲ 0.768	▲ 0.781	▲ 0.757	▲ 0.824	▲ 0.790	▲ 0.793
TT[BioNLP/Angio]	▼ 0.206	▲ 0.865	▼ 0.117	▲ 0.691	▼ 0.551	▼ 0.592
TT[GE+EPI+Angio/ID]	▼ 0.237	▲ 0.706	▼ 0.143	▲ 0.589	▲ 0.553	▲ 0.647
TT[GE+ID+Angio/EPI]	▼ 0.292	▼ 0.257	▼ 0.339	▼ 0.358	▼ 0.561	▲ 0.702
TT[EPI+ID+Angio/GE]	▼ 0.291	▲ 0.656	▼ 0.187	▲ 0.589	▼ 0.559	▼ 0.624
TT[Average]	▼ 0.257	▲ 0.621	▼ 0.196	▲ 0.557	▼ 0.556	▲ 0.641

B.4 TF-IDF Vectors of Trigrams, POS and Chunk Tags Combinations

These results are discussed in section 3.1.6.4. Comparison icons in tables in this section consider the results for TF-IDF vectors of bags-of-trigrams (Table B.7) as the baseline.

Table B.8: Classification performance results for TF-IDF vectors of POS tags trigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.442	▼ 0.566	▼ 0.363	▼ 0.574	▼ 0.609	▼ 0.686
CV[GE]	▼ 0.565	▼ 0.610	▼ 0.526	▼ 0.666	▼ 0.645	▼ 0.665
CV[EPI]	▼ 0.134	▼ 0.361	▼ 0.082	▼ 0.305	▼ 0.525	▼ 0.807
CV[ID]	▼ 0.455	▼ 0.510	▼ 0.411	▼ 0.574	▼ 0.582	▼ 0.622
CV[Angio]	▼ 0.639	▼ 0.664	▼ 0.617	▼ 0.727	▼ 0.679	▼ 0.685
TT[BioNLP/Angio]	▲ 0.317	▼ 0.691	▲ 0.206	▼ 0.628	▲ 0.565	▲ 0.599
TT[GE+EPI+Angio/ID]	▲ 0.273	▼ 0.516	▲ 0.186	▼ 0.508	▼ 0.538	▼ 0.619
TT[GE+ID+Angio/EPI]	▲ 0.326	▲ 0.269	▲ 0.412	▲ 0.394	▲ 0.582	▼ 0.691
TT[EPI+ID+Angio/GE]	▲ 0.331	▼ 0.580	▲ 0.232	▼ 0.564	▼ 0.557	▼ 0.613
TT[Average]	▲ 0.312	▼ 0.514	▲ 0.259	▼ 0.523	▲ 0.561	▼ 0.631

Table B.9: Classification performance results for TF-IDF vectors of POS-Chunk tag pairs trigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.476	▼ 0.564	▼ 0.411	▼ 0.589	▼ 0.623	▼ 0.689
CV[GE]	▼ 0.588	▼ 0.623	▼ 0.557	▼ 0.681	▼ 0.660	▼ 0.678
CV[EPI]	▼ 0.220	▼ 0.425	▼ 0.149	▼ 0.364	▼ 0.552	▼ 0.809
CV[ID]	▼ 0.465	▼ 0.521	▼ 0.420	▼ 0.582	▼ 0.589	▼ 0.628
CV[Angio]	▼ 0.646	▼ 0.662	▼ 0.633	▼ 0.731	▼ 0.682	▼ 0.686
TT[BioNLP/Angio]	▲ 0.373	▼ 0.717	▲ 0.252	▼ 0.654	▲ 0.585	▲ 0.616
TT[GE+EPI+Angio/ID]	▲ 0.341	▼ 0.542	▲ 0.249	▼ 0.540	▲ 0.559	▼ 0.630
TT[GE+ID+Angio/EPI]	▲ 0.340	▲ 0.264	▲ 0.480	▲ 0.419	▲ 0.591	▼ 0.663
TT[EPI+ID+Angio/GE]	▲ 0.358	▼ 0.578	▲ 0.259	▼ 0.571	▲ 0.563	▼ 0.616
TT[Average]	▲ 0.353	▼ 0.525	▲ 0.310	▼ 0.546	▲ 0.574	▼ 0.631

Table B.10: Classification performance results for the combination of TF-IDF vectors of POS tags trigrams and TF-IDF vectors of POS-Chunk tag pairs trigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.482	▼ 0.553	▼ 0.428	▼ 0.589	▼ 0.623	▼ 0.685
CV[GE]	▼ 0.593	▼ 0.619	▼ 0.569	▼ 0.683	▼ 0.661	▼ 0.677
CV[EPI]	▼ 0.240	▼ 0.390	▼ 0.174	▼ 0.357	▼ 0.557	▼ 0.801
CV[ID]	▼ 0.482	▼ 0.528	▼ 0.444	▼ 0.593	▼ 0.598	▼ 0.633
CV[Angio]	▼ 0.654	▼ 0.667	▼ 0.644	▼ 0.736	▼ 0.688	▼ 0.693
TT[BioNLP/Angio]	▲ 0.384	▼ 0.691	▲ 0.266	▼ 0.644	▲ 0.584	▲ 0.614
TT[GE+EPI+Angio/ID]	▲ 0.349	▼ 0.534	▲ 0.259	▼ 0.540	▲ 0.559	▼ 0.628
TT[GE+ID+Angio/EPI]	▲ 0.330	▼ 0.254	▲ 0.468	▲ 0.409	▲ 0.582	▼ 0.655
TT[EPI+ID+Angio/GE]	▲ 0.371	▼ 0.558	▲ 0.278	▼ 0.567	▲ 0.561	▼ 0.611
TT[Average]	▲ 0.359	▼ 0.509	▲ 0.318	▼ 0.540	▲ 0.572	▼ 0.627

Table B.11: Classification performance results for the combination of TF-IDF vectors of bags-of-trigrams, TF-IDF vectors of POS tags trigrams and TF-IDF vectors of POS-Chunk tag pairs trigrams.

Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.732	▼ 0.757	▼ 0.709	▼ 0.783	▼ 0.795	▼ 0.822
CV[GE]	▼ 0.768	▼ 0.770	▼ 0.765	▼ 0.816	▼ 0.802	▼ 0.809
CV[EPI]	▼ 0.726	▼ 0.766	▲ 0.692	▼ 0.757	▲ 0.822	▼ 0.905
CV[ID]	▼ 0.803	▼ 0.832	▼ 0.776	▼ 0.847	▼ 0.839	▼ 0.853
CV[Angio]	▼ 0.761	▼ 0.766	▼ 0.757	▼ 0.816	▼ 0.782	▼ 0.785
TT[BioNLP/Angio]	▲ 0.239	▼ 0.834	▲ 0.140	▼ 0.682	▲ 0.558	▲ 0.598
TT[GE+EPI+Angio/ID]	▲ 0.262	▼ 0.639	▲ 0.165	▼ 0.563	▲ 0.553	▼ 0.643
TT[GE+ID+Angio/EPI]	▲ 0.314	▲ 0.266	▲ 0.382	▲ 0.380	▲ 0.574	▼ 0.697
TT[EPI+ID+Angio/GE]	▲ 0.331	▲ 0.679	▲ 0.218	▲ 0.610	▲ 0.573	▲ 0.635
TT[Average]	▲ 0.286	▼ 0.605	▲ 0.226	▲ 0.559	▲ 0.565	▲ 0.643

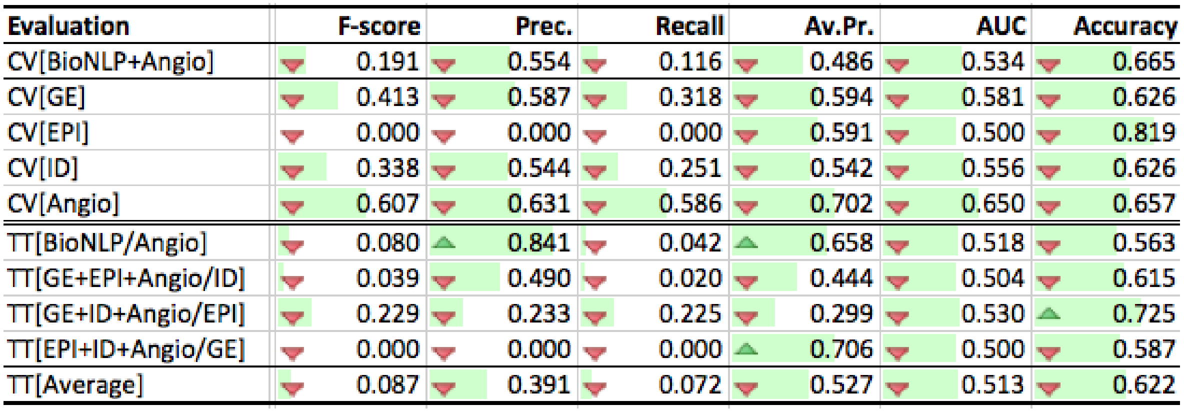
B.5 Verbs, Concepts and Syntactic Arcs

These results are discussed in section 3.1.6.5.

Table B.12: Classification performance results for biologically relevant verbs.

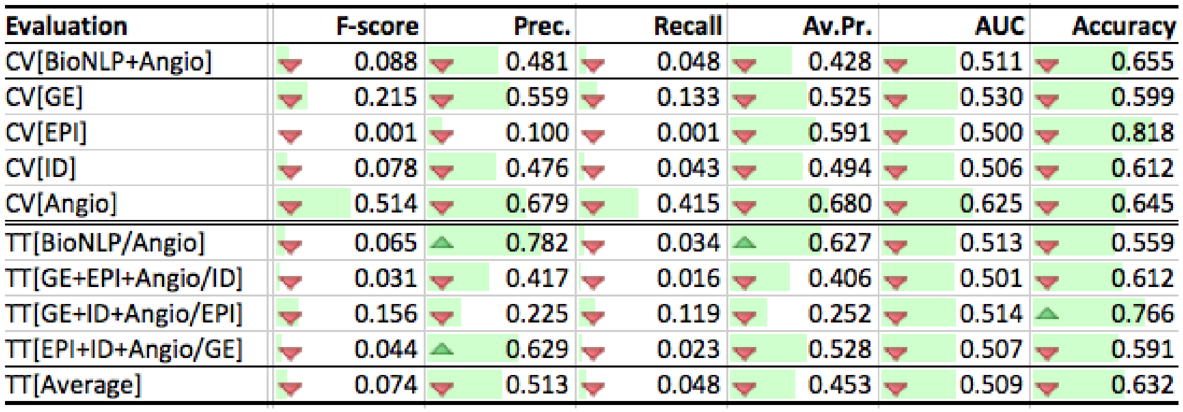
Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.357	▼ 0.558	▼ 0.263	▼ 0.537	▼ 0.577	▼ 0.675
CV[GE]	▼ 0.498	▼ 0.610	▼ 0.421	▼ 0.635	▼ 0.616	▼ 0.650
CV[EPI]	▼ 0.266	▼ 0.540	▼ 0.178	▼ 0.433	▼ 0.572	▼ 0.823
CV[ID]	▼ 0.503	▼ 0.592	▼ 0.439	▼ 0.623	▼ 0.624	▼ 0.666
CV[Angio]	▼ 0.572	▼ 0.654	▼ 0.510	▼ 0.693	▼ 0.644	▼ 0.656
TT[BioNLP/Angio]	▼ 0.270	▲ 0.728	▼ 0.166	▲ 0.636	▼ 0.557	▲ 0.594
TT[GE+EPI+Angio/ID]	▼ 0.248	▼ 0.574	▼ 0.158	▼ 0.528	▼ 0.542	▼ 0.631
TT[GE+ID+Angio/EPI]	▼ 0.273	▼ 0.229	▼ 0.340	▼ 0.344	▼ 0.543	▼ 0.672
TT[EPI+ID+Angio/GE]	▼ 0.205	▼ 0.540	▼ 0.126	▼ 0.514	▼ 0.525	▼ 0.595
TT[Average]	▼ 0.249	▼ 0.518	▼ 0.198	▼ 0.505	▼ 0.542	▼ 0.623

Table B.13: Classification performance results for biological concepts.



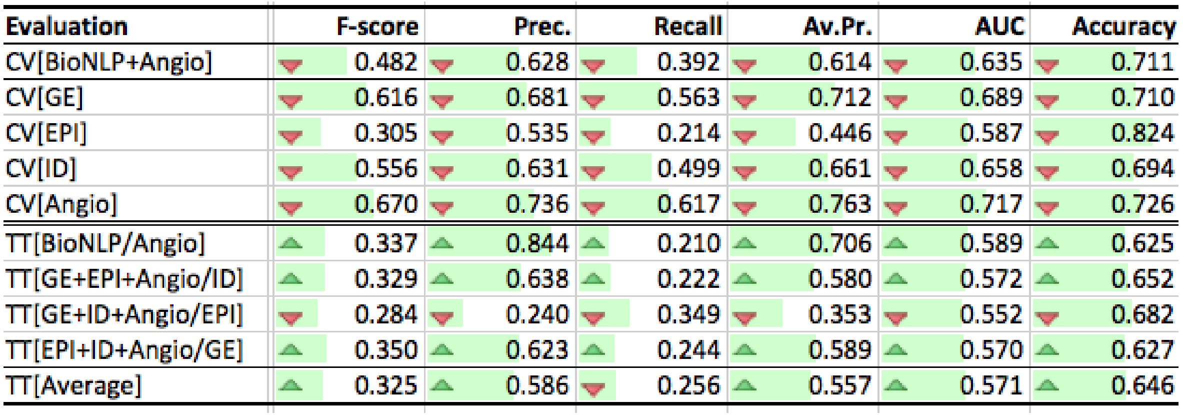
Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.191	▼ 0.554	▼ 0.116	▼ 0.486	▼ 0.534	▼ 0.665
CV[GE]	▼ 0.413	▼ 0.587	▼ 0.318	▼ 0.594	▼ 0.581	▼ 0.626
CV[EPI]	▼ 0.000	▼ 0.000	▼ 0.000	▼ 0.591	▼ 0.500	▼ 0.819
CV[ID]	▼ 0.338	▼ 0.544	▼ 0.251	▼ 0.542	▼ 0.556	▼ 0.626
CV[Angio]	▼ 0.607	▼ 0.631	▼ 0.586	▼ 0.702	▼ 0.650	▼ 0.657
TT[BioNLP/Angio]	▼ 0.080	▲ 0.841	▼ 0.042	▲ 0.658	▼ 0.518	▼ 0.563
TT[GE+EPI+Angio/ID]	▼ 0.039	▼ 0.490	▼ 0.020	▼ 0.444	▼ 0.504	▼ 0.615
TT[GE+ID+Angio/EPI]	▼ 0.229	▼ 0.233	▼ 0.225	▼ 0.299	▼ 0.530	▲ 0.725
TT[EPI+ID+Angio/GE]	▼ 0.000	▼ 0.000	▼ 0.000	▲ 0.706	▼ 0.500	▼ 0.587
TT[Average]	▼ 0.087	▼ 0.391	▼ 0.072	▼ 0.527	▼ 0.513	▼ 0.622

Table B.14: Classification performance results for syntactic arcs.



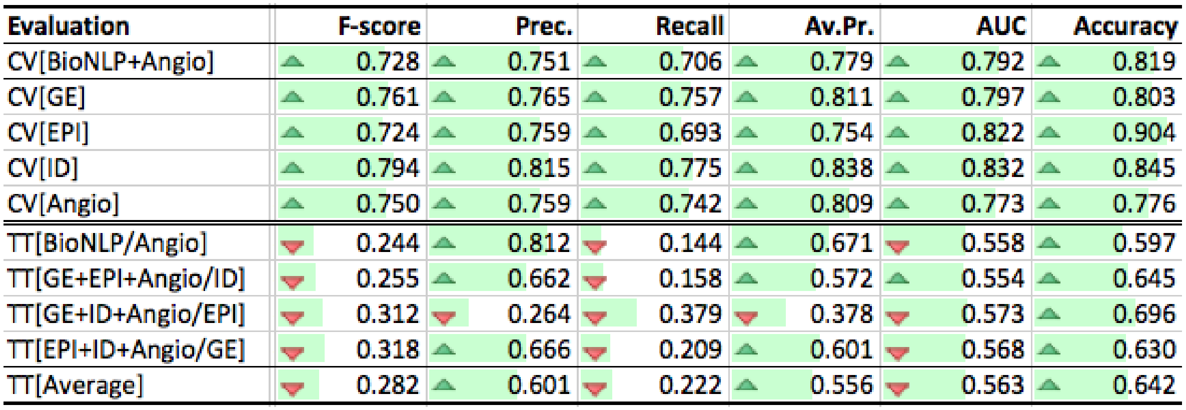
Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.088	▼ 0.481	▼ 0.048	▼ 0.428	▼ 0.511	▼ 0.655
CV[GE]	▼ 0.215	▼ 0.559	▼ 0.133	▼ 0.525	▼ 0.530	▼ 0.599
CV[EPI]	▼ 0.001	▼ 0.100	▼ 0.001	▼ 0.591	▼ 0.500	▼ 0.818
CV[ID]	▼ 0.078	▼ 0.476	▼ 0.043	▼ 0.494	▼ 0.506	▼ 0.612
CV[Angio]	▼ 0.514	▼ 0.679	▼ 0.415	▼ 0.680	▼ 0.625	▼ 0.645
TT[BioNLP/Angio]	▼ 0.065	▲ 0.782	▼ 0.034	▲ 0.627	▼ 0.513	▼ 0.559
TT[GE+EPI+Angio/ID]	▼ 0.031	▼ 0.417	▼ 0.016	▼ 0.406	▼ 0.501	▼ 0.612
TT[GE+ID+Angio/EPI]	▼ 0.156	▼ 0.225	▼ 0.119	▼ 0.252	▼ 0.514	▲ 0.766
TT[EPI+ID+Angio/GE]	▼ 0.044	▲ 0.629	▼ 0.023	▼ 0.528	▼ 0.507	▼ 0.591
TT[Average]	▼ 0.074	▼ 0.513	▼ 0.048	▼ 0.453	▼ 0.509	▼ 0.632

Table B.15: Classification performance results for the combination of biologically relevant verbs, biological concepts and syntactic arcs.



Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▼ 0.482	▼ 0.628	▼ 0.392	▼ 0.614	▼ 0.635	▼ 0.711
CV[GE]	▼ 0.616	▼ 0.681	▼ 0.563	▼ 0.712	▼ 0.689	▼ 0.710
CV[EPI]	▼ 0.305	▼ 0.535	▼ 0.214	▼ 0.446	▼ 0.587	▼ 0.824
CV[ID]	▼ 0.556	▼ 0.631	▼ 0.499	▼ 0.661	▼ 0.658	▼ 0.694
CV[Angio]	▼ 0.670	▼ 0.736	▼ 0.617	▼ 0.763	▼ 0.717	▼ 0.726
TT[BioNLP/Angio]	▲ 0.337	▲ 0.844	▲ 0.210	▲ 0.706	▲ 0.589	▲ 0.625
TT[GE+EPI+Angio/ID]	▲ 0.329	▲ 0.638	▲ 0.222	▲ 0.580	▲ 0.572	▲ 0.652
TT[GE+ID+Angio/EPI]	▼ 0.284	▼ 0.240	▼ 0.349	▼ 0.353	▼ 0.552	▼ 0.682
TT[EPI+ID+Angio/GE]	▲ 0.350	▲ 0.623	▲ 0.244	▲ 0.589	▲ 0.570	▲ 0.627
TT[Average]	▲ 0.325	▲ 0.586	▼ 0.256	▲ 0.557	▲ 0.571	▲ 0.646

Table B.16: Classification performance results for the combination of TF-IDF vectors of bags-of-trigrams, TF-IDF vectors of POS tags trigrams, TF-IDF vectors of POS-Chunk tag pairs trigrams, biologically relevant verbs, biological concepts and syntactic arcs.



Evaluation	F-score	Prec.	Recall	Av.Pr.	AUC	Accuracy
CV[BioNLP+Angio]	▲ 0.728	▲ 0.751	▲ 0.706	▲ 0.779	▲ 0.792	▲ 0.819
CV[GE]	▲ 0.761	▲ 0.765	▲ 0.757	▲ 0.811	▲ 0.797	▲ 0.803
CV[EPI]	▲ 0.724	▲ 0.759	▲ 0.693	▲ 0.754	▲ 0.822	▲ 0.904
CV[ID]	▲ 0.794	▲ 0.815	▲ 0.775	▲ 0.838	▲ 0.832	▲ 0.845
CV[Angio]	▲ 0.750	▲ 0.759	▲ 0.742	▲ 0.809	▲ 0.773	▲ 0.776
TT[BioNLP/Angio]	▼ 0.244	▲ 0.812	▼ 0.144	▲ 0.671	▼ 0.558	▲ 0.597
TT[GE+EPI+Angio/ID]	▼ 0.255	▲ 0.662	▼ 0.158	▲ 0.572	▲ 0.554	▲ 0.645
TT[GE+ID+Angio/EPI]	▼ 0.312	▼ 0.264	▼ 0.379	▼ 0.378	▼ 0.573	▲ 0.696
TT[EPI+ID+Angio/GE]	▼ 0.318	▲ 0.666	▼ 0.209	▲ 0.601	▼ 0.568	▲ 0.630
TT[Average]	▼ 0.282	▲ 0.601	▼ 0.222	▲ 0.556	▼ 0.563	▲ 0.642

B.6 Overall Features Comparison

These results are discussed in section 3.1.6.6.

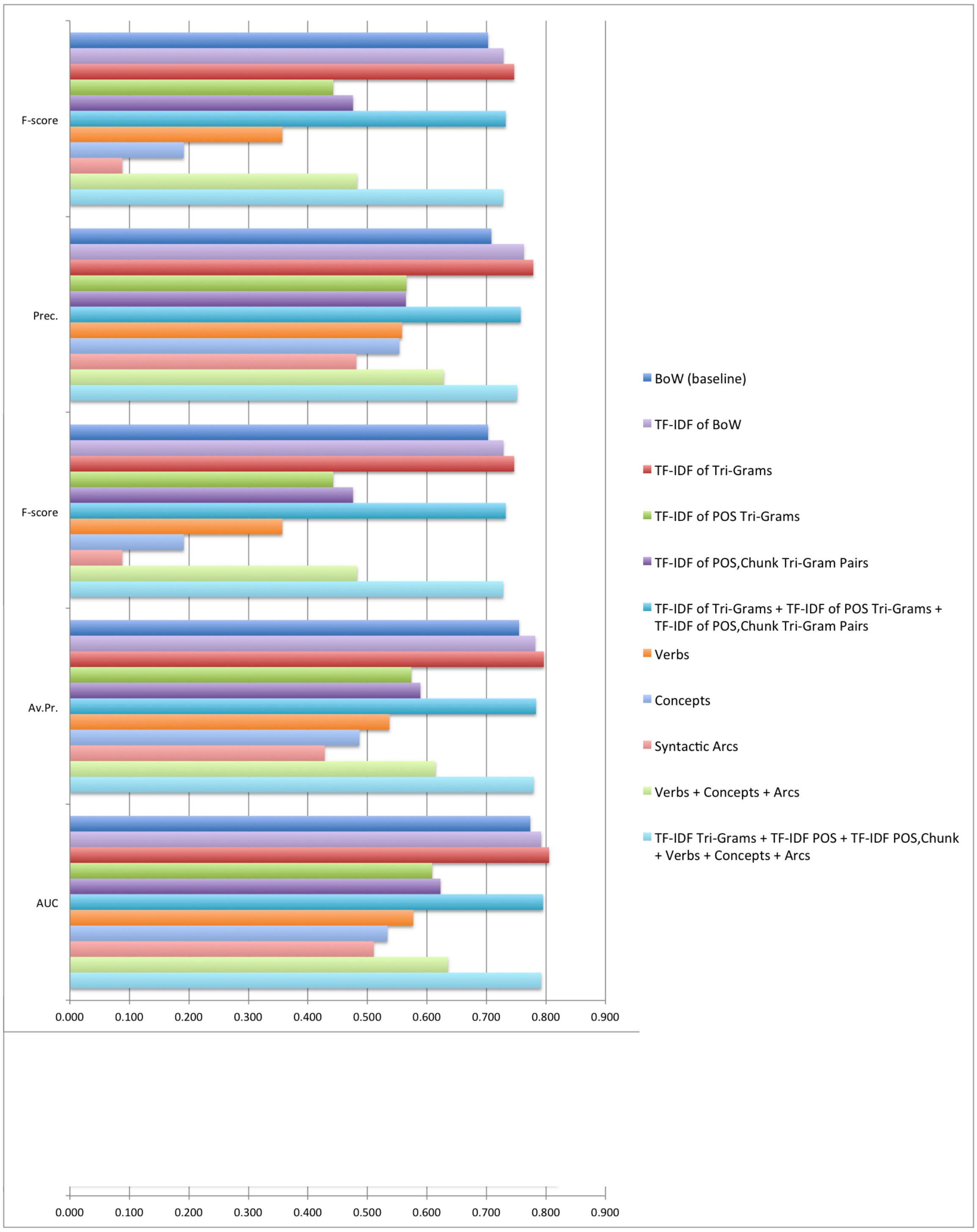


Figure B.7: Classification performance results for different types of features using stratified 10-fold cross-validation on the complete dataset (CV[BioNLP+Angio]).

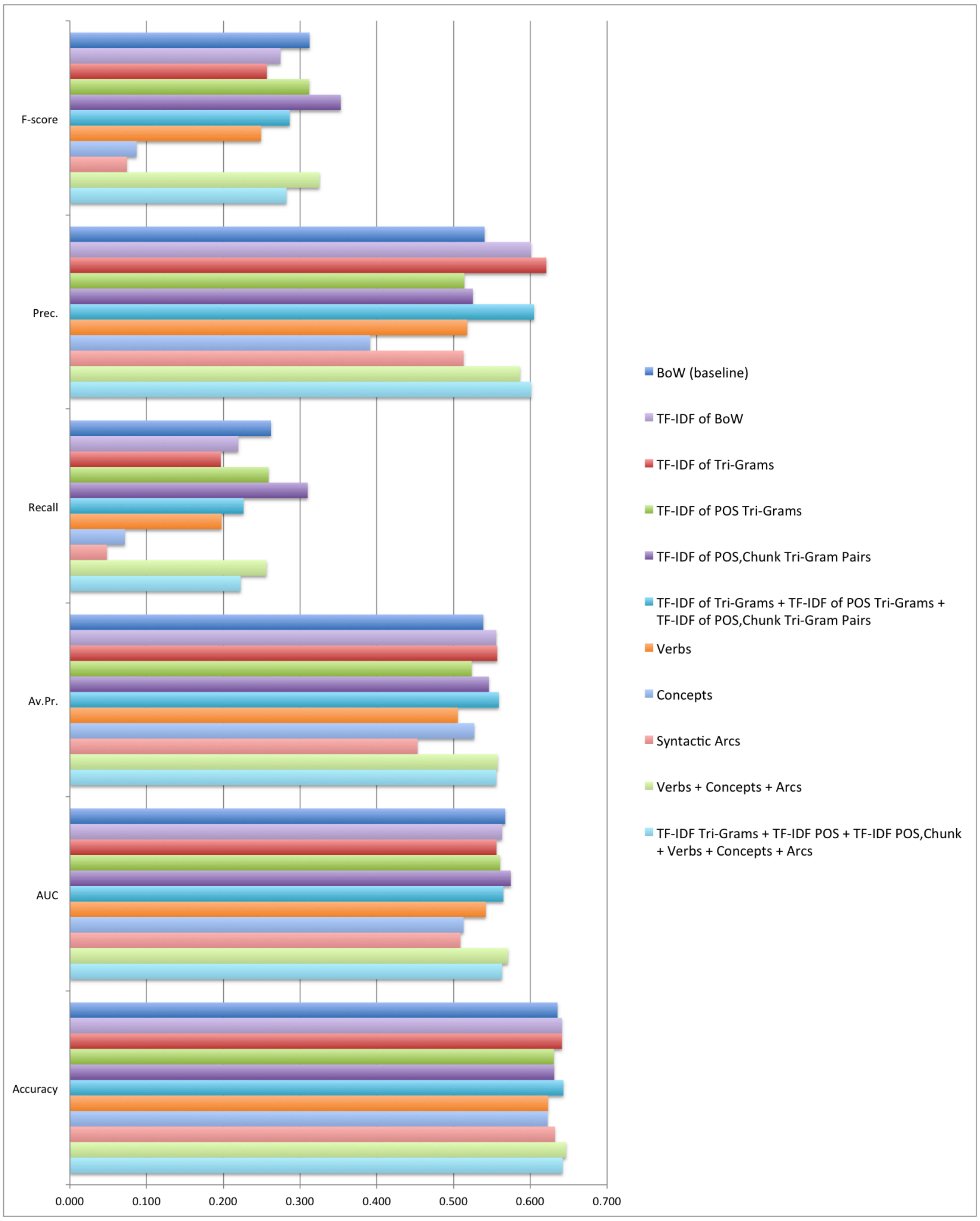


Figure B.8: Classification performance results for different types of features using leave-one-dataset-out evaluation averages (TT[Average]).

Appendix C

Sentence Retrieval API Response Samples

This appendix contains sample responses from the sentence retrieval API implemented for the purpose of this thesis. The API is described in section 3.2.7.

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 6
  },
  "response": {
    "numFound": 138089,
    "start": 0,
    "docs": [
      {
        "id": "16702786-5",
        "text": "Leukocytes␣obtained␣from␣AD␣patients␣had␣increased␣
           spontaneous␣TNF-alpha␣release␣...",
        "journal": "Aging␣Clin␣Exp␣Res",
        "pub_date": "2006␣Apr",
        "pmid": "16702786",
        "entities": [
          "IL-10␣production|GO:0032613::PROC|107",
          "IL-8␣production|GO:0032637::PROC|198",
          ...
        ],
        "rank": 1,
        "clf_score_all": 2.96038761499,
        "clf_score_ptm": -1.55149714025,
        "clf_score_genes": 3.87705171144,
        "clf_score_angio": -0.19408579026,
        "concept_types_count": 7
      },
      ...
    ],
    "concepts": {
        "GO:0032613::PROC": {
            "name": "interleukin-10␣production"
            "refs": [
                "GO:0032613",
                "NCIm:C1819437"
            ],
        },
        ...
    },
  },
  "request": {
    "query": "id:\"UMLS:C1521724:T046:DISO\"",
    "tokens": [
      {
        "id": "UMLS:C1521724:T046:DISO",
        "name": "Alzheimer's␣Disease"
      }
    ],
    "sort": "auto"
  }
```

Listing 5: Search API response sample (redacted for brevity)

```
[
  {
    "id": "UMLS:C0018787:T023:ANAT",
    "name": "Cardiac"
  },
  {
    "id": "CHEBI:22984:T103:CHED",
    "name": "Calcium"
  },
  {
    "id": "UMLS:C0007226:T022:ANAT",
    "name": "Cardiovascular"
  },
  ...
]
```

Listing 6: Concept suggestions API response sample (redacted for brevity)

Bibliography

[1] D. Fallows, *The Internet and daily life: Many Americans use the Internet in everyday activities, but traditional offline habits still dominate.* Pew Internet & American Life Project, 2004.

[2] Z. Lu, “PubMed and beyond: a survey of web tools for searching biomedical literature”, *Database: the journal of biological databases and curation*, vol. 2011, 2011.

[3] R. Blumberg and S. Atre, “The problem with unstructured data”, *DM REVIEW*, vol. 13, pp. 42–49, 2003.

[4] M. A. Hearst, “Untangling text data mining”, in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, Association for Computational Linguistics, 1999, pp. 3–10.

[5] A. M. Turing, “Computing machinery and intelligence”, *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[6] T. Nunes, D. Campos, S. Matos, and J. L. Oliveira, “Becas: biomedical concept recognition services and visualization”, *Bioinformatics*, 2013.

[7] J.-D. Kim, T. Ohta, Y. Tsuruoka, Y. Tateisi, and N. Collier, “Introduction to the bio-entity recognition task at jnlpba”, in *Proceedings of the international joint workshop on natural language processing in biomedicine and its applications*, Association for Computational Linguistics, 2004, pp. 70–75.

[8] L. Hirschman, A. Yeh, C. Blaschke, and A. Valencia, “Overview of biocreative: critical assessment of information extraction for biology”, *BMC bioinformatics*, vol. 6, no. Suppl 1, S1, 2005.

[9] M. Krallinger, A. Morgan, L. Smith, F. Leitner, L. Tanabe, J. Wilbur, L. Hirschman, and A. Valencia, “Evaluation of text-mining systems for biology: overview of the second biocreative community challenge”, *Genome Biol*, vol. 9, no. Suppl 2, S1, 2008.

[10] W. W. Lau, C. A. Johnson, and K. G. Becker, “Rule-based human gene normalization in biomedical text with confidence estimation”, in *Comput Syst Bioinformatics Conf*, World Scientific, vol. 6, 2007, pp. 371–379.

[11] G. Navarro, “A guided tour to approximate string matching”, *ACM computing surveys (CSUR)*, vol. 33, no. 1, pp. 31–88, 2001.

[12] T. Ono, H. Hishigaki, A. Tanigami, and T. Takagi, “Automated extraction of information on protein–protein interactions from the biological literature”, *Bioinformatics*, vol. 17, no. 2, pp. 155–161, 2001.

[13] E. M. Marcotte, I. Xenarios, and D. Eisenberg, “Mining literature for protein–protein interactions”, *Bioinformatics*, vol. 17, no. 4, pp. 359–363, 2001.

[14] J. M. Temkin and M. R. Gilder, “Extraction of protein interaction information from unstructured text using a context-free grammar”, *Bioinformatics*, vol. 19, no. 16, pp. 2046–2053, 2003.

[15] N. Daraselia, A. Yuryev, S. Egorov, S. Novichkova, A. Nikitin, and I. Mazo, “Extracting human protein interactions from medline using a full-sentence parser”, *Bioinformatics*, vol. 20, no. 5, pp. 604–611, 2004.

[16] A. Abi-Haidar, J. Kaur, A. Maguitman, P. Radivojac, A. Retchsteiner, K. Verspoor, Z. Wang, and L. M. Rocha, “Uncovering protein interaction in abstracts and text using a novel linear model and word proximity networks”, *Genome biology*, vol. 9, no. 2, S11, 2008.

[17] M. Song, H. Yu, and W.-S. Han, “Combining active learning and semi-supervised learning techniques to extract protein interaction sentences”, *BMC bioinformatics*, vol. 12, no. Suppl 12, S4, 2011.

[18] T. Polajnar, T. Damoulas, M. Girolami, *et al.*, “Protein interaction sentence detection using multiple semantic kernels”, *Journal of biomedical semantics*, vol. 2, no. 1, pp. 1–18, 2011.

[19] C. Nédellec, “Learning language in logic-genic interaction extraction challenge”, in *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Citeseer, vol. 7, 2005.

[20] S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, J. F. Hurdle, *et al.*, “Extracting information from textual documents in the electronic health record: a review of recent research”, *Yearb Med Inform*, vol. 35, pp. 128–44, 2008.

[21] Y. Garten, A. Coulet, and R. B. Altman, “Recent progress in automatically extracting information from the pharmacogenomic literature”, *Pharmacogenomics*, vol. 11, no. 10, pp. 1467–1489, 2010.

[22] M. Krallinger, F. Leitner, and A. Valencia, “Analysis of biological processes and diseases using text mining approaches”, in *Bioinformatics Methods in Clinical Research*, Springer, 2010, pp. 341–382.

[23] U. Hahn, K. B. Cohen, Y. Garten, and N. H. Shah, “Mining the pharmacogenomics literature—a survey of the state of the art”, *Briefings in bioinformatics*, vol. 13, no. 4, pp. 460–494, 2012.

[24] J.-D. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii, “Overview of bionlp’09 shared task on event extraction”, in *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, Association for Computational Linguistics, 2009, pp. 1–9.

[25] J.-D. Kim, S. Pyysalo, T. Ohta, R. Bossy, N. Nguyen, and J. Tsujii, “Overview of bionlp shared task 2011”, in *Proceedings of the BioNLP Shared Task 2011 Workshop*, Association for Computational Linguistics, 2011, pp. 1–6.

[26] P. Stenetorp, S. Pyysalo, G. Topić, T. Ohta, S. Ananiadou, and J. Tsujii, “Brat: a web-based tool for nlp-assisted text annotation”, in *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, Association for Computational Linguistics, 2012, pp. 102–107.

[27] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval.* Cambridge University Press Cambridge, 2008, vol. 1.

[28] R. Baeza-Yates, B. Ribeiro-Neto, *et al.*, *Modern information retrieval.* ACM press New York, 1999, vol. 463.

[29] D. Kuropka, “Modelle zur repräsentation natürlichsprachlicher dokumente-information-filtering und-retrieval mit relationalen datenbanken”, *Advances in Information Systems and Management Science*, vol. 10, 2004.

[30] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing.* MIT press, 1999.

[31] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters”, *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[32] R. H. Creecy, B. M. Masand, S. J. Smith, and D. L. Waltz, "Trading mips and memory for knowledge engineering", *Communications of the ACM*, vol. 35, no. 8, pp. 48–64, 1992.

[33] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization", in *Proceedings of the seventh international conference on Information and knowledge management*, ACM, 1998, pp. 148–155.

[34] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning.* The MIT Press, 2012.

[35] F. Sebastiani, "Machine learning in automated text categorization", *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002.

[36] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, *Supervised machine learning: a review of classification techniques*, 2007.

[37] L. J. Jensen, J. Saric, and P. Bork, "Literature mining for the biologist: from information retrieval to biological discovery", *Nature reviews genetics*, vol. 7, no. 2, pp. 119–129, 2006.

[38] D. L. Wheeler, T. Barrett, D. A. Benson, S. H. Bryant, K. Canese, V. Chetvernin, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, *et al.*, "Database resources of the national center for biotechnology information", *Nucleic acids research*, vol. 35, no. suppl 1, pp. D5–D12, 2007.

[39] R. Hoffmann and A. Valencia, "A gene network for navigating the literature", *Nature genetics*, vol. 36, no. 7, pp. 664–664, 2004.

[40] H. Chen and B. M. Sharp, "Content-rich biological network constructed by mining pubmed abstracts", *BMC bioinformatics*, vol. 5, no. 1, p. 147, 2004.

[41] Y. Miyao, T. Ohta, K. Masuda, Y. Tsuruoka, K. Yoshida, T. Ninomiya, and J. Tsujii, "Semantic retrieval for the accurate identification of relational concepts in massive textbases", in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2006, pp. 1017–1024.

[42] H.-M. Müller, E. E. Kenny, and P. W. Sternberg, "Textpresso: an ontology-based information retrieval and extraction system for biological literature", *PLoS biology*, vol. 2, no. 11, e309, 2004.

[43] A. Divoli and T. K. Attwood, "Bioie: extracting informative sentences from the biomedical literature", *Bioinformatics*, vol. 21, no. 9, pp. 2138–2139, 2005.

[44] M. J. Cafarella, D. Downey, S. Soderland, and O. Etzioni, "Knowitnow: fast, scalable information extraction from the web", in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2005, pp. 563–570.

[45] P. M. Andersen, P. J. Hayes, and A. K. Huettner, "Automatic extraction of facts from press releases to generate news stories", in *Proceedings of the third ...*, 1992.

[46] Ö. Uzuner, X. Zhang, and T. Sibanda, "Machine learning and rule-based approaches to assertion classification", *Journal of the American Medical Informatics Association*, vol. 16, no. 1, pp. 109–115, 2009.

[47] G. Hripcsak, C. Friedman, P. O. Alderson, W. DuMouchel, S. B. Johnson, and P. D. Clayton, "Unlocking clinical data from narrative reports: a study of natural language processing", *Annals of internal medicine*, vol. 122, no. 9, pp. 681–688, 1995.

[48] C. Friedman, P. O. Alderson, J. H. Austin, J. J. Cimino, and S. B. Johnson, "A general natural-language text processor for clinical radiology", *Journal of the American Medical Informatics Association*, vol. 1, no. 2, pp. 161–174, 1994.

[49] C. Arighi, Z. Lu, M. Krallinger, K. Cohen, W. Wilbur, A. Valencia, L. Hirschman, and C. Wu, "Overview of the biocreative iii workshop", *BMC bioinformatics*, vol. 12, no. Suppl 8, S1, 2011.

[50] D. Campos, S. Matos, and J. L. Oliveira, "A modular framework for biomedical concept recognition", *BMC bioinformatics*, vol. 14, no. 1, p. 281, 2013.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: machine learning in Python", *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[52] S. Bird, "Nltk: the natural language toolkit", in *Proceedings of the COLING/ACL on Interactive presentation sessions*, Association for Computational Linguistics, 2006, pp. 69–72.

[53] W. Hersh and E. Voorhees, "TREC genomics special issue overview", *Information Retrieval*, vol. 12, no. 1, pp. 1–15, Dec. 2008.

[54] X. Wang, I. McKendrick, I. Barrett, I. Dix, T. French, J. Tsujii, and S. Ananiadou, "Automatic extraction of angiogenesis bioprocess from text.", *Bioinformatics*, vol. 27, no. 19, pp. 2730–2737, Oct. 2011.

[55] J.-D. Kim, N. Nguyen, Y. Wang, J. Tsujii, T. Takagi, and A. Yonezawa, "The Genia Event and Protein Coreference tasks of the BioNLP Shared Task 2011.", *BMC Bioinformatics*, vol. 13 Suppl 11, S1, 2012.

[56] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "Genia corpus—a semantically annotated corpus for bio-textmining", *Bioinformatics*, vol. 19, no. suppl 1, pp. i180–i182, 2003.

[57] S. Pyysalo, T. Ohta, R. Rak, D. Sullivan, C. Mao, C. Wang, B. Sobral, J. Tsujii, and S. Ananiadou, "Overview of the ID, EPI and REL tasks of BioNLP Shared Task 2011.", *BMC Bioinformatics*, vol. 13 Suppl 11, S2, 2012.

[58] E. International, *The JSON Data Interchange Format*, ECMA-404.

[59] C. S. T. 2. Organizers, *Conll data format*, [Online; accessed November-2012]. [Online]. Available: `http://nextens.uvt.nl/depparse-wiki/DataFormat`.

[60] P. H. [B. (N. C. for Biotechnology Information (US)., *Pubmed stopwords*, [Online; accessed November-2012], 2005-. [Online]. Available: `http://www.ncbi.nlm.nih.gov/books/NBK3827/table/pubmedhelp.T43/`.

[61] A. Rajaraman and J. D. Ullman., *Mining of Massive Datasets.* 2011.

[62] S. P. Abney, *Parsing by chunks.* Springer, 1992.

[63] Y. Sasaki, S. Montemagni, P. Pezik, D. Rebholz-Schuhmann, J. McNaught, and S. Ananiadou, "Biolexicon: a lexical resource for the biology domain", in *Proc. of the third international symposium on semantic mining in biomedicine (SMBM 2008)*, vol. 3, 2008, pp. 109–116.

[64] K. Sagae and J. Tsujii, "Dependency parsing and domain adaptation with lr models and parser ensembles.", in *EMNLP-CoNLL*, vol. 2007, 2007, pp. 1044–1050.

[65] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization", in *ICML*, vol. 97, 1997, pp. 412–420.

[66] J. F. Kenney and E. S. Keeping, "Mathematics of statistics, part 2", 1962.

[67] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection", in *IJCAI*, vol. 14, 1995, pp. 1137–1145.

[68] B. Efron, "Estimating the error rate of a prediction rule: improvement on cross-validation", *Journal of the American Statistical Association*, vol. 78, no. 382, pp. 316–331, 1983.

[69] B. Efrom and R. J. Tibshirani, “An introduction to the bootstrap”, *Champman and Hall/CRC*, 1993.

[70] D. H. Wolpert, “Stacked generalization”, *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[71] T. Fawcett, “An introduction to roc analysis”, *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[72] M. Padlipsky, *Perspective on the ARPANET reference model*, RFC 871, Internet Engineering Task Force, 1982.

[73] M. Potel, “Mvp: model-view-presenter the taligent programming model for c++ and java”, *Taligent Inc*, 1996.

[74] S. Tilkov and S. Vinoski, “Node.js: using javascript to build high-performance network programs”, *Internet Computing, IEEE*, vol. 14, no. 6, pp. 80–83, 2010.

[75] C. Strauch, U.-L. S. Sites, and W. Kriha, “Nosql databases”, *URL: http://www.christof-strauch.de/nosqldbs.pdf*, 2011.

[76] E. P. U. H. [B. (N. C. for Biotechnology Information (US)., *E-utilities quick start*, [Online; accessed November-2012], 2010-. [Online]. Available: `http://www.ncbi.nlm.nih.gov/books/NBK25500/`.

[77] D. Rebholz-Schuhmann, H. Kirsch, and G. Nenadic, “Iexml: towards a framework for interoperability of text processing modules to improve annotation of semantic types in biomedical text”, *Proc. of BioLINK, ISMB 2006*, 2006.

[78] M. F. Porter, “An algorithm for suffix stripping”, *Program: electronic library and information systems*, vol. 14, no. 3, pp. 130–137, 1980.

[79] O. Bodenreider, “The unified medical language system (umls): integrating biomedical terminology”, *Nucleic acids research*, vol. 32, no. suppl 1, pp. D267–D270, 2004.

[80] K. Hettne, R. Stierum, M. Schuemie, P. Hendriksen, B. Schijvenaars, E. Mulligen, J. Kleinjans, and J. Kors, “A dictionary to identify small molecules and drugs in free text”, *Bioinformatics*, vol. 25, no. 22, pp. 2983–2991, 2009.

[81] L. Geer, A. Marchler-Bauer, R. Geer, L. Han, J. He, S. He, C. Liu, W. Shi, and S. Bryant, “The ncbi biosystems database”, *Nucleic acids research*, vol. 38, no. suppl 1, pp. D492–D496, 2010.