Semi-supervised learning for biomedical information extraction

Andreas Vlachos

University of Cambridge
Computer Laboratory
Peterhouse

December 2009

This dissertation is submitted for
the degree of Doctor of Philosophy

Declaration

This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except where specifically indicated in the text.

This dissertation does not exceed the regulation length of 60 000 words, including tables and footnotes.

Semi-supervised learning for biomedical information extraction

Andreas Vlachos

Summary

This thesis explores the application of semi-supervised learning to biomedical information extraction. The latter has emerged in recent years as a challenging application domain for natural language processing techniques. The challenge stems partly from the lack of appropriate resources that can be used as labeled training data. Therefore, we choose to focus on semi-supervised learning techniques which enable us to take advantage of human supervision combined with unlabeled data.

We begin with a short introduction to biomedical information extraction and semi-supervised learning in Chapter 1. Chapter 2 focuses on the task of biomedical named entity recognition. Using raw abstracts and a dictionary of gene names we develop two systems for this task. Furthermore, we discuss annotation issues and demonstrate how the performance can be improved using user feedback in realistic conditions. In Chapter 3 we develop two biomedical event extraction systems: a rule-based one and a machine learning based one. The former needs only an annotated dictionary and syntactic parsing as input, while the latter requires partial event annotation additionally. Both systems achieve performances comparable to systems utilizing fully annotated training data. Chapter 4 discusses the task of lexical-semantic clustering using Dirichlet process mixture models. We review the unsupervised learning method used, which allows the number of clusters discovered to be determined by the data. Furthermore, we introduce a new clustering evaluation measure that addresses some shortcomings of the existing measures. Chapter 5 introduces a method of guiding the clustering solution using pairwise links between instances. Furthermore, we present a method of selecting these pairwise links actively in order to decrease the amount of supervision required. Finally, Chapter 6 assesses the contributions of this thesis and highlights directions for future work.

Acknowledgements

I would first like to thank my supervisor Ted Briscoe for his support and guidance through my PhD. Also, I would like to thank Zoubin Ghahramani for offering advice and technical expertise. While I often found myself between conflicting views, I firmly believe that this resulted in a better thesis. I would like to express my gratitude to the Gates Cambridge Trust, which funded my studies in Cambridge, as well as Peterhouse and the Computer Laboratory for supporting my trips to conferences.

I am grateful to the members of the research groups I have been involved with during my studies in Cambridge, namely the Natural Language and Information Processing group at the Computer Laboratory and the Machine Learning group at the Engineering department. In particular, I am indebted to my collaborators in the FlySlip project, Caroline Gasperin, Nikiforos Karamanis and Ian Lewin, to Anna Korhonen for her help with verb clustering and to Jurgen Van Gael for technical advice. Special thanks go to Marek Rei and Diarmuid Ó Séaghdha for reading my thesis. Diarmuid gets extra thanks for reading each of the chapters as I was writing them. Advaith, Diarmuid and Niki accompanied me in all the travelling, the beer drinking and the chats about all things, inside and outside the lab, thus making it a fun place to be. My life would have been empty without my friends in music, tango and football who provided me with refreshing escapes from work. Finally, I want to thank my parents, Ioannis and Stavroula for giving me solid foundations early on, my sister Georgia and my partner Dina for supporting me emotionally and listening to me patiently while rambling about my work.

Contents

Chapter 1

Introduction

This thesis explores the application of semi-supervised learning to biomedical information extraction (BioIE). BioIE has emerged in recent years as a challenging application domain for natural language processing (NLP) components. The challenge stems partly from the lack of appropriate resources that can be used as labeled training data, thus inhibiting the use of supervised learning techniques which are commonly applied in NLP. Therefore, we chose to focus on learning techniques which enable us to take advantage of human supervision combined with unlabeled data. The broad spectrum of these techniques is referred to as semi-supervised learning.

In this chapter, we introduce biomedical information extraction and discuss its relation to general information extraction and NLP research (Section 1.1). Then, we define semi-supervised learning and position the approaches explored with respect to existing work (Section 1.2). Finally, we define the goals of this thesis (Section 1.3) and summarize the contents of each chapter (Section 1.4).

1.1 Biomedical information extraction

Information extraction (IE) focuses on extracting structured information from natural language text. The inferred structure can be in a variety of forms. The simplest and most common one is a set of labels assigned to textual strings, such as names of *persons* or *organizations* in the context of named entity recognition (Tjong Kim Sang and De Meulder, 2003). More complex scenarios have involved predefined templates with appropriate slots that IE systems need to fill in, as in the MUC-7 shared task (Chinchor, 1998). More recently, IE research has focused on extracting relations between entities, e.g. the relations specified in the Automated Content Extraction (ACE) project (Doddington et al., 2004). In order for IE to achieve its aims, it makes use of various NLP components such as part-of-speech tagging, syntactic parsing and word sense disambiguation.

Biomedical information extraction (BioIE) focuses on extracting information from biomedical scholarly publications. It emerged as a research field due to the increasingly vast volume of literature published in biomedicine. Scientists in this field are interested in methods that can extract information automatically. While some structured knowledge databases exist (examples include model organism databases and ontologies), it is acknowledged that the curation process of these resources tends to lag behind the literature being published. Moreover, while these databases are designed in order to accommodate the information needs of a variety of researchers, they are unlikely to meet specialized requests that might arise. For example, a researcher might be interested in a particular definition of a biomedical event that was not considered during database design and therefore such events were not curated. Research in BioIE can be useful to both curators and users of curated databases by providing them with means of processing the literature effectively.

For the IE community (as well as the broader natural language processing community), the biomedical domain is a testbed on which the portability of the methods developed can be evaluated. Most of the IE systems are designed with portability as a given, and it is assumed that a technique that performs well in the domain it was developed will perform equally well in a different domain. The majority of the techniques used in IE are applied to newswire text and the task definitions are tailored to this genre, since resources from this domain were available earlier. Research in BioIE ports existing IE techniques and task definitions, tailors them to the domain and the resources available and evaluates them. Usually, this process gives rise to interesting issues concerning the adaptation of the techniques as well as the task definitions themselves, therefore providing useful feedback to the research conducted in IE. The biomedical domain is particularly suitable to this end because its users (biomedical scientists) are actively interested in the results of this research in order to advance their own work. Therefore, they provide the BioIE research community with labeled data, evaluation methods and other resources. Evidence of the importance of BioIE to both research communities is the number of specialized workshops (e.g. the BioNLP, BioCreative and BioLink workshops) which involve researchers in natural language processing as well as in biomedical sciences.

Research in BioIE has focused on a variety of tasks of increasing complexity, following the footsteps of mainstream IE. Initially, the community focused on biomedical named entity recognition and the closely connected task of gene normalization, e.g. the evaluations of JNLPBA (Kim et al., 2004) and BioCreative (Hirschman et al., 2005b). The reasonable performance levels achieved led to the progressive inclusion of some more complex tasks, such as the detection of protein-protein interactions in LLL 2005 (Nédellec, 2005) and BioCreative II (Krallinger et al., 2008). More recently, the BioNLP 2009 shared task (Kim et al., 2009) focused on event extraction exclusively, considering named entity recognition as a given. It is worth pointing out that event extraction has followed a similar trajectory in the broader NLP community, with the ACE event extraction evaluation (Doddington

et al., 2004) following the good performances achieved in the named entity recognition shared tasks that preceded it (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003).

In this thesis, we explore a range of tasks that are considered important in biomedical information extraction. We begin our exploration with named entity recognition, proceed with event extraction and end with semantic verb clustering.

1.2 Semi-Supervised Learning for information extraction

In natural language processing, supervised learning methods have been commonly applied to a variety of tasks. These methods learn statistical patterns from a set of examples labeled with the answers that a system is supposed to produce. While the performances achieved in most tasks are satisfactory, these methods need explicitly labeled data for training. This requirement becomes a pressing issue when attempting new tasks and/or considering new domains. The production of labeled data requires a substantial amount of human labour, which typically employs domain experts. Therefore, labeled data is expensive to create and it is available in limited quantities. Furthermore, even when such resources are created, they tend to be task- or domain-specific and as a result unlikely to be reusable. For example, a dataset with gene names annotated according to a particular definition is unlikely to be useful as it is if we need to consider a different definition of gene names.

For these reasons, unsupervised learning approaches to natural language processing have been gaining popularity. These approaches do not require labeled data, but rely on discovering structure in unlabeled data. Since labels are not needed, they can take advantage of the large datasets that are commonly available for natural language processing. This attribute can be useful in overcoming data sparsity issues. However, while appealing, unsupervised approaches are used relatively infrequently because they tend to have worse performance than their supervised counterparts. Moreover, they are harder to design since they need to compensate for the lack of labels in the data they observe.

Semi-supervised learning (SSL) encompasses attempts to combine unlabeled data with human supervision. The latter is usually in the form of explicitly labeled data (Zhu and Goldberg, 2009), but it can also be in the form of rules, dictionaries and other resources. For example, pairwise constraints (Wagstaff and Cardie, 2000) can be used for clustering tasks, or dictionaries for part-of-speech tagging (Haghighi and Klein, 2006). It is important to note that different kinds of supervision require different amounts of human effort, e.g. annotating large amounts of running text with part-of-speech tags is likely to be more expensive than compiling a tag dictionary, especially considering that

the latter might be readily available for some languages. It is also important to minimize the effort required by tailoring to the task and the learning algorithm used. For this purpose, active learning techniques (Settles, 2009) are frequently considered in which the learning algorithm selects the supervision to be added. Overall, semi-supervised learning approaches aim at obtaining good performance at a low cost by combining (potentially large) amounts of unlabeled data with human supervision.

In this thesis, we focus on SSL methods that incorporate supervision in the form of manually curated resources and/or partially labeled examples. This is particularly suitable to the biomedical domain, where there are many structured knowledge sources, e.g. model organism databases, as well as domain experts from whom it is easier to obtain answers to specific annotation requests rather than explicitly labeled data.

1.3 Research Goals

The goal of this thesis is to develop semi-supervised approaches for biomedical information extraction tasks. Initially we tackle biomedical named entity recognition (BioNER), which apart from being an important IE task, is also an essential component for more complex tasks (Chapter 2). We focus on gene name recognition and present two systems that are evaluated on abstracts and full papers. The supervision used is a dictionary of gene names from a model organism database. Additionally, we improve our performance further by taking advantage of user feedback.

Following this, we discuss the task of bio-molecular event extraction in the context of the BioNLP 2009 shared task (Kim et al., 2009), which involves the extraction of relations between biomedical entities (Chapter 3). First we develop a rule-based approach that requires only an annotated dictionary and the output of a syntactic parser, which allows us to gain a better understanding of the task. Then we improve it by incorporating machine learning using partial event annotations which are simpler to annotate than complete events. Despite using less annotation, the systems developed are competitive with those that took part in the shared task.

The differences observed in the language of biomedical literature prompted us to explore lexical-semantic verb clustering in order to gain insights that can be useful to IE tasks (Chapter 4). For this purpose we employ a non-parametric Bayesian model which is completely unsupervised and, unlike previous approaches, is able to determine the number of clusters in a dataset.

Finally, we extend the clustering approach presented to incorporate pairwise constraints in the clustering discovered (Chapter 5). This enables users to adapt the output with respect to their intuitions on which verbs should be clustered together or separately, thus injecting supervision that is relevant to the application at hand. In order to minimize

their annotation effort we additionally propose an active learning scheme in which the model requests for particular constraints to be incorporated.

While verb clustering is not a typical IE task and thus represents a shift away from the main topic of the thesis. Nevertheless, we argue it is still relevant since it can be used to explore the biomedical domain and we maintain our focus on semi-supervised approaches. We did not attempt anaphora resolution, despite it being a common IE task, because in the tasks tackled it would have been of limited use. In particular, anaphora resolution commonly uses the output of named entity recognition as input, while in the event extraction task it would not be helpful due to the low overall performances as well as the lack of appropriate resources (more details in Section 3.3). Furthermore, while we did not explore the issue of domain-adaptation of syntactic parsing, the use of existing techniques in this thesis allows us to assess its importance to BioIE.

The tasks and the techniques used to tackle them are presented in detail in their respective chapters. We explore a range of approaches, from rule-based systems to non-parametric Bayesian modeling, the choice being dependent on the task in question and the resources available. Independently of the approach chosen, our main aim is to minimise the amount of human supervision required in order to reach competitive performance levels. We argue that this is an important criterion with respect to the applicability of the approaches, since the less supervision required, the more likely an approach is to be deployed in a real application setting.

1.4 Thesis Summary

In what follows, we summarize the contents of each chapter:

- Chapter 2 tackles the task of biomedical named entity recognition. We generate labeled training data using raw abstracts and a dictionary of gene names and develop two systems for this task. Furthermore, we ameliorate annotation issues by improving existing guidelines and evaluate the systems developed on abstracts and full papers, unlike most previous work which focuses on abstracts only. Also, we demonstrate how the performance can be improved using user feedback in realistic conditions rather than in simulated experiments which is the case in most existing work.

- Chapter 3 describes two biomedical event extraction systems: a rule-based one and a machine learning based one. The former needs only an annotated dictionary and syntactic parsing as input, while the latter requires partial event annotation additionally. Both systems achieve performances comparable to systems utilizing fully annotated training data and external resources.

- Chapter 4 discusses the task of lexical-semantic clustering using Dirichlet process mixture models. The main advantage of this unsupervised learning method is that it allows the number of clusters discovered to be determined by the data, while in previous work the number of clusters must be defined in advance. We present state-of-the-art results and introduce a new clustering evaluation measure that addresses some shortcomings of the existing measures.

- Chapter 5 introduces a novel method of guiding the clustering solution using pairwise links between instances. Furthermore, we present a method of selecting these pairwise links actively in order to decrease the amount of supervision required.

- Chapter 6 assesses the contributions of this thesis and highlights directions for future work.

Chapter 2

Biomedical named entity recognition

2.1 Introduction

In this chapter we begin our exploration of the application of semi-supervised learning techniques to biomedical information extraction by tackling the task of biomedical named entity recognition (BioNER), which has attracted a lot of attention recently. It is an important task because it is a prerequisite to other more complex ones, such as anaphora resolution (Gasperin, 2006), entity normalization (Hirschman et al., 2005a) and event extraction (Kim et al., 2009).

Named entity recognition was initially explored in the context of newswire text in MUC-6 (Sundheim, 1995). More recent incarnations include the CoNLL shared tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003) in which the task is defined as the recognition of names of persons, locations, organizations and of miscellaneous entities that do not belong to the previous three groups. In the following example sentence, the named entities are in boldface with their entity type in parentheses:

> **Wolff** (*PERSON*), currently a journalist in **Argentina** (*LOCATION*), played with **Del Bosque** (*PERSON*) in the final years of the seventies in **Real Madrid** (*ORGANIZATION*).

In the biomedical domain, there have been three shared tasks (BioNLP/NLPBA 2004 (Kim et al., 2004), BioCreative (Hirschman et al., 2005b) and BioCreative2 (Krallinger and Hirschman, 2007)) which involved a version of BioNER using manually annotated training material and supervised machine learning methods. The systems built tend to be quite similar to those developed for generic named entity recognition (NER). In parallel, there have been successful efforts in bootstrapping BioNER systems using automatically generated training data from extant domain resources, e.g. the approach of Morgan et al.

(2004). Such approaches have a significant appeal, since they do not require manual annotation of training material, which is an expensive and lengthy process.

In this chapter we build on the idea of bootstrapping, which has been applied by Collins and Singer (1999) in the newswire domain and by Morgan et al. (2004) in the biomedical domain. It is based on creating training material automatically using existing domain resources, as suggested by Craven and Kumlien (1999), and then training a supervised named entity recognition system. In this work we apply this technique to the recognition of gene names in articles from the *Drosophila* literature. *Drosophila* is a very popular organism in biomedicine and it has its own dedicated model organism database, FlyBase.[1] The latter is manually curated by biologists and in the context of this work it provides us with two important resources: materials that can be used to develop a BioNER system and users who are eager to provide their feedback.

We begin our exploration by reproducing the experiments of Morgan et al. (2004) in bootstrapping a BioNER recognizer and evaluating it on abstracts from the *Drosophila* literature (Section 2.2). The results obtained motivate us to discuss annotation guidelines and evaluation issues (Sections 2.3 and 2.4 respectively). Then we switch our focus to full papers and describe a new evaluation corpus (Section 2.5). The new evaluation and datasets exposed some shortcomings of the BioNER system built which we address by developing a new system based on conditional random fields and syntactic parsing (Section 2.6). Following this, we demonstrate how feedback from the users of a curation-assistance interface can be collected and used in order to improve performance (Section 2.7). We close the chapter with discussion of the results and pointers to future work (Section 2.8).

2.2 Reproducing the Morgan et al. experiment

FlyBase provides a dictionary of all *Drosophila* genes and their synonyms that appear in the current curated literature indicating where a specific name is used to refer to a particular gene. Morgan et al. (2004) exploited this information to create annotated material to train a gene name recognizer. In brief, abstracts were tokenized and the gene names linked to specific abstracts in FlyBase were automatically tagged applying longest-extent pattern matching. This process resulted in a large but automatically annotated noisy corpus which was in turn used to train a hidden Markov model (HMM) (Rabiner, 1990).

In this section, we replicate this experiment using an enlarged dataset and different toolkit. Initially we construct a list of all the articles for which FlyBase recorded at least one gene mentioned within it. Then all the abstracts of those articles are retrieved resulting in a total of 16,609 abstracts (9.5% more than Morgan et al. (2004)). The abstracts

[1] http://www.flybase.net

are split in sentences and tokenized using the corresponding components of the publicly available syntactic parsing toolkit RASP (Briscoe et al., 2006).[2] Then, following Morgan et al. (2004), all the gene names licensed by the associated FlyBase gene name list are annotated in each abstract.

The 16,609 abstracts processed contain approximately 16,881 distinct gene names out of a total of 97,227 gene names and synonyms recorded in FlyBase.[3] Therefore, many gene names and synonyms do not appear in the training material. In addition, as Morgan et al. (2004) note, there are gene synonyms that are common English words, such as "to" and "by", resulting in precision errors in the training data. It is worth observing though that by using the FlyBase gene list for each abstract such problems occur only in abstracts that are associated with such gene synonyms. A different problem is that sometimes genes mentioned in abstracts are not in the respective FlyBase gene lists of those articles (as only relevant sections of the article are curated), resulting in recall errors. Finally, not all synonyms used in the abstracts for each gene are recorded in FlyBase. Nevertheless, our expectation is that given the relatively restricted language used in abstracts, the noise introduced will not be detrimental to our efforts.

The hidden Markov model we use in our experiments is part of the open source toolkit LingPipe.[4] It is a hybrid 1st/2nd-order HMM that uses Witten-Bell smoothing. For each token $t[n]$ and possible label $l[n]$, the following joint probability is computed, conditioned on the previous two tokens and the previous label:

$$P(t[n], l[n] | l[n-1], t[n-1], t[n-2]) \tag{2.1}$$

The possible labels for each token are **B**egin-entity, **I**nside-entity and **O**utside-entity. This results in the tokens before and after a gene name being labeled as **O**utside-entity, the first token of the gene name being labeled **B**egin-entity and all the other tokens of the gene name (in case of a multi-token gene name) being labeled as **I**nside-entity. Tokens unseen in the training data are assigned to simple morphological classes depending on capitalization, existence of numerals, etc. During training, rare tokens are replaced by their respective classes in order to obtain probability estimates for the classes to which unseen tokens are assigned during testing.

Overall, this approach is highly lexical and conservative compared to others (e.g. Crim et al. (2005)) which deploy more abstract and general features to achieve greater domain-independence. The system discussed in this section achieves relatively high precision by only generalizing to unseen names in lexical contexts which are clearly indicative of gene names in the training data.

[2] http://www.cogs.susx.ac.uk/lab/nlp/rasp/

[3] Exact figures depend on how much normalization (e.g. homogenizing punctuation, Greek letters, capitalization and whitespace) one applies to the names before counting them.

[4] http://www.alias-i.com/lingpipe/. The version used in these experiments is 2.1.1.

To evaluate the performance of the trained recognizer we use the dataset created by Morgan et al. (2004). It consists of 86 abstracts doubly-annotated by a biologist curator and a computational linguist. The performance of the BioNER system described on each annotation version was 80.9%/74.9%/77.7% and 84.2%/84.8%/84.5% respectively (Recall/Precision/F-score). To calculate these figures we used the evaluation script of the BioNLP 2004 shared task.[5] For comparison, Morgan et al. (2004) evaluated on the biologist's annotations and reported 71%/78%/75% (Recall/Precision/F-score). The performance on the same annotation achieved in our experiments is better, mostly due to improved recall.

The large difference in performance measured against each version of the annotation (6.8% in F-score) motivated us to examine the annotation guidelines. The inter-annotator agreement reported in Morgan et al. (2004) was only 87% F-score, which the authors attribute to the difficulty of the task. According to the guidelines used, gene names were tagged not only when they refer to genes, but also when they are part of mentions of proteins or transcripts, as in "the zygotic Toll protein". Only *Drosophila* genes were tagged, excluding reporter genes, genes that are not part of the natural *Drosophila* genome, gene families, particular alleles or protein complexes. However, *Drosophila* genes can be synonymous with foreign genes (e.g. "Hsp90"), family names are often synonymous with specific names (e.g. "CSP"), and foreign and reporter genes are often not mentioned as such in text. Additionally, mutant genes, which are not part of the natural genome, are usually referred to using the name of the original gene, leading to inconsistencies in the annotation of cases like "dunce mutations" or "eye PKCI700D mutant", since one annotator would annotate the gene name only ("dunce" or "PKCI700D") and the other would annotate the whole string. Overall, we observe that the biologist's annotations were more accurate given the annotation guidelines proposed. He avoided tagging reporter genes synonymous with specific ones (e.g. "Gal4"), mutants, or gene families (e.g. "Hedgehog Hh"), tagging fewer genes (909) than the linguist (989). However, the inconsistencies observed motivated us to develop new guidelines and create new datasets in order to evaluate the performance of BioNER systems with greater confidence.

2.3 New guidelines and dataset

In this section we describe new guidelines and a new dataset in order to evaluate named entity recognition for *Drosophila* genes. The guidelines developed were partially inspired by those developed for the ACE project (Doddington et al., 2004). The basic notion is that gene names (in bold in the examples that follow) are annotated in any position in the text, including cases where they are not referring to the actual gene but to a different biomedical entity. As a result, names of gene families, reporter genes and genes not

[5] http://research.nii.ac.jp/~collier/workshops/JNLPBA04st.htm

belonging to *Drosophila* are tagged as gene names:

... the **faf** gene ...
... the **Toll** protein ...
... the **string-LacZ** reporter genes ...

In addition, following the ACE guidelines, for each gene name we annotate the shortest surrounding noun phrase, referred to as the base noun phrase (Lewin, 2007). The latter is broadly defined as the initial portion of a non-recursive noun phrase up to the head noun. We classify base noun phrases into gene mentions and other mentions, depending on whether they refer to a gene or not respectively. In some cases, this distinction can be made by looking at the head noun of the noun phrase. In the examples below the mentions are underlined and their type is in parentheses:

... the **faf** gene... (*gene mention*)
... the **Reaper** protein... (*other mention*)

However, in many cases the base noun phrase itself is not sufficient to classify the mention, especially when the latter consists of the gene name only, because it is common in the biomedical literature to use a gene name to refer to a protein or to other related entities. In order to classify such cases, the context of the mention needs to be taken into account. In the following examples, the word of the context that enables us to make the distinction between gene mentions and other mentions is in italics:

... ectopic *expression* of **hth** ... *(gene mention)*
... *transcription* of **string** ... *(gene mention)*
... **Rols7** *localizes*... *(other mention)*

It is worth mentioning that more than one gene name may appear inside the same noun phrase. As the following examples demonstrate, distinguishing between gene names and mentions enables us to annotate consistently cases of coordination, which is another source of disagreement, as noted by Dingare et al. (2005):

... the homeotic gene **Sex combs reduced** (**Scr**) ...
... **male-specific lethal-1**, **-2** and **-3** genes ...

Following these guidelines, we constructed a new dataset consisting of 82 abstracts from articles curated by FlyBase. We used the RASP tokenizer to process the text, resulting

in 15,703 tokens. The size and the characteristics of the new dataset are comparable with that of Morgan et al. (2004) as can be observed from the statistics reported in Table 2.1, except for the number of non-unique gene names. Apart from the different guidelines, another difference is that we used the original text of the abstracts, without any post-processing apart from tokenization. The dataset of Morgan et al. (2004) had been stripped of all punctuation marks, e.g. periods and commas. Keeping the text intact renders this new dataset more realistic and most importantly, it facilitates the use of tools that rely on this information, such as syntactic parsers.

	auto-annotated abstracts	Morgan et al. (2004)	abstracts	full papers
abstracts/papers	16,609	86	82	5
tokens	2,923,199 (176)	16,779 (195)	15,703 (192)	34,106 (6,821)
gene names	117,279 (7)	1,032 (12)	629 (8)	1,980 (396)
unique gene names	16,944 (1)	347 (4)	326 (4)	336 (67)
English words	60,943 (4)	2,803 (33)	3,018 (37)	4,113 (823)

Table 2.1: Statistics of the datasets mentioned in this chapter. In parentheses are the rounded averages per abstract/paper.

The annotation of the gene names is performed by a computational linguist and a FlyBase curator. We estimate the inter-annotator agreement in two ways. Firstly, we calculate the F-score achieved between them, which is 91%. Second, we use the Kappa coefficient (Carletta, 1996), which has become the standard inter-annotator agreement evaluation metric, and the score obtained is 0.905. This high agreement score can be attributed to the clarification of what a gene name should capture through the introduction of the concepts of gene mention and other mention.

The annotation of the base noun phrases presents greater difficulty, because a computational linguist does not have sufficient knowledge of biology in order to use the context of the mentions, whilst the curator is not trained to identify noun phrases in text. In this effort, the boundaries of the mentions are defined by the computational linguist and the classification is performed by the curator. A more detailed description of the guidelines, as well as the corpus itself tokenized with the entities marked-up using the IOB scheme (**I**nside-entity, **O**utside-entity, **B**egin-entity), are available online.[6]

2.4 Evaluating NER

The standard evaluation metric used for NER is the F-score (Van Rijsbergen, 1979), which is the harmonic mean of recall and precision. It penalizes appropriately systems

[6]http://www.wiki.cl.cam.ac.uk/rowiki/NaturalLanguage/FlySlip/Flyslip-resources

that underperform in either of these two competing criteria, providing an intuitive way of assessing the balance between them. Also, it takes into consideration the existence of multi-token entities by rewarding systems able to identify the entity boundaries correctly and penalizing them for partial matches. In this section we suggest an extension to F-score in order to assess the performance of trainable NER systems in a more informative way.

There are two main requirements for trainable NER systems. The first one is that they are able to identify entities that they have encountered in their training data. This is not always straightforward because in many domains common English words can be (part of) entity names. For example, in *Drosophila* literature "to" is used as a synonym for the gene "take-out". Moreover, the same name can be used to refer to different entity types. For example, the name of a gene can be used to name a protein associated with it, or as part of the name of its mutant. The second requirement is that trainable NER systems are able to learn patterns that generalize to unseen named entities. Features that are dependent on the context rather than the tokens themselves are likely to perform better in this respect. The ability to recognize unseen named entities is very important because it is unlikely that any training dataset can cover all possible cases, especially in domains in which new names are created over time.

A common way of assessing these two aspects is to measure the performance on seen and unseen tokens separately. This is straightforward in tasks with token-based evaluation, such as part-of-speech tagging (Curran and Clark, 2003). However, in the case of NER, this is not entirely appropriate due to the existence of multi-token entities. For example, consider the case of the gene name "head inhibition defective", which consists of three English words that are likely to occur independently of each other in a training dataset. If this gene name appears in the test set but not in the training set, with a token-based evaluation its identification (or not) would contribute towards the performance on seen tokens. Moreover, in such an evaluation, a system would be rewarded or penalized for each of the entity tokens individually. As a result, (not) recognizing any of the tokens of "head inhibition defective" as part of a gene name would result in three (in-) correct answers, thus becoming more important to the evaluation than a single-token entity.

One way to circumvent these problems is to replace all the named entities of the test set with strings that do not appear in the training data, as in Morgan et al. (2004). There are two problems with this approach. Firstly, it changes the morphology of the unseen named entities, which is usually a source of good features to recognize them. Secondly, it alters the contexts in which the unseen named entities occur, which might influence the results inappropriately.

In order to overcome these problems, we separate the gene names that exist in the test set and/or are tagged by the system according to whether they have been encountered in the training data as gene names or not. Then, the standard recall, precision and F-score

	gold	Recall	Precision	F-score
Seen	434	94.48	93.62	94.05
Unseen	195	33.51	68.42	44.98
Overall	629	76.31	89.14	81.86

Table 2.2: Evaluation of the HMM-based system trained on the automatically annotated abstracts and evaluated on the abstracts dataset.

metrics are calculated for each of these two lists independently. So, if "to" has been encountered at least once as a gene name in the training data but an occurrence of it in the test set is erroneously tagged as a gene name, this will count as a precision error on seen named entities. Accordingly, if "to" has never been encountered in the training data as a gene name but an occurrence of it in the test set is erroneously tagged as a common English word, this will count as a recall error on unseen named entities. In a multi-token example, if "head inhibition defective" is a gene name in the test set and it has been seen as such in the training data but the NER system tagged (erroneously) "head inhibition" as a gene name (which is not the training data), then this would result in a recall error on seen named entities and a precision error on unseen named entities.

Using this extended evaluation and the new dataset of abstracts described in Section 2.3, we evaluated the HMM-based BioNER system described in Section 2.2 and the results are in Table 2.2. The large difference in performance on seen and unseen named entities (94.05% versus 44.98% F-score respectively) can be attributed to the highly lexicalized nature of the BioNER system. As explained in Section 2.2, tokens that have not been seen in the training data are passed on to a module that classifies them according to their morphology, which, given the variety of gene names and their overlap with common words, is unlikely to be sufficient. Also, the limited context window used by the tagger (previous label and previous two tokens) cannot capture reliably contexts that are likely to be useful in recognizing unseen gene names.

We believe that this evaluation allows for a fair comparison of the data generation process that created the training data and the HMM-based BioNER system trained on this data subsequently. In order to measure the performance of the dictionary-based data generation process described in Section 2.2, we evaluated it against the same dataset of abstracts using the lists of genes created by the curators. The performance was 73.5%/93%/82.1% (Recall/Precision/F-score). In order to compare it against the HMM-based BioNER system we took into account the performance of the latter only on seen named entities, since the data generation process can be applied only to those abstracts for which lists of the genes mentioned have been compiled manually by the curators. The result of this comparison supports the HMM-based system, which achieves 94.05% F-score compared to 82.1% for the dictionary-based data generation process, mainly due to the improved recall (94.48% versus 73.5%). This is a very encouraging result for bootstrapping techniques

using automatically annotated training material, because it demonstrates that the trained classifier can deal effectively with the noise introduced by the data generation process.

In order to further assess the usefulness of the bootstrapping method employed, we evaluated the performance of the HMM-based BioNER system trained on manually annotated data. For this purpose we used the annotated data from BioCreative Task 1A (Hirschman et al., 2005b). In that task, the participants were requested to identify which terms in a biomedical research article are gene and/or protein names, which is very similar to the task dealt with in this work. Therefore we could expect that, even though the material used for the annotation is not drawn from exactly the same domain as our test data, it should still be useful to train a system to identify gene names. The results were rather disappointing, since the performance achieved was 35.9%/37.4/36.7% (Recall/Precision/F-score). Apart from the domain shift, the deterioration of the performance should also be attributed to the different guidelines used. However, given that the tasks are very similar, it is interesting that manually annotated training material leads to so poor performance compared to using automatically annotated training data. This evidence suggests that manually annotated resources, which are expensive to obtain, might not be ideal even for slightly different tasks than those they were originally created for. Generating large amounts of training material automatically is expected to result in datasets with better coverage of sparse feature sets. For these reasons, the use of semi-supervised methods is worth exploring when moving to new domains, which supports the aims of this thesis.

2.5 From abstracts to full papers

BioNER systems are commonly evaluated on abstracts, as in the work of Morgan et al. (2004), or on sentences selected from abstracts, as in the BioCreative NER shared tasks (Hirschman et al., 2005b; Krallinger and Hirschman, 2007). The main reason for this is that abstracts of scientific papers are publicly available, thus avoiding copyright issues that are common in full papers and consequently inhibit the distribution of datasets based on them. However, in a real world setting BioNER systems are going to be applied to full papers as well, either on their own or in support of more complex tasks. Full papers though are expected to present additional challenges to the systems, so it is important to evaluate on them in order to obtain a clearer picture of the systems and the task, as noted by Ananiadou and Mcnaught (2005).

For this purpose, we use the full paper corpus described in Gasperin et al. (2007). It consists of 5 publicly available full papers which were annotated by a computational linguist and a FlyBase curator with named entities as well as anaphoric relations marked up in XML. The gene names were annotated using the guidelines presented in Section 2.3. In order to use this dataset for gene name recognition experiments, we converted it from XML to IOB format keeping only the gene name annotation. Both versions of the dataset

	gold	Recall	Precision	F-score
Seen	801	91.14	87.32	89.19
Unseen	1,179	37.23	71.38	48.94
Overall	1,980	59.04	80.57	68.14

Table 2.3: Evaluation of the HMM-based system trained on the automatically annotated abstracts and evaluated on the full papers dataset.

(gene names in IOB and gene names, gene/other-mentions and anaphoric relations in XML) are available online.[7]

In Table 2.1 we compare the full paper corpus to the abstract corpus. A first observation is that the gene names in full papers tend to be repeated more frequently than the gene names in the manually annotated abstracts (5.9 compared to 1.9 times on average, respectively). Moreover, the full papers contain more gene names per 100 tokens compared to the abstracts (5.8 versus 4 on average, respectively). The manually annotated abstracts contain approximately 2 unique gene names every 100 tokens while the full papers contain only 1. This evidence suggests that annotating abstracts is more likely to provide us with a greater variety of gene names. Interestingly, the automatically annotated abstracts contain only 0.6 unique gene names every 100 tokens which suggests that many gene names are left untagged. Another observation is that, while the manually annotated abstracts and full papers contain roughly the same number of unique genes, the full papers contain 36% more common English words. This suggests that the full papers contain a greater variety of contexts in which gene names appear, as well as of negative examples, therefore potentially presenting greater difficulty to a gene name recognizer.

We evaluated the HMM-based BioNER system used in the previous sections on the full paper dataset, trained on the large automatically annotated set of abstracts constructed in Section 2.2. The performance was substantially worse, as can be seen in Table 2.3. The overall F-score was 68.14%, far lower compared to the one achieved on abstracts (81.86%). The key reason for this is the low performance on unseen gene names, 48.94% in F-score. While this is not worse than the score achieved for unseen gene names in the abstracts dataset (in fact, it is slightly better), the percentage of unseen gene names in the full papers is much higher, 59.55% versus 31%. Therefore, the performance on unseen gene names has greater influence on the overall performance. In particular, the recall achieved is very low, only 37.23%, which can be attributed to the limited context taken into account by the system.

On the other hand, the performance on seen gene names remains high, 89.19% in F-score, albeit not as high as on the abstracts (94.05%). This is mainly due to the drop in precision from 93.62% to 87.32%, which indicates that common English words which

[7] http://www.wiki.cl.cam.ac.uk/rowiki/NaturalLanguage/FlySlip/Flyslip-resources

have been encountered as gene names in the training data are incorrectly tagged as gene names more frequently in full papers.

Using some examples from these experiments, the word "caspase", which is not always part of a gene name, is very frequently tagged erroneously as such. Moreover, enumerations using letters, as in "panels I-K" are erroneously tagged as gene names, since such cases are unlikely to be encountered in training data derived from abstracts.

2.6 Conditional Random Fields and Syntactic Parsing

From the analysis performed in the previous section, it becomes apparent that the weakness of the HMM-based BioNER system in identifying unseen gene names is very important when applying it to full papers. This motivated us to build a system that would be able to generalize better to unseen gene names. HMMs, while fast and effective, are not very flexible in terms of adding features. While using a higher-order model could be an option, it is unlikely to provide an adequate solution to capturing the appropriate context for recognizing unseen gene names, since not all preceding words are likely to be relevant. It would be more appropriate to selectively add contextual cues relevant to the task. For this purpose we develop a system employing conditional random fields and syntactic parsing which aims at ameliorating this issue.

2.6.1 Conditional Random Fields

Conditional random fields (CRFs) (Lafferty et al., 2001) are undirected graphical models trained to maximize the conditional probability of the output sequence given the inputs, or, in the case of token-based natural language processing tasks, the conditional probability of the sequence of labels y given a sequence of tokens x. Like HMMs, the number of previous labels taken into account defines the order of the CRF model. More formally:

$$P(y|x) = \frac{1}{Z(x)} exp\{\sum_{t=1}^{T}\sum_{k=1}^{K}\lambda_k f_k(y', x_t)\} \tag{2.2}$$

In the equation above, $Z(x)$ is a normalization factor computed over all possible label sequences, f_k is a feature function and λ_k its respective weight. y' represents the labels taken into account as context and it is defined by the order of the CRF. For a n-th order model, y' becomes $y_t, y_{t-1}..., y_{t-n}$. f_k are the features extracted for each token, which can include features extracted by taking the whole input sequence into account, not just the token in question.

CRFs have been used successfully in various natural language processing tasks, including named entity recognition in various domains (McCallum and Li, 2003; Settles, 2004),

```
The D-mib mutant phenotype could be almost fully
rescued by a leaky UAS-D-mib transgene .

(|ncsubj| |rescue+ed:9_VVN| |phenotype:4_NN1| _)
(|aux| |rescue+ed:9_VVN| |could:5_VM|)
(|ncmod| _ |rescue+ed:9_VVN| |fully:8_RR|)
(|aux| |rescue+ed:9_VVN| |be:6_VB0|)
(|passive| |rescue+ed:9_VVN|)
(|iobj| |rescue+ed:9_VVN| |by:10_II|)
(|dobj| |by:10_II| |transgene:14_NN1|)
(|det| |transgene:14_NN1| |a:11_AT1|)
(|ncmod| _ |transgene:14_NN1| |leaky:12_JJ|)
(|ncmod| _ |transgene:14_NN1| |UAS-D-mib:13_NP1|)
(|ncmod| _ |fully:8_RR| |almost:7_RR|)
(|det| |phenotype:4_NN1| |The:1_AT|)
(|ncmod| _ |phenotype:4_NN1| |D-mib:2_NP1|)
(|ncmod| _ |phenotype:4_NN1| |mutant:3_NN1|)
```

Figure 2.1: Sample output from RASP.

shallow parsing (Sha and Pereira, 2003), gene normalization (Wellner, 2005) and coreference resolution (McCallum and Wellner, 2004). Their main advantage is that, as a conditionally-trained model, they do not need to specify the dependencies between features, which, as a consequence, allows the use of features dependent on each other. Compared to HMMs, their main disadvantage is that during training, the computation time required is significantly longer. The interested reader is referred to the tutorial of Sutton and McCallum (2006) for more details.

2.6.2 Syntactic parsing with RASP

In this section we describe how we extracted features from the output of a syntactic parser in order to capture contextual features that would allow us to recognize gene names more effectively. For this purpose we use the domain-independent RASP parser (Briscoe et al., 2006). While it is unlikely to be as accurate as a domain-adapted parser, we believe its use is more realistic given that domain adaptation usually requires annotated data.

Syntactic parsing with RASP relies on an HMM-based part-of-speech (PoS) tagger, which was parameterized to generate multiple PoS tags for each token in order to ameliorate unseen token errors. The syntactic parser uses these sequences of PoS tags to generate parses for each sentence. The output of RASP used is the set of grammatical relations (GRs) for each sentence, which specify the syntactic dependencies between the tokens.

A sample GR output from RASP, without the XML tags for brevity and readability, is shown in Figure 2.1. The first two lines contain the tokens of the sentence, followed by the list of GRs. Each GR contains the PoS tags and the lemmas of the tokens it connects.

The features extracted from RASP's output for each token are listed in Table 2.4. It should be noted at this point that such features may contain noise, since there is normally more than one possible parse for a sentence and the system uses an unlexicalized probabilistic model to select the most likely one.

the lemma and the PoS tag associated with the token
the lemmas for the previous two and the following two tokens
the lemmas of the verbs to which this token is subject (*ncsubj* relation)
the lemmas of the verbs to which this token is object (*dobj* relation)
the lemmas of the nouns to which this token acts as modifier (*ncmod* relation)
the lemmas of the modifiers of this token (*ncmod* relation)

Table 2.4: Features extracted from the output of RASP.

In the sentence of Figure 2.1, apart from the lemmas and the PoS tag for each token, we extract the following features representing relations between them:

- "D-mib" and "mutant" modify "phenotype"
- "phenotype" is the subject of "rescue" in passive voice
- "almost" modifies "fully"
- "fully" modifies "rescue"
- "leaky" and "UAS-D-mib" modify "transgene"

The features that "D-mib" and "UAS-D-mib" modify, "phenotype" and "transgene" respectively, are useful cues in order to recognize them as gene names. These syntactic features are domain-independent, since we only specify the grammatical relations to be considered but not the lemmas of verbs and nouns appearing in them.

2.6.3 Experimental setup

For the experiments of this section we use a second order CRF implementation provided by the publicly available toolkit MALLET (McCallum, 2002).[8] Apart from the syntactic parsing features described in the previous section, simple orthographic features were

[8] http://mallet.cs.umass.edu/

extracted for every token (Table 2.5). Also, we use the IOBEW scheme (Siefkes, 2006) for labeling the resulting tokens – the first token of a multi-token gene name is tagged as **B**egin-entity, the last token as **E**nd-entity, the inner ones as **I**nside-entity, tokens containing a whole gene name as **W**hole-entity and tokens not part of a name as **O**utside-entity. We expect that better performance can be obtained with this scheme than with the standard IOB scheme, due to the large number of multi-token gene mentions and their overlap with common English words or other biomedical terms.[9] For example, the token "son" is unlikely to be a gene name on its own, but it is found as the first token of the gene name "son of sevenless". The IOBEW scheme captures this distinction, since in the former case "son" would have been tagged as **W**-gene while in the latter it is tagged as **B**-gene. On the other hand, under IOB it would be tagged as **B**-gene in both cases.

the token itself	if it contains digit(s)
if it is alphanumeric	if it contains only digits
if it is alphabetic	if it contains dash(es)
if it is titlecase	if it contains dot(s)
if it is lowercase	if it contains any punctuation marks
if it is uppercase	if it contains punctuation marks and digits
if it is mixed case	2 and 3 letter prefixes and suffixes

Table 2.5: Simple orthographic features used in our experiments.

2.6.4 Results

As in the previous sections, we trained the CRF+syntax BioNER system on the automatically annotated abstracts and evaluated it on the abstracts and the full papers datasets. The results of Table 2.6 show that there is a substantial improvement on full papers compared to the HMM-based NER system (Table 2.3) in the overall performance (73.62% vs 68.14% F-score), mainly due to the improved performance on unseen gene names (66.21% vs 48.94%). In the seen gene names, the performance dropped (83.48% vs 89.19%), due to the reduced recall (74.78% vs 91.14%). On the other hand, the precision of the CRF+syntax system was better in all cases, reaching 90.25% on average. The drop in seen gene names recall can be attributed to the additional context features which render the CRF+syntax system more conservative. On the abstracts dataset, the performance of the HMM-based system is superior (Table 2.2), since there the performance on seen gene names matters more, as most of the gene names are encountered in the training data (69%). This was expected, since as noted in Section 2.2, the majority of the gene names

[9]This was confirmed in preliminary experiments during the BioCreative 2 gene mention tagging task by Vlachos (2007).

		gold	Recall	Precision	F-score
Abstracts	Seen	434	76.32	95.4	84.8
	Unseen	195	34.54	73.63	47.02
	Overall	629	63.43	90.89	74.72
Full papers	Seen	801	74.78	94.48	83.48
	Unseen	1,179	53.6	86.58	66.21
	Overall	1,980	62.17	90.25	73.62

Table 2.6: Evaluation of the CRF+syntax system trained on the automatically annotated abstracts and evaluated on the abstracts and the full papers dataset.

found in the dictionary from FlyBase were not found in the abstracts tagged to generate the training data.

The CRF+syntax system uses a complex but more general representation of the context based on the features extracted from the output of the syntactic parser, namely the lemmas, the part-of-speech tags and the grammatical relationships, while the HMM-based system uses a simple morphological rule-based classifier. Also, the former system takes the two previous labels into account, while the latter only the previous one. Therefore, it is expected that the CRF+syntax system has superior performance on unseen genes. This difference between the two systems is more pronounced when evaluating on full papers (65.03% versus 46.66% respectively) than on abstracts (47.02% versus 44.98% respectively). This can be attributed to the fact that the training data used is generated from abstracts, and when evaluating on full papers the change in genre can be handled more effectively by the CRF+RASP system due to its more complex feature set. On the other hand, the simpler HMM-based one is likely to perform better on seen genes, whose effective recognition does not rely on complex features.

In order to assess the contribution of the syntactic features, we trained the CRF+syntax system excluding the features extracted from the GR output of RASP, i.e. the last four features in Table 2.4. The resulting performance of the system was 60.25%/91.84%/72.77% (Precision/Recall/F-score) on full papers, which was lower than the one achieved with these features included (73.62% F-score). As expected, this is mainly due to the lower performance on unseen gene names (62.22% vs 66.21%). Apart from being affected by noise in parsing which reduces their effect, the syntactic features are likely to be useful relatively rarely, since in many cases simpler features suffice. For example, in the sample sentence of Figure 2.1 the link between the gene name "UAS-D-mib" and the rather strongly indicative token "transgene" is also obtained by adding the latter as the token following the former. However in the sentence "Endogenous CED-4 is normally localized...", the feature that "CED-4" is the subject of the verb "localize" in passive voice is crucial in recognizing the former as a gene name.

On both abstracts and full papers, the performance of the HMM-based system is superior on seen genes while the CRF+syntax system performs better on unseen genes. Therefore, it was natural to attempt to combine the strengths of these two systems. In particular, since the HMM-based system is performing very well on seen gene names, for each sentence we check whether it has recognized any gene names unseen in the training data (potential unseen precision errors) or if it considered as ordinary English words any tokens not seen as such in the training data (potential unseen recall errors). If either of these is true, then we pass the sentence to the CRF+syntax system, which has better performance on unseen gene names.

Such a strategy is expected to trade some of the performance of the seen gene names of the HMM-based system for improved performance on the unseen gene names by using the predictions of the CRF+syntax system. This occurs because seen and unseen gene names may appear in the same sentence and choosing the predictions of the latter system could result in more errors on the seen gene names. This strategy is likely to improve the performance on datasets where there are more unseen gene names, since the CRF+syntax system is substantially better than the HMM-based one on them. Using this strategy the overall F-score increased to 75.26% on the full paper corpus which contains a lot of unseen gene names (57% of the total gene names). Out of the 1,220 sentences of the corpus, 759 were passed on to the CRF+syntax system for tagging. For the manually annotated abstracts, 185 out of 600 sentences were passed to the CRF+syntax system and the overall performance was 80.21%, which is lower than the one achieved by the HMM-based system alone (81.86%). This is expected since the majority of gene names (69%) are seen in the training data and the performance of the CRF+syntax system on the unseen data is better than the HMM-based one only by a small margin (47.02 vs 44.98 in F-score respectively).

Overall, the performances on abstracts were better than on full papers, which can be attributed to the fact that the training data used consists of abstracts only. Annotating full papers automatically, as performed in Section 2.2 for abstracts, is unlikely to provide us with training data of adequate quality given the more complex and variable language used in them.

2.7 Using annotation from the users

In this section we attempt to improve the performance of the BioNER systems described in the previous sections by incorporating user feedback. While the performances already achieved are reasonable, we want to explore how they could be improved further, especially on full papers, in which the performance was lower than on abstracts. In the context of the FlySlip project[10] the HMM-based BioNER system described in Section 2.2 was deployed

[10]http://www.wiki.cl.cam.ac.uk/rowiki/NaturalLanguage/FlySlip

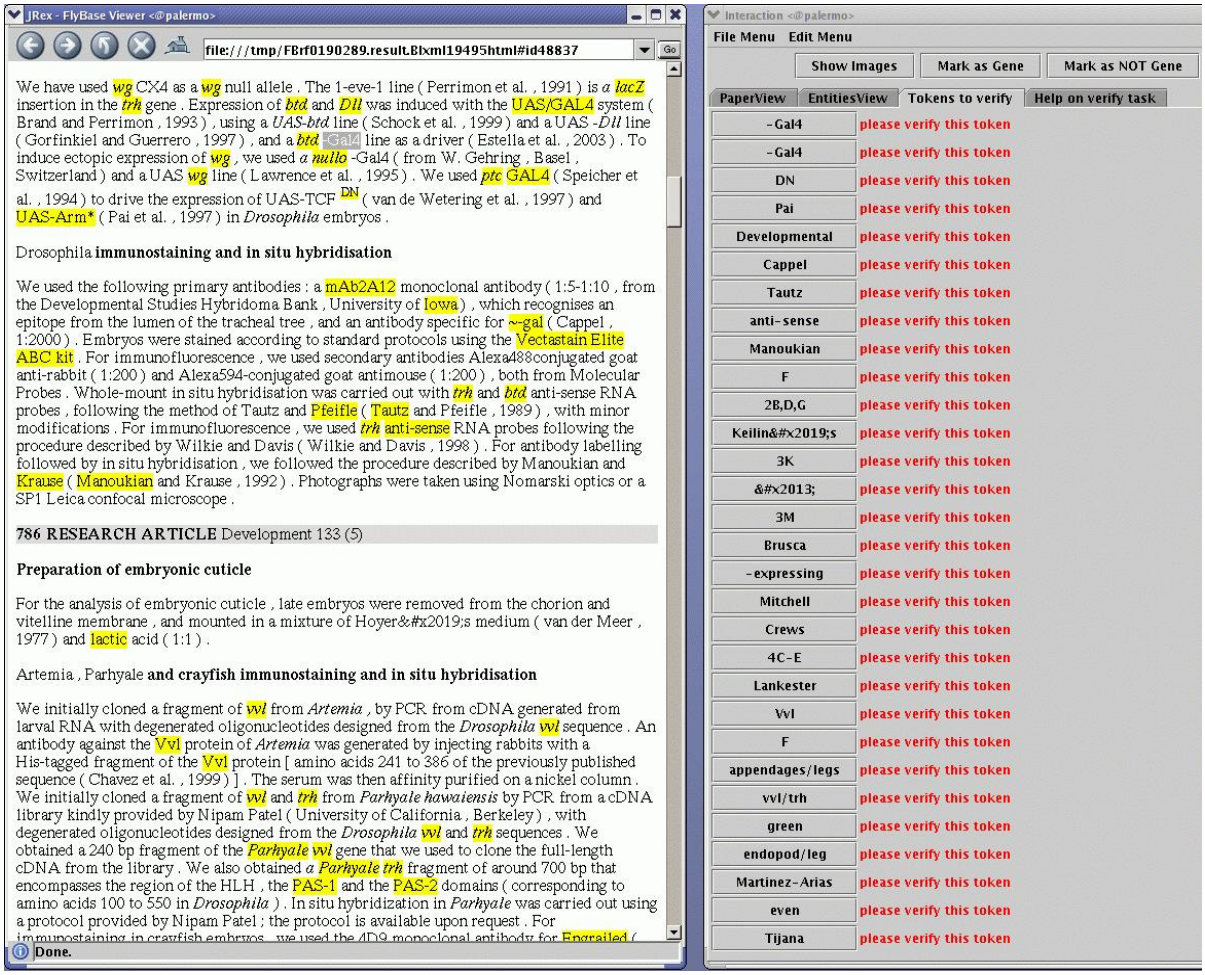

Figure 2.2: Screenshot of the interface used to collect the feedback from the users.

as part of a pipeline which was used to preprocess full papers. They were then curated by FlyBase staff using a specialized curation interface which was developed under a user-centered approach and was shown to improve curation performance (Karamanis et al., 2007, 2008). This process presented us with an opportunity to obtain feedback from the users.

For each article, the curators were asked to give feedback on the annotation performed by the HMM-based BioNER system. A screenshot of the interface appears in Figure 2.2. The tokens to be corrected were chosen according to the uncertainty of the system, which was estimated as the conditional entropy of the decision on the current token given the decision on the previous one. These tokens were chosen independently and no attempt was made to identify sequences of tokens to be corrected by the users. Each chosen token was highlighted and the curators had to confirm whether it was a gene name (by clicking on the "Mark as Gene" button) or not. They were not requested to distinguish between tokens that were gene names on their own and tokens being part of multi-token gene names. If needed, they could also modify the boundaries of the gene name by selecting the appropriate textual string.

	CRF+syntax			HMM		
	R	P	F	R	P	F
seen	79.52%	91.2%	84.96%	92.54%	87.57%	89.99%
unseen	55.45%	92.53%	69.34%	37.94%	63.52%	47.5%
overall	67.02%	91.96%	77.53%	64.19%	78.46%	70.61%

Table 2.7: Evaluation of the performance using user feedback on full papers

In the experiments described below, we collected data from 28 curated papers (other than those included in the full paper gold standard). The curators provided feedback on 677 tokens, resulting in 549 sentences containing 1,668 gene names and 19,449 tokens in total. This feedback was requested once the curators had finished the curation of the paper. The data collected contained noise, since the curators were asked to make decisions on particular tokens, without correcting any other errors found in the sentences containing these tokens. As a consequence, errors that were not related to the token in question were not corrected. We add these sentences to the automatically annotated training data described in Section 2.2. The purpose is to provide the systems with training data from the exact domain they would be applied to, which is the full papers rather than their abstracts.

We re-trained the HMM-based and the CRF+syntax BioNER systems and we evaluated their performance on the full paper corpus described in Section 2.5. As the results in Table 2.7 demonstrate, the improvements achieved by adding the sentences from the full papers are substantial. The performance gains for the CRF+syntax and the HMM-based systems were 3.91% and 2.47% in F-score respectively. This is due not only to the increased coverage of gene names (48.1% were now seen in the training data, compared to 40.5%), but also to the fact that contexts and features useful in recognizing gene names in full papers that had not been encountered in the abstracts were now used. For example, a source of errors for the CRF+syntax system was that identifiers of image panels in parentheses were commonly mistaken for gene names, especially when they followed a gene name, since in abstracts, the pattern "gene_name (token)" is a very strong indication that the token in parentheses is in fact a gene name as well. By adding data from the full papers to the training material, such errors were avoided.

As in Section 2.6.4, we observe again that the performance of the HMM-based NER system is better on seen gene names than that of the CRF+syntax system. Therefore we combine them, keeping the tagging provided by the former for sentences containing only seen gene names and the tagging of the latter for the rest. The performance achieved was 68.64% / 89.94% / 77.86% (Recall / Precision / F-score), or 0.33% better in F-score than the (re-trained) CRF+syntax system on its own. The improvement obtained by combining the systems is smaller than it was in the experiments of Section 2.6 when the feedback

from the users was not used (1.64%). This is due to the fact that the CRF+syntax system took better advantage of the additional in-domain training data, therefore using the predictions of the HMM-based system has a lesser effect. Note that direct comparison of the performances on seen and unseen genes of this section with those reported in Section 2.6.4 is not appropriate, since the training data is different and therefore the gene names considered seen or unseen have changed.

2.8 Discussion - Related work

Overall, the performances achieved are 81.86% and 77.86% F-score in abstracts and full papers respectively, which are comparable to those achieved using manually annotated training data and evaluating on abstracts. For comparison, in the BioCreative 2 Gene Mention tagging task (Krallinger and Hirschman, 2007) the median of the performances achieved was 80.95% F-score. In particular, the CRF+syntax system described in Section 2.6 participated in that task achieving 82.84% F-score (Vlachos, 2007). Even though the evaluation dataset was different, given that no explicitly annotated training material was used in the work presented in this chapter but only a dictionary of gene names associated with raw abstracts and limited user feedback, we believe this is a very strong result.

Incorporating syntactic features for named entity recognition tasks has been attempted by other authors. In the biomedical domain, Finkel et al. (2004) experimented with syntactic features in the context of the BioNLP/NLPBA (Kim et al., 2004) and BioCreative (Hirschman et al., 2005b) shared tasks, reporting that such features were not very useful when only one type of named entity was being recognized. Our results suggest they are useful albeit not very frequently, which can be attributed to the improved annotation guidelines we developed to annotate our datasets. Later work by Smith and Wilbur (2009) reports that syntactic features improve gene name recongntion by a small but statistically significant amount, thus confirming our results. Uzuner et al. (2008) employed syntactic features in order to enhance the performance of a medical record de-identifier based on support vector machines. The latter though are instance-level classifiers and as such they cannot capture the sequential nature of the task, which is the reason we employed HMMs and CRFs in our experiments. Dang and Aizawa (2008) bootstrapped a named entity recognizer using seed named entities to identify syntactic dependency patterns which are used to extract new named entities in an iterative fashion similar to Snowball (Agichtein and Gravano, 2000). However such methods are commonly troubled by errors in early iterations and their coverage can be limited by the fact that named entities do not always appear in an indicative syntactic context. By incorporating syntactic features in a trainable named entity recognizer we avoid relying on syntactic features exclusively. Furthermore, as observed in the experiments of this chapter, the noise introduced by the

automatic annotation of the training data is not detrimental to the performance of our approach.

In this work we focused on using automatically annotated training material. A further step would be to take advantage of unlabeled data without directly labeling it, exploring techniques such as those suggested by Ando (2007) and Blitzer et al. (2006). Another promising direction for future research are the generalized expectation criteria (Druck et al., 2008) that elicit supervision in the form of labels on features rather than instances.

A different direction is to study further the feedback collection process in order to take better advantage of the users' effort. The aim of the approach described in Section 2.7 is to gather user feedback with minimal interruption to the actual curation task. The approach itself can be viewed as one round of active learning with uncertainty based sampling (Cohn et al., 1996) using the tokens of each paper as the pool of unlabeled data. In more recent work, Settles and Craven (2008) explored a range of methods for performing active learning for sequential tagging using CRFs. The reason we did not implement a more principled active learning approach is that while it could lead to better performance for the same number of tokens, in practice it would intrude more on the work of the curators. Implementing a standard active learning framework would require re-training of the system at regular intervals as well as gathering instances from more than one paper in the pool. However, this would mean that the curators would need to read passages of text not related to the curation of the paper they are looking at, since the context is very often needed in order to annotate a token as a gene name or not. Furthermore, in order to apply the more advanced techniques described by Settles and Craven (2008) the users would need to annotate whole sentences, while in our experiments they were required to annotate a single token or entity.

The goal of our approach was to take advantage of the application context in order to obtain the users' feedback with minimal interruption to the curation process and use it to complement the automatically annotated data. While it is unlikely to be optimal from a machine learning perspective, the results obtained in practice demonstrate its potential, and the curators were happy to provide their feedback in this fashion, since it involved minimal additional effort. The results reported here are significant because they were obtained by applying the techniques presented in a real working environment through an appropriately designed interface. The users were not involved in the development and they provided their feedback without being distracted from their work.

A related approach is corrective feedback (Culotta et al., 2006). This scenario involves curating records in order to extract information using a CRF model trained on manually annotated data and feedback from the users. The main difference is that the annotation unit used consists of a paragraph, thus the users need to correct a text passage much longer than a named entity, which was the case in our experiments. While correcting a longer passage, such as a sentence, would deal with the issue of noise, we opted not to use

it because it would increase the amount of work for the curators. Another difference is that user feedback is used to adapt the output of the system, which would be an interesting direction for future work. The fact that positive results were obtained, even if the method employed was not optimal from a machine learning perspective, is very promising with respect to applying such methods in real-world situations.

On a related issue, the effect of the varying difficulty in annotating different examples has been studied by various authors (Baldridge and Osborne, 2004; Hachey et al., 2005; Culotta et al., 2006). We did not consider it in our experiments mainly because the biologists would provide their feedback once they had finished the curation of the paper. Consequently, they would be familiar with the text and the gene names mentioned so that the corrections required should not differ greatly in difficulty for them.

Finally, we restricted our focus to the recognition of gene names. The task of building a classifier to distinguish between the two types of mentions (*gene-mentions* and *other-mentions*) was tackled by Korkontzelos et al. (2007), and can be viewed as a domain-specific version of semantic type induction (Ng, 2007).

2.9 Summary

In this chapter we explored the potential of using a dictionary of gene names to generate training material in order to perform biomedical named entity recognition. We re-implemented previous work and its evaluation led to improved guidelines and annotated datasets consisting of both abstracts and full papers. The low performance on the latter was improved by developing a BioNER system involving CRFs and syntactic features. Finally, the performance was improved further by obtaining feedback from users in the context of a real working environment, with minimal disruption to the users' actual work.

Chapter 3

Biomedical event extraction

3.1 Introduction

The term *biomedical event extraction* is used to refer to tasks whose aim is the extraction of information beyond the entity level.[1] It commonly involves recognizing actions and relations between one or more entities. Some incarnations of biomedical event extraction are inspired by and therefore closely resemble curation tasks, while others are geared towards annotating events in the text. Both approaches to the task are of great interest to the biomedical community, since the first would allow the effective population of curation databases, while the second would enable searching for information that is beyond the entity level, e.g., one could search for causes of regulation of a certain protein.

Extraction of relations between entities relies heavily on the successful recognition of the entities involved. Therefore it started receiving more attention recently, as the performances on biomedical named entity recognition reached adequate levels. Indicatively, the best performance in the BioCreative II Gene Mention task was 87.21% in F-score, while the top-10 systems in the same task achieved F-score higher than 80%. An early approach was the Learning Language in Logic 2005 (LLL 2005) Genic interaction shared task (Nédellec, 2005), which focused on protein-protein interactions (PPI). However, the datasets involved were rather small in size (55 sentences and 103 interactions for training, 80 sentences and 54 interactions for test), not allowing confident conclusions on system performances. LLL 2005 was followed by the protein-protein interaction pair subtask of BioCreative II (Krallinger et al., 2008). In the latter, the annotated datasets provided were produced by extracting curation information from relevant databases. This meant that there was no text-bound annotation, thus making the application and evaluation of existing NLP techniques difficult, resulting in rather low performances. Indicatively, the best performance achieved was 29% in F-score, while many of the teams scored below 10%. More recently, the BioNLP 2009 shared task on event extraction (Kim et al., 2009)

[1] The term *biomedical relation extraction* is also used to describe the same kind of tasks.

focused on a number of relations of varying complexity using a text-bound annotation scheme.[2] The performances achieved ranged from 16% to 52% in F-score, suggesting improvements in task definitions, data annotation and participating systems.

These community-wide efforts were preceded by some independent ones, including the systems described by Humphreys et al. (2000), Thomas et al. (2000) and Hobbs (2002). More recently, Bunescu et al. (2005) compared a variety of methods for protein-protein interactions with their main finding being that methods employing machine learning performed better than rule-based ones. Fundel et al. (2007) applied a small set of rules operating on syntactic dependencies in order to extract relations between genes and proteins. Bundschus et al. (2008) proposed a cascade of conditional random fields in order to identify relations between genes and diseases. Finally, Roberts et al. (2008) report that their machine learning-based approach to relation extraction achieved performance close to human inter-annotator agreement on a corpus of oncolgy narratives. While these independent efforts helped advance research in the field, the variation in datasets and annotation guidelines renders direct comparisons between them rather difficult.

In this chapter we focus on event extraction in the context of the BioNLP 2009 shared task. We implemented two approaches for the task, in both cases aiming to minimize the annotation effort required for system development, in line with the goals set in Chapter 1. The overall approach for event extraction is similar to the approach presented in Chapter 2. We start with raw abstracts and a dictionary of terms that denote events. Instead of generating noisy training data for a supervised classifier in order to identify event triggers, we take advantage of the fact that the domain restricts the semantic content of these terms and use the triggers extracted by the dictionary directly (Section 3.3.1). Following this, we apply an unsupervised rule-based method in order to identify event arguments (Section 3.3). The goal we set for it was to avoid using training data explicitly annotated for the task. While we acknowledge the utility of supervision (in the form of annotated data), we believe it is valuable to explore an unsupervised rule-based approach. Firstly, manually annotated data is expensive to create and the annotation process itself is difficult and unavoidably results in inconsistencies, even in well-explored tasks such as named entity recognition (NER). Secondly, unsupervised approaches, even if they fail to reach the performance of supervised ones, are likely to be informative in identifying useful features for the latter. Thirdly, exploring the potential of such a system may highlight what annotated data is useful and its potential contribution to performance. Building on the results and experience of the rule-based approach, we replace the rule-based component with support vector machine classifiers (Section 3.5). The training data is produced by seeking partial event annotation in the form of trigger-argument associations, making use of the dictionary-based event trigger extraction. This partial annotation is somewhat unconventional with respect to the typical semi-supervised learning framework which as-

[2]The exception to this is one of the optional subtasks which did not have a text-bound scheme. This is discussed in more detail in Section 3.2.1

sume the use of a few fully labeled examples. Nevertheless, we argue that the partial event annotation used in this chapter lies within this framework, as it aims at reducing the annotation effort by not requiring fully annotated training data.

3.2 The BioNLP 2009 shared task

3.2.1 Definition

The BioNLP 2009 shared task (Kim et al., 2009) focused on extraction of events involving proteins. The definition of protein is rather broad encompassing both gene and protein entities, roughly following the annotation of the GENETAG corpus (Ohta et al., 2009). For the purposes of the shared task, the recognition of protein names was considered a given not only due to the high performances reported in the literature, but also in order to focus the research efforts on the novel aspects of the task. It took place over a period of 12 weeks and 24 teams submitted final results.

The BioNLP 2009 shared task was divided into 3 subtasks, the obligatory core event detection subtask (Task 1) and the optional event enrichment (Task 2) and negation/speculation (Task 3) subtasks. Task 1, in which all teams participated, involved the extraction of the primary arguments of 9 different event types. Task 2, with 6 participants, involved the extraction of the secondary arguments for some of the events extracted in Task 1. Task 3, with 6 participants (not the same as those of Task 2) involved the detection of whether the events extracted in Task 1 were negated or speculated upon. An overview of the event types and their arguments appears in Table 3.1. Every event has a trigger which is the textual string denoting the event. Triggers and arguments can be shared across events, a phenomenon commonly observed when co-ordination is involved. Furthermore, the same textual string can be a trigger for events of different types. Finally, event triggers and arguments are contiguous strings that can span one or more tokens, as well as a part of a token.

In order to represent the event components (triggers and arguments), a text-bound stand-off annotation scheme was employed. Each textual string relevant to the events annotated has an entry with a unique identifier prefixed by `T`, its text span and its type. For Tasks 1&2, event triggers, protein names and secondary arguments are annotated in this fashion. Each event is represented by a unique event identifier prefixed by `E`, its trigger and its argument(s). However, for Task 3 the speculation or the negation is not annotated in the text but only as a modifier on an existing event. In other words, the textual string that causes the annotation of an event as negated or speculated is not annotated, which is acknowledged by the organizers as the most likely reason for the low performances on this subtask (Kim et al., 2009).

Event class	Event type	Primary arguments	Secondary arguments
SIMPLE	Gene expression	Theme(P)	
	Transcription	Theme(P)	
	Protein catabolism	Theme(P)	
	Phosphorylation	Theme(P)	Site
	Localization	Theme(P)	AtLoc, ToLoc
BINDING	Binding	Theme(P)+	Site+
REGULATION	Regulation	Theme(P/E), Cause(P/E)	Site, CSite
	Positive Regulation	Theme(P/E), Cause(P/E)	Site, CSite
	Negative Regulation	Theme(P/E), Cause(P/E)	Site, CSite

Table 3.1: Description of the event types involved in the BioNLP 2009 shared task. In parentheses appear the argument types: P=protein, E=Event. All the secondary arguments are of the generic type Entity.

In what follows, we provide examples for each class of events which will help clarify the annotation scheme used as well as highlight some aspects of the task. In all the examples, the protein names and entities are in bold and the triggers are underlined. The textual spans are omitted, since these refer to absolute positions in the abstracts containing the examples.

- In Task 1, Simple events take one Theme argument which is always a protein name, e.g.:

  > However, the HTLV-I genes including **Tax** are not expressed significantly in primary leukemic cells from ATL patients.

  The gold standard annotation for the event of the above example is:

  ```
  T1 Protein Tax
  T2 Gene_expression expressed
  E1 Gene_expression:T2 Theme:T1
  ```

  During testing, the systems were provided with the line(s) containing the protein name(s) (the first line in the example above) and they need to generate the lines representing the event(s). An event must be extracted only if it involves a protein name. In the example above, if "including **Tax**" had been omitted an event should not be extracted since "HTLV-I genes" is not annotated as a protein name. As expected, not all annotated protein names participate in events. Finally, all the protein names in an abstract are annotated in the the data, independently of whether they participate in an event.

Phosphorylation and Localization events receive optional secondary arguments in Task 2, e.g.:

> ... **nuclear** translocation of only a fraction of the entire cytoplasmic pool of **RelA**.

The gold standard annotation for the event of the above example is:

```
T1 Entity nuclear
T2 Localization translocation
T3 Protein RelA
E1 Localization:T2 Theme:T3 ToLoc:T1
```

It is worth pointing out that, unlike protein names, entities are not provided to the systems during testing. Furthermore, in contrast with protein names again, during training only the entities that participate in events are provided.

- Binding events have one or more protein names as Themes, and for Task 2 they can have optional Site arguments, e.g.:

> The **KBF1** factor, which binds to the **enhancer A** located in the promoter of the mouse MHC class I gene **H-2Kb**, ...

The gold standard annotation for the event of the above example is:

```
T1 Protein KBF1
T2 Binding binds
T3 Entity enhancer A
T4 Protein H-2Kb
E1 Binding:T2 Theme:T1 Theme2:T4  Site2:T3
```

The numbering of the Theme arguments is not used in Task 1, since no distinct roles are assigned to them. It is only used with respect to Task 2, in which each Binding Theme can be associated with a Site argument. In the example above, "enhancer A" is annotated as `Site2` since it is located on "H-2kb" which is annotated as `Theme2`, but they could have been annotated as `Site` and `Theme` respectively.[3]

Whether two protein names associated with the same binding trigger form a single event or multiple events depends on whether proteins are bound together or independently, e.g.:

> **TRADD** was the only protein that interacted with wild-type **TES2** and not with isoleucine-mutated **TES2**.

The gold standard annotation for the event of the above example is:

[3]This change would require "KBF1" to be annotated as Theme2 .

```
T1 Protein TRADD
T2 Protein TES2
T3 Protein TES2
T4 Binding interacted
E1 Binding:T4 Theme:T1 Theme2:T2
E2 Binding:T4 Theme:T1 Theme2:T3
```

The two occurrences of the same protein name ("TES2") are distinguished by their spans (not shown here). Furthermore, this example contains a negation, which is the focus of Task 3. In particular, the following line is added to denote the fact that event E2 is negated:

```
M1 Negation E2
```

- Regulation events can have other events as well as protein names as their Theme, which results in nested events. Furthermore, they have an optional Cause argument that can be either a protein or an event, e.g.:

> ...SQ 22536 suppressed **gp41**-induced **IL-10** production in monocytes.

The gold standard annotation for the events of the above example is:

```
T1 Protein gp41
T2 Protein IL-10
T3 Negative_regulation suppressed
T4 Positive_regulation induced
T5 Gene_expression production
E1 Negative_regulation:T3 Theme:E2
E2 Positive_regulation:T4 Theme:E3 Cause:T1
E3 Gene_expression:T5 Theme:T2
```

This example demonstrates the complexity of the task, since participating systems, given the two annotated protein names, need to generate three appropriately nested events. Furthermore, the annotated textual strings (protein names and triggers) can span over multiple, possibly partial tokens, as in the case of "gp41-induced" above.[4]

A different aspect of the task is that the components of an event can reside in different sentences of the abstract, thus giving rise to events through anaphoric relations. The most

[4]In this example and throughout this chapter we use the token boundaries provided by the shared task organizers.

common case involves anaphoric links between a protein name and some referring expression, commonly an appropriate pronoun or definite description, such as "the protein". A further anaphoric phenomenon involved coreference between events, for example:

> ... the expression of **LAL-mRNA** is induced. This induction is dependent on...

The induced event ("expression of LAL-mRNA") is described in the first sentence but it acts as the theme of the Regulation event in the second sentence which is triggered by the word "induction". Anaphoric relations were not annotated in the data, with the exception of protein names referring to the same entity which are commonly introduced via appositive relations, e.g.:

> **IL-4 Stat**, also known as **Stat6** ...

The gold standard annotation for the above sentence contains the following lines:

```
T1 Protein IL-4 Stat
T2 Protein Stat6
* Equiv T1 T2
```

Finally, the same textual span can be annotated with more than one trigger. This annotation is used for triggers that give rise to more than one event of different types. Commonly, this involves a textual string being annotated as trigger of a Simple event and a Regulation event, with the former event becoming the Theme argument of the latter. For example:

> Although overexpressed **Egr-1** was ineffective ...

The gold standard annotation for the above sentence contains the following lines:

```
T1 Protein Egr-1
T2 Positive_regulation overexpressed
T3 Gene_expression overexpressed
E1 Gene_expression:T3 Theme:T1
E2 Positive_regulation:T2 Theme:E1
```

3.2.2 Evaluation

The evaluation of the shared task is based on the standard Recall/Precision/F-measure set of metrics. It takes into account only whole events and ignores the extraction of individual components, i.e. protein names and triggers (and entities for Task 2 only).

For an extracted event to be considered as a True Positive, all its components must be extracted correctly. Consequently, since protein names are given, extraction of triggers and entities, while necessary in order to construct events, is not evaluated separately. Instead, a few (relaxed) variants of what is considered to be a correctly identified event are employed, which allow for more flexible evaluation of the results. More specifically:

- Strict matching: An event is considered extracted correctly only if all its components are identified with their exact spans. In case of nested events, for the top-level event to be correct all the events being referred to need to be correctly identified. This is the strictest criterion employed in the evaluation.

- Approximate span matching: An event is considered extracted correctly if all its components are extracted with spans within an one-token extension of the correct spans. This means that a component with span $begin, end$ is correctly identified if $begin \geq gold\_begin - previous\_length$ and $end \leq gold\_end + next\_length$, where $gold\_begin, gold\_end$ are the span of the gold standard annotation and $previous\_length$ and $next\_length$ are the lengths (in characters) of the previous and the next token respectively.

- Approximate Span Matching/Approximate Recursive Matching: Like the approximate span matching with the added flexibility that in case of nested events, the events being referred to need only their Theme argument to be correctly identified so that the top-level event is considered correct. In the regulation event example mentioned earlier, event `E1` would be considered correctly identified even if the Cause argument of event `E2` is incorrect or missing. This would not hold in strict matching. For the purposes of the shared task, this variant was used as the main performance criterion.

- Event Decomposition/Approximate Recursive Matching: Like the approximate recursive matching with the added flexibility that events with multiple arguments are reduced to multiple events with single arguments. In the second of the two Binding example mentioned earlier, this would result in three single Theme Binding events (one per protein name), while in the Regulation example this would result in event `E2` being decomposed in two events with the same trigger, one for the Theme and one with the Cause argument.

- Event Decomposition/Approximate Span Matching/Approximate Recursive Matching: It is a combination of all three aforementioned relaxations, thus resulting in the most lenient evaluation.

3.2.3 Datasets and resources

The shared task involved three datasets, training, development and test, which comprised of 800, 150 and 260 abstracts respectively. These abstracts were taken from the GENIA event corpus (Kim et al., 2008), but their annotation was tailored to the shared task definition as described in Kim et al. (2009). More detailed information on the events appearing in these datasets appears in Table 3.2.

Event Type/Class	train	development	test
Gene_expression	1738	356	722
Transcription	576	82	137
Protein_catabolism	110	21	14
Phosphorylation	165	47	135
Localization	263	53	174
SIMPLE	2852	559	1182
BINDING	880	248	347
Regulation	960	169	291
Positive_regulation	2843	617	983
Negative_regulation	1062	196	379
REGULATION	4865	982	1653
TOTAL	8597	1789	3182

Table 3.2: Statistics of the data sets involved in the BioNLP 2009 shared task.

During the shared task, the participants were provided with the full annotation for the training data, and the protein name annotation for the development and test data. Furthermore, an on-line evaluation server allowed the participants to assess their performance on the development data. After the official results of the shared task were announced, the full annotation of the development data became available and an on-line evaluation server was activated in order to allow the evaluation on the test data. Registered users of this service were permitted to submit new runs for evaluation at least 24 hours after their last submission. This restriction was placed in order to avoid systematic tuning of the systems on the test data, thus allowing for comparisons with the shared task participants who had access to the test data evaluation only after the end of the shared task. For parameter tuning and feature selection, researchers were encouraged to use the development data.

An additional resource made available for the task was the output of four syntactic parsers for all datasets, training, development and test. The output of the parsers was provided in their respective native formats as well as in the form of syntactic dependencies. In particular, the parsers considered were (the short names used in later sections appear in parentheses):

- Bikel's re-implementation of Collins' parsing model (Bikel, 2004) (Bikel). This parser is trained on newswire data exclusively.

- The re-ranking parser of Charniak and Johnson (2005) adapted to the biomedical domain using self-training (McClosky and Charniak, 2008) (McClosky). More specifically, this parser uses the in-domain part-of-speech (PoS) tagger developed by Lease and Charniak (2005) trained on the GENIA corpus PoS annotation (Kim et al., 2003; Tateisi and Tsujii, 2004) (18,500 sentences) and the self-training process used a random selection of un-annotated Medline abstracts (270,000 sentences).[5] Data from the GENIA treebank (Tateisi et al., 2005) was used as development data in their experiments.

- The C&C Combinatory Categorial Grammar (CCG) parser (Curran et al., 2007) adapted to the biomedical domain by Rimell and Clark (2009) (CCG). The adaptation involved re-training the PoS tagger and the CCG supertagger using in-domain resources. For the former, the GENIA corpus PoS annotation (Kim et al., 2003; Tateisi and Tsujii, 2004) was used, while for the latter 1,000 sentences from GENIA were annotated with lexical categories. The third component which derives the actual parse tree using the information from the other two components was not adapted. For development, 600 sentences of the BioInfer corpus (Pyysalo et al., 2007) were used.

- The GDep dependency parser (Sagae and Tsujii, 2007) trained for the biomedical domain in the experiments of Miyao et al. (2008) (GDep). In this case, the parser was trained exclusively for the biomedical domain using the GENIA treebank (Tateisi et al., 2005), which consists of 500 abstracts annotated following a scheme based on Penn TreeBank II (Beis et al., 1995).

The native Penn TreeBank output of Bikel's and McClosky's parser were converted to the Stanford Dependency collapsed dependency format (de Marneffe and Manning, 2008) using the Stanford tools.[6] The output of the CCG parser was also converted to the same dependency format, while the output of GDep was provided in a different dependency format used for the dependency parsing CoNLL shared task. The sentence splitting and tokenization for all parsers was performed using the GENIA Sentence Splitter and the GENIA Tagger respectively, provided by U-Compare (Kano et al., 2009).

From the description above, it becomes apparent that the various parsers used have different levels of adaptation to the biomedical domain. While it is difficult to assess quantitatively the actual annotation effort involved, it is possible to make some comparisons. Clearly, Bikel's parser is not adapted to the domain, therefore it would be the cheapest one to deploy. McClosky and CCG use in-domain corpora annotated with PoS tags for

[5] http://www.ncbi.nlm.nih.gov/PubMed

[6] http://nlp.stanford.edu/software/lex-parser.shtml

training, with the latter using some additional annotation for lexical categories. Furthermore, they were tuned using in-domain syntactic treebanks. Therefore, they represent a more expensive option in terms of annotation cost. Finally, GDep was trained (and tuned) using an in-domain treebanked corpus, thus representing the alternative with the highest annotation cost. It is important to remember these costs need to be considered each time we change domains. Nevertheless, annotation cost is likely to correlate with performance in both intrinsic evaluations (i.e. syntactic parsing quality) and extrinsic ones (e.g., as a component of an event extraction system). The former was explored by Clegg and Shepherd (2007), while the latter was the focus of Miyao et al. (2008) in the context of the protein-protein interaction task. In Section 3.4 we combine the rule-based event extraction system described in the next section with these parsers, assessing the performance taking into account the annotation costs, which is the goal of this thesis.

3.3 System description

The architecture of the system developed for event extraction is the following. Initially, event triggers are identified and labeled with event types using a dictionary (Section 3.3.1). Based on the dependency output of a syntactic parser, the triggers are associated with candidate arguments using either a simple set of rules (Section 3.3.2), or a combination of trained Support Vector Machine classifiers (Section 3.5). Finally, the triggers connected with appropriate arguments are post-processed to generate the final set of events (Section 3.3.3). Experiments with an initial version of the rule-based system appear in Vlachos et al. (2009). Each of these stages are described in detail in subsequent sections, followed by experiments and discussion.

We only considered extraction of events that are contained in a single sentence. As described in the shared task definition, the anaphoric phenomena involved are rather complex and participants were not provided with appropriate data for development of such components. Even if an anaphora resolution was available, apart from introducing some noise, it would also propagate errors of the event extraction system. Given the rather low overall performances achieved in the task, its effect on performance could even be a negative one. Further support to this decision is that the majority of the participants did not employ anaphora resolution in their systems, as well as the very limited impact observed in initial experiments reported by Vlachos et al. (2009) who developed a simple anaphora resolution module. It must be noted that events involving anaphoric relations contained within a single sentence can still be handled via syntactic dependencies, but they are likely to be harder to extract correctly.

3.3.1 Trigger extraction

We perform trigger identification using a dictionary of terms each of which is associated with the event type (or the combination of event types) it indicates. The underlying assumption is that a particular lemma has the same semantic content in every occurrence, which results in extracting all of its occurrences as triggers of the same event type. This is clearly an over-simplification, but the restricted domain and the task definition alleviate most of the problems caused. Most importantly, it relieves us from requiring explicitly annotated triggers in the data. It is important to note that while in general English events are denoted mainly using verbs, in the biomedical literature verb nominalizations are employed very frequently for the same purpose, as shown by Cohen et al. (2008). Therefore, we did not restrict the dictionary to contain verbs only. In order to construct the dictionary of terms with their associated event types we use the trigger annotation from the training data, but we argue that such information could be obtained from domain experts. Such efforts can take advantage of semantic verb clustering, which is explored in detail in Chapters 4 and 5.

Another issue is that some terms (e.g. "expression") have different senses but in the context of the shared task they are annotated as a trigger only when they denote a biomedical event with an appropriate argument. Using a dictionary, we would be extracting all their occurrences in text as triggers, therefore over-generating triggers, since not all occurrences denote a biomedical event. This can be either because they are not connected with appropriate arguments or because they are used with a sense irrelevant to the task. However, at the argument identification stage not all occurrences will receive arguments and as a result a substantial number of them will eventually be ignored, including those that are irrelevant to the task. For example, if an occurrence of "expression" has a protein name in an appropriate argument position then it is rather certain it is used with its biomedical sense. While this effectively postpones the resolution of ambiguous trigger words until the argument identification stage, it proves to be rather effective in our experiments.

The one-sense-per-term assumption is further challenged by the fact that occurrences of the same term can denote events of different types. For example, "expression" is used as a trigger of four different event types in the training data, namely Gene_expression, Transcription, Localization and Positive_regulation. While it can be argued that in some cases this is due to annotation inconsistencies (see further discussion in Section 3.7), it is generally accepted that context can alter the semantics of a token. Such phenomena cannot be handled appropriately using a dictionary-based tagger.

In order to ameliorate this problem we employed the concept of trigger transparency. It was inspired by the observation on the training data that some terms associated with a particular event type, when modified by a term associated with a different event type, denote events of the type of their modifier instead of their own. For example, "regulation" generally denotes Regulation events, unless it has a modifier of a different event type, e.g.

"positive". In these cases, "regulation" becomes part of a multi-token Positive_regulation trigger (e.g. "positive regulation"). However, if the actual tokens are not adjacent, only "regulation" is annotated as a Positive_regulation trigger, which is due to the requirement that triggers (and T-entities in general) are contiguous textual strings. From this point onwards we will refer to terms that change their triggered event type according to their modifier as *semi-transparent triggers*.

Additionally, we observed that some lemmas were triggering events only when being modified by another lemma associated with an event type. For example, "activity" when occurring without a modifier was not considered a trigger of any event, however, when modified by "binding" then it becomes a Binding event trigger, similarly to the "positive regulation" example. We will refer to lemmas exhibiting this behaviour as *ultra-transparent triggers*.[7]

Implementation-wise, the *transparency* markers become part of the event type associated with a term in the dictionary. They take effect at the argument identification stage (Section 3.3.2), since we need dependency parsing in order to detect modifications.

The process used has the following steps:

1. Removal of triggers encountered only once in the data.
2. Lemmatization of the triggers with the morphology component of the RASP toolkit *morpha* (Minnen et al., 2001).[8]
3. Removal of stop-words such as prepositions.
4. Using the single-token triggers only, we associate each lemma with its most common event type. In cases where a lemma was consistently generating more that one event trigger of different types (typically one of the Simple event class and one of the Regulation class), we associate the lemma with all the relevant event types.
5. Using the last token of each multi-token trigger, we compile a list of the *semi-transparent* triggers.
6. If the lemma is encountered as part of multi-token triggers of a different event-type more often than with the event type associated with it as a single-token trigger, then it is characterized as an *ultra-transparent* trigger.

In the first step, we remove triggers encountered only once in the data in order to avoid processing non-indicative triggers. We avoid normalizing the triggers further, e.g. by applying stemming, because suffixes distinguish lemmas in an important way given the

[7] Kilicoglu and Bergler (2009) made similar observations, they only report observations on the lemma "activity" item without formalizing them.

[8] http://www.cogs.susx.ac.uk/lab/nlp/rasp/

task. For example, "activation" denotes Positive_regulation events, while "activity" as discussed earlier is an *ultra-transparent* trigger whose event type is determined by its modifier. We only keep lemmas which were associated at least 4 times with a particular event type, since below that threshold the annotation was deemed rather inconsistent.

During testing, using the tokenized text provided, we attempt to match each token with one of the lemmas associated with an event type. We perform this by relaxing the matching successively, using the token lemma first and if no match is found allowing a partial match in order to deal with particles (e.g. so that "co-express" matches "express"). This process returns single-token triggers, some of which are processed further according to their transparency properties in the following stage.

It must be noted that this process does not aim to reproduce faithfully the trigger annotation of the data. Apart from the issue of overgeneration discussed earlier, some events have multi-token triggers that are not due to transparency properties. Sometimes they include prepositions or other terms that are considered stopwords ("in response to") and in some (rare) cases they are whole phrases ("have a prominent increase"). We did not attempt at capturing such cases because in most cases the actual trigger tends to be a single term which is not a stopword. We do not expect such phenomena to affect our performance, since the majority of the triggers consist of one token and multi-token triggers affect performance only when strict matching criteria are used for evaluation.

3.3.2 Rule-based argument identification

Given a set of extracted triggers, we connect them with appropriate arguments using syntactic dependencies. For this purpose, in this section we develop a set of simple unlexicalized rules. Considering the task complexity, a rule-based approach is unlikely to achieve state-of-the-art performance. Nevertheless, we implemented it in order to gain a better understanding of the task. Most importantly, we want to explore the potential of a method that does not require training data to be developed, which apart from being cheaper in terms of annotation cost, will allow us to identify the issues that can be ameliorated using training data.

The rules were implemented using the Stanford Dependency scheme (de Marneffe and Manning, 2008). This is a hierarchy of grammatical relations which capture syntactic dependencies between the tokens in a sentence. These relations are expressed in the form of *relation_type(governor, dependent)*. We used the Stanford Dependency scheme in order to take advantage of the output of the different parsers provided by the organizers. The rules were developed using the development data provided by the organizers. This does not contradict the goals of our approach because we do not require annotated data for training, while development data is always needed in order to develop a system.

The rules developed define syntactic dependency paths that connect tokens containing

triggers (*trigger-tokens*) with tokens containing their arguments (*arg-tokens*). For multi-token protein names, it is sufficient that a path reaches any of its tokens. Note that we refer to tokens containing the triggers and the protein names since both can be annotated at the sub-token level. Finally, for Regulation event class triggers (denoting Positive_regulation, Negative_regulation and Regulation event types) we consider as *arg-tokens* not only tokens containing (parts of) protein names but also the *trigger-tokens* found in the same sentence. For triggers known to trigger multiple events from the analysis performed in Section 3.3.1, we treat them as Simple event class triggers at this processing stage. The rules defined are the following:

- If a *trigger-token* is the governor of an *arg-token* in subject relation, then the latter is identified as the Theme argument of the former, e.g. "**Stat1** expresses". In the Stanford Dependency Scheme, the subject relation (*subj*) subsumes the following relations: nominal subject (*nsubj*), passive nominal subject (*nsubjpass*), clausal subject (*csubj*) and passive clausal subject (*ncsubjpass*). The only exception to this rule is that when the trigger denotes Regulation class events and the nominal subject relation (*nsubj*) is observed, the *arg-token* is identified as a Cause argument, e.g. "**gp41** induces".

- If a *trigger-token* is the governor of an *arg-token* in a prepositional relation, then the latter is identified as the Theme argument of the former, e.g. "expression of **Stat1**". In the Stanford Dependency scheme with collapsed dependencies, prepositional relations are of the form *prep_*(governor, dependent)*, where * is replaced by the actual preposition used, e.g. *prep_of(expression, Stat1)*. Note that in the basic version of the scheme a prepositional relation is expressed using a prepositional modifier relation (*prep*) and an object of preposition relation (*pobj*) instead.

- If a *trigger-token* is the governor of an *arg-token* in modifier relation then the latter is identified as the Theme argument of the former, e.g. "**Stat1** expression". We restrict the definition of the modifier relation to subsume only the following relations: adjectival modifier (*amod*), infinitival modifier (*infmod*), participial modifier (*partmod*), adverbial modifier (*advmod*), relative clause modifier (*rcmod*), quantifier modifier (*quantmod*), temporal modifier (*tmod*) and noun compound modifier (*nn*) relations. This restriction is placed in order to avoid matches irrelevant to the task.

- If a *trigger-token* is the governor of an *arg-token* in object relation then the latter is identified as the Theme argument, e.g. "SQ 22536 suppressed **gp41**". In the Stanford Dependency Scheme, the object relation (*obj*) subsumes direct object (*dobj*) and indirect object (*iobj*) relations. Note that the object of preposition relation (*pobj*) is not used in the collapsed dependencies version of the scheme.

- If a Regulation event class trigger and a protein name are found in the same token, then the protein name is identified as the Cause argument, e.g. "**gp41**-induced".

A pre-processing step taken was to propagate modifier and prepositional relations over tokens that were co-ordinated or in an appositive relation. This was necessary since the Stanford dependency output provided by the organizers is in the collapsed format, which treats co-ordinated tokens asymmetrically, without propagating their dependencies.[9] At this stage we process the *transparency* markers identifying modifiers using the rule above. For multi-token triggers due to transparency, we consider the token containing the modified trigger for path extraction. For example, if "positive regulation" is the multi-token trigger in question we consider "regulation" as the root of the paths to protein names.

3.3.3 Event construction

At the event post-processing stage, we form complete events considering the trigger-argument pairs produced at the argument extraction stage whose arguments are resolved either to a protein name or to an event trigger. The latter are considered only for Regulation event class triggers.

For each Simple or Binding trigger-argument pair, we generate a single event with the argument marked as Theme. Given that we are dealing only with the core event extraction task, this approach is expected to deal adequately with all event types except for Binding, which can have multiple themes.[10] Regulation class events are formed in the following way. Given that the cause argument is optional, we generate Regulation events for trigger-argument pairs whose argument is a protein name or a trigger that has an already formed event. Since Regulation events can have other Regulation events as Themes, we repeat this process until no more events can be formed. Finally, for triggers that consistently generate two events, we generate at this stage the required Regulation class event whose Theme argument is the Simple class event triggered by the same textual string.

3.4 Rule-based system results

In this section we present results on the BioNLP 2009 event extraction shared task using the system described in Section 3.3. Overall, we expected that this approach would achieve high precision but relatively low recall. The results of the system are reported using the approximate span matching/approximate recursive matching variant of the evaluation in order to be able to compare with the results reported by the participants of the shared task.

[9] In August 2009, the shared task organizers re-generated the dependencies distributed in the propagation format. We avoid using them in order to be able to compare against other shared task participants who did not have access to the updated resources.

[10] Binding events with multiple Themes are evaluated as multiple Binding events with a single Theme only in the Event decomposition evaluation variants.

We first compared the performances obtained using the output of the different parsers provided by the organizers. For this purpose we used the development data, and the complete results using CCG, Bikel and McClosky parsers are presented in Tables 3.3, 3.4 and 3.5 respectively. In these tables, the numbers in parentheses (matches) are the events identified correctly by the system. The best performance was achieved using McClosky (39.66%), followed by CCG (38.73%) and Bikel (36.97%). The performances for the various event types and classes follows the same trends, with the exception of Binding events, in which McClosky has worse performance than the other two. As expected, the overall performance correlates roughly with the adaptation cost involved in the development of these parsers as described in Section 3.2.3. Bikel, which is essentially unadapted, has the worst performance overall, but it has been the cheapest to develop.

While this can be viewed as a task-based parser comparison, similar to the experiments of Miyao et al. (2008), one should be careful with the interpretation of the results. As pointed out by Miyao et al., this type of evaluation cannot be a substitute for a parsing evaluation against an appropriately annotated corpus (e.g. the experiments of Clegg and Shepherd (2007)), since in the context of a task only some aspects of parsing are likely to be relevant. Furthermore, as explained in Section 3.3.2, in our experiments we are are not using the native output of the parsers but its conversion to the Stanford Dependency (SD) format. Therefore, unavoidably we evaluate the conversion as well as the parsing itself. For this reason we avoided using the output of GDep which is not in the SD format.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (211)	259 (210)	59.27	81.08	68.48
Transcription	82 (33)	70 (33)	40.24	47.14	43.42
Protein_catabolism	21 (13)	15 (13)	61.90	86.67	72.22
Phosphorylation	47 (29)	37 (29)	61.70	78.38	69.05
Localization	53 (26)	27 (26)	49.06	96.30	65.00
SIMPLE	559 (312)	408 (311)	55.81	76.23	64.44
BINDING	248 (55)	184 (55)	22.18	29.89	25.46
Regulation	169 (33)	103 (33)	19.53	32.04	24.26
Positive_regulation	617 (137)	384 (137)	22.20	35.68	27.37
Negative_regulation	196 (42)	120 (42)	21.43	35.00	26.58
REGULATION	982 (212)	607 (212)	21.59	34.93	26.68
TOTAL	1789 (579)	1199 (578)	32.36	48.21	38.73

Table 3.3: Performance of the rule-based system using the CCG parser on the development data

We performed error analysis on the results obtained using the McClosky parser.[11] First

[11]For the error analysis we used the online evaluation for the development data provided by the organizers. At the time of writing this service seems to be withdrawn.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (202)	246 (201)	56.74	81.71	66.97
Transcription	82 (28)	58 (28)	34.15	48.28	40.00
Protein_catabolism	21 (16)	17 (16)	76.19	94.12	84.21
Phosphorylation	47 (23)	31 (23)	48.94	74.19	58.97
Localization	53 (22)	26 (22)	41.51	84.62	55.70
SIMPLE	559 (291)	378 (290)	52.06	76.72	62.03
BINDING	248 (42)	154 (42)	16.94	27.27	20.90
Regulation	169 (28)	91 (28)	16.57	30.77	21.54
Positive_regulation	617 (120)	318 (120)	19.45	37.74	25.67
Negative_regulation	196 (44)	108 (44)	22.45	40.74	28.95
REGULATION	982 (192)	517 (192)	19.55	37.14	25.62
TOTAL	1789 (525)	1049 (524)	29.35	49.95	36.97

Table 3.4: Performance of the rule-based system using the Bikel parser on the development data.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (228)	272 (227)	64.04	83.46	72.47
Transcription	82 (35)	72 (35)	42.68	48.61	45.45
Protein_catabolism	21 (15)	17 (15)	71.43	88.24	78.95
Phosphorylation	47 (29)	38 (29)	61.70	76.32	68.24
Localization	53 (27)	31 (27)	50.94	87.10	64.29
SIMPLE	559 (334)	430 (333)	59.75	77.44	67.45
BINDING	248 (38)	153 (38)	15.32	24.84	18.95
Regulation	169 (34)	104 (34)	20.12	32.69	24.91
Positive_regulation	617 (133)	355 (133)	21.56	37.46	27.37
Negative_regulation	196 (44)	107 (44)	22.45	41.12	29.04
REGULATION	982 (211)	566 (211)	21.49	37.28	27.26
TOTAL	1789 (583)	1149 (582)	32.59	50.65	39.66

Table 3.5: Performance of the rule-based system using the McClosky parser on the development data.

we wanted to assess the trigger extraction. Examining the lists of False Positive and False Negative events we observed that the most common triggers of events not extracted correctly had lemmas that were included in the dictionary constructed from the training data, such as “binding”, “expression”, “induction”, “activation”. This suggests that most event extraction errors are due to argument identification. Disabling the processing of the transparency markers on the triggers the performance on the development data drops

to 39.28%, the main reason being the decreased recall in Binding events. We conclude that using a dictionary for trigger extraction can provide sufficient support to the event extraction task, despite the rather strong assumptions it is based upon.

Concerning the performances on individual event types and classes, Simple events are easier than the other event classes due to their simpler structure, with 59.75% recall, 77.44% precision and 67.45% F-score on average. In particular, Phosphorylation and Protein_catabolism event types are extracted with good precision and recall using only the rules described in Section 3.3.2 and few annotated lemmas, namely "phosphorylate" and "phosphorylation", and "degrade", "degradation" and "proteolysis" respectively.

Performance on Binding events was much lower (18.95% F-score), but this is partly due to the fact that the system does not construct events with multiple Themes. Using the Event decomposition evaluation variant which relaxes this requirement the Binding performance becomes 33.97%/69.28%/45.59% (R/P/F). Performance on Regulation events was rather low, 21.49%, 37.28% and 27.26% in Recall, Precision and F-score respectively. We hypothesize that this can be attributed to three factors. First, Regulation events have an optional Cause argument which must be identified correctly, unless we use the Event decomposition evaluation variant. Second, they can have other events as their arguments as well as protein names, therefore argument identification becomes harder. Finally, if the argument of a Regulation event is another event, then the latter must be identified correctly as well, therefore errors in these bottom-level events propagate to events that have them as arguments. The variety of lexical triggers is unlikely to be the cause of the drop in performance though, since our system performs reasonably well in the Gene_Expression and Localization classes which exhibit similar lexical variation. Rather it is due to the combination of the lexical variation with the requirement to make the distinction between the theme and optional cause argument, which cannot be handled appropriately by the small set of unlexicalized rules employed.

Based on the comparison performed on the development data, we ran our system using the McClosky parser on the test data. The results appear in Table 3.6 and the overall performance achieved (35.39%) is relatively close to the one obtained on the development set (4% lower). This is important since rule-based systems are prone to overfitting their development data due to the way they are built. Also, it is worth pointing out that trigger transparency processing has greater effect on the test data, without it the performance drops to 34.35%.

Compared to the performances achieved by the shared task participants, the system presented would be ranked 7th in overall performance, 10th in the Simple events, 11th in Binding events and 5th in Regulation events, out of 24 participating systems. Given that no annotated training data were needed[12], we believe this is a strong result, since it

[12]We used the training data to construct the trigger dictionary but, as argued in Section 3.3.1, this can be obtained from domain experts.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	722 (336)	428 (336)	46.54	78.50	58.43
Transcription	137 (36)	126 (36)	26.28	28.57	27.38
Protein_catabolism	14 (4)	4 (4)	28.57	100.00	44.44
Phosphorylation	135 (88)	107 (88)	65.19	82.24	72.73
Localization	174 (56)	63 (56)	32.18	88.89	47.26
SIMPLE	1182 (520)	728 (520)	43.99	71.43	54.45
BINDING	347 (71)	186 (71)	20.46	38.17	26.64
Regulation	291 (46)	196 (46)	15.81	23.47	18.89
Positive_regulation	983 (208)	630 (208)	21.16	33.02	25.79
Negative_regulation	379 (65)	221 (65)	17.15	29.41	21.67
REGULATION	1653 (319)	1047 (319)	19.30	30.47	23.63
TOTAL	3182 (910)	1961 (910)	28.60	46.40	35.39

Table 3.6: Performance of the rule-based system using the McClosky parser on the test data.

surpasses some of the systems that used supervised machine learning methods. Restricting the comparison to rule-based systems, it would have the 2nd best performance out of 9 such systems, most of which used external knowledge sources in order to improve their performance. Our system is based on the datasets and the syntactic parses provided by the organizers only. The best rule-based system (Kilicoglu and Bergler, 2009) had overall performance of 44.62% in F-score, ranking 3rd overall. The main difference is that it used a much larger set of dependency rules (27) which were extracted using the training data. We only used the development data for this purpose. Furthermore, Kilicoglu and Bergler (2009) lexicalized the rules they developed and employed heuristics in order to correct syntactic parsing errors based on work by Schuman and Bergler (2006). While the benefits from these additional processing steps are indisputable, they involved a lot of manual work, both for rule construction as well as for the annotation of the data used to extract the rules. We argue that these performance benefits could be obtained using machine learning methods aimed at ameliorating the main weakness of the current system, the argument identification stage. In Section 3.5, we present a method to achieve this using machine learning methods with partial event annotation.

Compared to the rule-based approach of Vlachos et al. (2009) which is very similar (dictionary-based trigger extraction, similar unlexicalized rules[13]), the performance is improved substantially. The main difference between that system and the one presented here is that the former uses the domain-independent RASP parser. While its performance is still reasonable (it was ranked 10th overall, 21.12/56.90/30.80 (Recall/Precision/F-score)), these results lag behind those reported here. Note that a direct comparison using the out-

[13]The only lexicalization is the prepositions used in the equivalent of the second rule in Section 3.3.2.

put of RASP is not possible since the latter uses its own syntactic dependency scheme as described in Chapter 2 and there is no lossless conversion to the Stanford dependency scheme.

Overall, the results of this section demonstrate that the use of domain-adapted parsing is beneficial to event extraction. This is not very surprising since the system presented depends heavily on the parsing output and it is partly due to the domain-adapted PoS tagger used by McClosky and CCG. What we consider more interesting to observe is that a combination of a simple rule-based system that does not require training data with an unadapted syntactic parser has performance comparable with systems that require substantial amounts of training data. Nevertheless, the use of a domain-adapted syntactic parser improves the performance. We argue that the annotation cost of this adaptation is a good investment because, unlike the task-specific training data required by all the better-performing systems[14], improved syntactic parsing is likely to be useful for different event extraction tasks, or even other IE tasks, e.g. anaphora resolution. On the contrary, as demonstrated in the experiments of Chapter 2, even small changes in the task definition can minimize the utility of task-specific annotated data drastically. Therefore, we suggest that domain-adaptation of the syntactic parsing should be considered before creating task-specific training data, at least in tasks that are heavily dependent on syntactic parsing, such as event extraction.

3.5 Improving argument identification with partial annotation and support vector machines

In this section, we present an approach to argument identification which attempts to overcome the drawbacks of the rule-based approach presented in Section 3.3.2. The results of Section 3.4 demonstrated that simple rules, while simple to construct, are not sufficient for this task. Therefore we resort to machine learning in order to learn complex rules involving lexicalized syntactic dependencies effectively. The annotation used for training is partial, since only associations between triggers and their arguments are needed. The technique used in Chapter 2, namely to automatically annotate training data is unlikely to be applicable, since there are no resources that could be used for this purpose. In the following subsections, we provide an overview of the machine learning method used (Section 3.5.1) and describe in detail how it was employed in the existing system architecture (Section 3.5.2).

[14]The only exception is the system of Kilicoglu and Bergler (2009) which, while it is not trainable, in contrast to our approach used the training data for development.

3.5.1 Support vector machines

Support vector machines (SVMs) (Vapnik, 1995) are a widely-used state-of-the-art statistical learning model. They have been used successfully in a variety of tasks, including text classification (Joachims, 1998b), handwritten digit recognition (LeCun et al., 1995) and relation extraction (Zelenko et al., 2003; Culotta and Sorensen, 2004). During training, a dataset D comprising of two classes $\{-1, +1\}$ is projected to a (possibly) higher dimensional space and a maximum margin separating hyperplane is found between the two classes. The separating hyperplane is defined by a set of datapoints $\{x_1, ..., x_n\}$ and their labels $\{y_1, ..., y_n\}$ which are the support vectors. Each of these datapoints is assigned a weight $a_1, ...a_n$. The projection to the higher dimensional space is performed using a suitable kernel function $K(x_i, x_j)$, which allows the calculations to take place in the original dimensional space of the instances. Intuitively, a kernel function encodes the similarity between two instances. During classification, the test datapoints are classified according to the side of the separating hyperplane on which they are found to lie. For a datapoint x, this is performed using the following function:

$$f(x, a) = sign(\sum_{i=1}^{n} y_i a_i K(x, x_i) + b) \tag{3.1}$$

The sign of the weighted sum of the inner products of the datapoint with the support vectors denotes the class. Its absolute value is the distance of the datapoint from the separating boundary, which is also referred to as the margin. This should not be confused with probability estimates that can be obtained from other statistical learning models. It ranges from 0 to ∞ and most importantly, the margins yielded by different SVM models are not comparable with each other because different datasets and/or kernel functions define different spaces. The decision values obtained from SVMs can be converted to probabilities by fitting a sigmoid function (Platt, 1999).

The choice of the kernel function defines the space in which the data is projected and it is very important because it affects the separating hyperplane to be discovered. For example, the simple and widely used linear kernel function, $K(x_i, x_j) = x_i \cdot x_j$, can only define linear separating hyperplanes. However, non-linear kernel functions such as the Gaussian kernel ($K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$) can discover more complex hyperplanes which can take advantage of non-linear combinations between features. This can result in better classification performance (Burges, 1998), which has been verified by various authors (Joachims, 1998b; Keerthi and Lin, 2003). However, they are slower to train and parameter optimization is required (normally performed through cross-validation on the training data) in order to obtain good performance. On the contrary, linear kernel SVMs can be implemented to run much faster and the parameters can be set quite effectively without cross-validation, as in SVM-Light (Joachims, 1998a).

Support vector machines in their standard formulation are binary classifiers. Their success

motivated researchers to investigate extensions that would allow multiclass classification with this model. Several methods have been presented for this purpose. The most popular of them decompose the multiclass task to several binary classification ones for training and combine the output of the binary classifiers during testing. One strategy for achieving this is the one-against-all scheme, in which for each class a separate classifier is trained against the rest of the data. During testing, the class whose classifier has the largest positive margin is selected (Vapnik, 1995). Another strategy is the one-against-one scheme in which binary classifiers are trained for each pair of classes and during testing voting among the classifiers takes place to decide on the class (Hsu and Lin, 2002). Other strategies involve error-correcting codes (ECOC) in order to reduce the multiclass task to binary ones (Rennie and Rifkin, 2001). It has been observed though that the performance of the combined multiclass classifier is more dependent on the performance of the binary ones, rather than the strategy used to combine them (Rennie and Rifkin, 2001).

3.5.2 Partial annotation for argument identification

In this section, we describe how we cast argument identification for event extraction as a classification task. Following the trigger extraction stage (described in Section 3.3.1), for each trigger combined with each of its valid arguments we create a classification instance, i.e. protein names for Simple and Binding event triggers, and triggers and protein names for Regulation event triggers. This results in a superset of the trigger-argument pairs used for the rule-based approach in Section 3.3.2. The classification task is to assign to an instance, i.e. a trigger-argument pair, the correct argument type. Therefore, we construct a binary classifier which determines whether a protein name is the Theme argument of a Simple or Binding trigger (*ThemePositive* or *ThemeNegative*) and a ternary classifier which determines whether a protein name or another trigger (and as consequence its associated events) is the Theme or the Cause argument of a Regulation trigger (*RegThemePositive*, *RegCausePositive*, *RegNegative*).

In order to acquire labeled instances for training, we process the gold standard (GS) annotation of the training data in the following way. First we decompose the GS events into multiple events with single arguments, as in the event decomposition evaluation variant (Section 3.2.2). In cases of events being arguments to Regulation events, the former are replaced by their triggers. We match the triggers extracted in Section 3.3.1 with those included in the gold standard, ignoring the event type annotation. Since we identify primarily single-token triggers, we replicate the approximate span matching used in evaluation in order to achieve better coverage. Then, we label each trigger-argument pair extracted using the decomposed GS events. If the instance being considered has a Simple or a Binding trigger, and if the pair is included in the GS then it is labeled as *ThemePositive*, else it labeled as *ThemeNegative*. If the instance being considered has a Regulation trigger that has been matched with a GS trigger, and if its argument is

a protein name and their pair is included in the GS then it is labeled according to the latter (*RegThemePositive* or *RegCausePositive*), else, if not found in the GS it is labeled as *RegThemeNegative*. The same process is followed if the argument is an event trigger which has been matched with a GS trigger. The process is presented in Algorithm 1.

Algorithm 1 Acquisition of labeled instances for argument identification from the gold standard (GS).

```
1:  Input: triggers T, trigger-argument pairs TA, GS annotation
2:  Output: labeled ta instances
3:  Identify the trigger T_gold ⊆ T matching GS
4:  Decompose GS into single-argument events, TA_gold
5:  for all ta in TA do
6:    if t is Simple/Binding then
7:      if ta in TA_gold then
8:        ta is labeled ThemePositive
9:      else
10:       ta is labeled ThemeNegative
11:   else if t in T_gold then
12:     if a is protein name then
13:       if ta in TA_gold then
14:         ta is labeled RegThemePositive/RegCausePositive
15:       else
16:         ta is labeled ThemeNegative
17:     else if a in T_gold then
18:       if ta in TA_gold then
19:         ta is labeled RegThemePositive/RegCausePositive
20:       else
21:         ta is labeled RegNegative
22:     else
23:       Ignore ta
24:   else
25:     Ignore ta
```

The reason we consider only Regulation triggers that are matched in the gold standard, therefore considering only those that have at least a Theme argument annotated (line 11 in Algorithm 1), is in order to avoid valid *RegCausePositive* instances being labeled as *RegNegative*. Recall that the Cause argument is optional, while Theme is obligatory for Regulation events. This means that if an appropriate Theme argument is not present, then it is possible that a Cause argument that is present is not annotated. Similarly, when considering event triggers as arguments, we acquire labels only for instances involving triggers that were annotated in the GS (line 17 in Algorithm 1). Since triggers without

an appropriate Theme are not annotated in the gold standard, it is possible that a valid *RegThemePositive* or *RegCausePositive* is labeled as *RegNegative* instance not because of the actual relation between the trigger and the argument but because the argument did not have an appropriate Theme present.

While we obtained these labeled examples from the annotated data provided by the organizers, in practice, such information could be easily obtained by domain experts, since it is in the form of binary or ternary decisions instead of complete trigger and event annotation. Re-visiting an example from Section 3.2.1:

However, the HTLV-I genes including **Tax** are not expressed significantly in primary leukemic cells from ATL patients.

The annotation for the event of the above example becomes:

```
T1 Protein Tax
T2 Gene_expression expressed
T2-T1 ThemePositive
```

Unlike in the gold standard annotation, all trigger lines are generated using the dictionary as described in Section 3.3.1. Furthermore, event types are also determined as part of this process, since only associations between triggers and arguments are annotated manually.

This process is certain to introduce some noise, since it relies on a dictionary for trigger extraction and for determining event type. Some triggers might be omitted due to limited dictionary coverage, thus resulting in partial annotation. However, this is unlikely to have a detrimental effect in the current conditions, since the results of Section 3.4 suggest that the trigger extraction is adequate. For the same reason, determining event type is unlikely to be an issue. It is important to note that even if the event type determined by the dictionary is incorrect, this is unlikely to affect the argument identification annotation, since the latter is dependent on the lemma of the trigger rather than its type. For example, the Theme argument of the trigger "expression" is unlikely to depend on whether the event denoted is Gene_expression or Transcription.

Trigger over-generation can also be a concern since it can give rise to more instances needing labeling, especially when many Regulation triggers exist in the same sentence. However, this is not so likely to happen given the restricted domain of the task, since if a trigger word and appropriate argument are present in the same sentence, they are likely to be associated. Furthermore, if they are not then this is also likely to be useful information to learn from. In a more complicated example from Section 3.2.1:

... SQ 22536 suppressed **gp41**-induced **IL-10** production in monocytes.

The annotation for the events of the above example becomes:

```
T1 Protein gp41
T2 Protein IL-10
T3 Negative_regulation suppressed
T4 Positive_regulation induced
T5 Gene_expression production
T5-T1 ThemeNegative
T5-T2 ThemePositive
T4-T1 RegCausePositive
T4-T2 RegNegative
T4-T3 RegNegative
T4-T5 RegThemePositive
T3-T1 RegNegative
T3-T2 RegNegative
T3-T4 RegThemePositive
T3-T5 RegNegative
```

Note that if "IL-10" was replaced by a string not containing a protein name (e.g. a definite description), this would result in omitting only the annotations concerning its relations with the triggers of the sentence. On the contrary, as explained in Section 3.2.1, in the annotation scheme used in the shared task it would result in no events being annotated, since the absence of a protein name which is the Theme of the bottom-level event would not allow any events to be annotated. We argue that a human annotator would need to make these decisions anyway, however this partial (with respect to the task definition) annotation scheme allows the encoding of this information in a more flexible way. Also this is likely to be a more effective way to use the annotation time, since annotators would be requested to annotate pre-determined trigger-argument pairs instead of searching for events from scratch, given only the protein name annotation.

Overall, the suggested partial annotation scheme results in data more suitable to training the components of the event extraction system presented. We wish to point out that this type of partial annotation, while it can provide insights to the event annotation scheme, is not meant to be a replacement for it. Fully annotated data is required for evaluation which is essential for system development. In the context of the BioNLP 2009 event extraction shared task, this would suggest changing the training dataset provided, but not the development and test datasets.

For each trigger-argument pair we extract the shortest dependency path connecting them using Dijkstra's algorithm (Cormen et al., 1990). We allow paths to follow the opposite dependency direction by incorporating the direction in the dependency labels. Apart from the dependency path, we extract as features the trigger-token, the trigger event type and the argument type (event type if the argument is a trigger or *Entity* in case of protein names). We discard trigger-argument pairs that do not have a dependency path

connecting them and filter the training set further, considering only instances in which the trigger was at a maximum distance of 4 dependencies away from the argument, since longer paths were too sparse to be useful in classifying unseen instances. This process resulted in 9,699 binary and 10,541 ternary decisions compared to 6,607 triggers and 9,597 events annotated in the training data provided.[15] However, it must be pointed out that the latter annotation scheme for each item annotated has multiple items annotated as negative implicitly, in other words, non-events are not annotated. If we consider only the positive instances, then the annotation scheme suggested results in 3,517 *ThemePositive* and 3,933 *RegTheme/CausePositive* instances, which are simpler since they do not need textual span and event type specification.

At classification time, we consider as *Theme/RegNegative* any instances in which the dependency path has not been encountered in the training data, as well as instances without a dependency path connecting trigger and argument. This is necessary in order to avoid instances being classified only on the basis of the trigger-token and the argument type. After the classifier has assigned labels to the trigger-argument pairs, we construct events as described in Section 3.3.3. The process sometimes runs into cyclic dependencies between Regulation event triggers, which are possible in cases where it is unclear (to the classifier) which is the trigger and which is the argument in a given pair of triggers. Similar problems occur involving 3 or more such triggers. We resolve such problems using the confidence of the classifier for each decision by removing the least confident *RegThemePositive* or *RegCausePositive* assignment from the cycle.

3.6 SVM-based system results

In this section we present results on the BioNLP 2009 event extraction shared task using the system described in Section 3.5. As in Section 3.4, the results are reported using the approximate span matching/approximate recursive matching variant of the evaluation in order to be able to compare with the results reported by the participants of the shared task. In our experiments we used the LIBSVM toolkit (Chang and Lin, 2001) which provides an implementation of Support Vector Machines with various kernels and uses the one-against-one scheme for multiclass problems. In all experiments, the Gaussian kernel was used in order to capture potential non-linear feature combinations, e.g. cases where the combination of dependency path and trigger-token would result in a different decision rather than each of them independently. The parameters were optimized using cross-validation on the training data.

We focused on using the output of the two domain-adapted parsers, namely CCG and

[15]The actual numbers depend on the parser used. We counted here the instances created combining McClosky and CCG, as done in the experiments of Section 3.6. For the gold standard, we considered the decomposed event definition.

McClosky. The reason for this is that, as argued in Section 3.4, given the importance of syntactic parsing to event extraction one should consider domain adaptation of syntactic parsing before developing task-specific training resources. We first compared the performances obtained using the output of the different parsers provided by the organizers. For this purpose we used the development data, and the complete results using CCG and McClosky parsers are presented in Tables 3.7 and 3.8 respectively.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (239)	289 (238)	67.13	82.35	73.97
Transcription	82 (39)	58 (39)	47.56	67.24	55.71
Protein_catabolism	21 (14)	15 (14)	66.67	93.33	77.78
Phosphorylation	47 (24)	27 (24)	51.06	88.89	64.86
Localization	53 (36)	36 (36)	67.92	100.00	80.90
SIMPLE	559 (352)	425 (351)	62.97	82.59	71.46
BINDING	248 (53)	148 (53)	21.37	35.81	26.77
Regulation	169 (42)	112 (42)	24.85	37.50	29.89
Positive_regulation	617 (217)	470 (217)	35.17	46.17	39.93
Negative_regulation	196 (51)	130 (51)	26.02	39.23	31.29
REGULATION	982 (310)	712 (310)	31.57	43.54	36.60
TOTAL	1789 (715)	1285 (714)	39.97	55.56	46.49

Table 3.7: Performance of the SVM-based system using the CCG parser on the development data.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (236)	271 (235)	66.29	86.72	75.14
Transcription	82 (44)	74 (44)	53.66	59.46	56.41
Protein_catabolism	21 (14)	18 (14)	66.67	77.78	71.79
Phosphorylation	47 (34)	44 (34)	72.34	77.27	74.73
Localization	53 (37)	41 (37)	69.81	90.24	78.72
SIMPLE	559 (365)	448 (364)	65.30	81.25	72.40
BINDING	248 (66)	188 (66)	26.61	35.11	30.28
Regulation	169 (49)	114 (49)	28.99	42.98	34.63
Positive_regulation	617 (232)	563 (232)	37.60	41.21	39.32
Negative_regulation	196 (59)	149 (59)	30.10	39.60	34.20
REGULATION	982 (340)	826 (340)	34.62	41.16	37.61
TOTAL	1789 (771)	1462 (770)	43.10	52.67	47.40

Table 3.8: Performance of the SVM-based system using the McClosky parser on the development data.

The main observation is that, using either parser, the results are much improved compared to those reported in Section 3.4, by approximately 8% in F-score in either case. Most of the improvement is due to higher recall, suggesting that the argument identification component is able to learn patterns that are relevant to the task. Overall, using the output of CCG results in higher precision, while McClosky results in higher recall. In an effort to take advantage of both parsers simultaneously, we combined them by adding for each trigger-argument pair the dependency paths extracted by both parsers. This improved the results further, as can be observed in Table 3.9.

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	356 (250)	292 (249)	70.22	85.27	77.02
Transcription	82 (42)	64 (42)	51.22	65.62	57.53
Protein_catabolism	21 (15)	16 (15)	71.43	93.75	81.08
Phosphorylation	47 (31)	35 (31)	65.96	88.57	75.61
Localization	53 (39)	40 (39)	73.58	97.50	83.87
SIMPLE	559 (377)	447 (376)	67.44	84.12	74.86
BINDING	248 (66)	181 (66)	26.61	36.46	30.77
Regulation	169 (50)	119 (50)	29.59	42.02	34.72
Positive_regulation	617 (244)	533 (244)	39.55	45.78	42.43
Negative_regulation	196 (54)	135 (54)	27.55	40.00	32.63
REGULATION	982 (348)	787 (348)	35.44	44.22	39.34
TOTAL	1789 (791)	1415 (790)	44.21	55.83	49.35

Table 3.9: Performance of the SVM-based system using the CCG and the McClosky parsers on the development data.

We then run the system combining the two parsers on the test data, obtaining the results presented in Table 3.10. Overall, the system presented would have had the 2nd best performance in the shared task achieving 41.42%/56.76%/47.89% in Recall/Precision/F-score. In the individual classes, the rankings were 2nd for Regulation events, 3rd for Simple event and 5th for Bindings. For a more detailed comparison, the top system presented by the University of Turku (Bjorne et al., 2009) achieved 46.73%/58.48%/51.95% (R/P/F) while the following one by the JULIE laboratory (Buyko et al., 2009) achieved 45.82%/47.52%/46.66%. It is important to consider that both these systems (like many of the ones below them) needed substantially more data for their development.

More specifically, the JULIE lab system makes use of many external knowledge sources. In order to improve the dictionary used for trigger extraction, all of the GENIA event corpus (1,999 abstracts) (Kim et al., 2008) combined with expert manual curation was used, compared to using only the training data provided by the organizers (a subset of GENIA consisting of 800 abstracts). Furthermore, for argument identification, features

Event Type/Class	gold (match)	answer (match)	recall	precision	fscore
Gene_expression	722 (445)	541 (445)	61.63	82.26	70.47
Transcription	137 (41)	66 (41)	29.93	62.12	40.39
Protein_catabolism	14 (6)	7 (6)	42.86	85.71	57.14
Phosphorylation	135 (106)	116 (106)	78.52	91.38	84.46
Localization	174 (71)	74 (71)	40.80	95.95	57.26
SIMPLE	1182 (669)	804 (669)	56.60	83.21	67.37
BINDING	347 (101)	223 (101)	29.11	45.29	35.44
Regulation	291 (69)	176 (69)	23.71	39.20	29.55
Positive_regulation	983 (364)	834 (364)	37.03	43.65	40.07
Negative_regulation	379 (115)	285 (115)	30.34	40.35	34.64
REGULATION	1653 (548)	1295 (548)	33.15	42.32	37.18
TOTAL	3182 (1318)	2322 (1318)	41.42	56.76	47.89

Table 3.10: Performance of the SVM-based system using the CCG and the McClosky parsers on the test data.

were derived from the Gene Ontology Annotation database[16], the Universal Protein Resource[17], the Medical Subject Headings thesaurus[18] and a PoS tagger and chunker trained on the GENIA corpus (Buyko et al., 2006). The authors report that the combination of the feature-based classifier (which incorporates most of these external resources) with the dependency graph kernel classifier improved their performance by up to 6% in F-score on the development set. While the use of these resources and their successful integration in an event extraction system is commendable, we believe it is important that the system presented here can achieve comparable performance using fewer resources.

The system by the University of Turku followed a machine learning approach to trigger extraction, which is unlike the dictionary-based approach followed by the other top-performing systems in the shared task, including the one presented here. While this is likely to be partially responsible for the performance difference observed when compared to the other participating systems (4% and 5.2% better than ours and the JULIE lab respectively), it obviously requires explicit trigger annotation, thus being more expensive. Furthermore, we argue that the data provided by the organizers are not suitable to train a trigger extractor, since only triggers participating in events are annotated, thus ignoring (semantically) valid triggers, as explained in Section 3.5.2. We hypothesize that this is the reason the authors had to adjust the decisions of their SVM classifiers using an external parameter, instead of using the decisions returned by the classifier directly (Bjorne et al., 2009).

[16]http://www.ebi.ac.uk/GOA/
[17]http://www.uniprot.org/
[18]http://www.nlm.nih.gov/mesh/

It is important to note that we explicitly avoided using the gold standard trigger information for training purposes, including the training of the argument identification component (Section 3.5.2).[19] The majority of the systems participating in the shared task used the gold standard associations between triggers and their arguments, thus implicitly using the gold standard trigger information. While this is likely to improve the performance, it means that the full annotation of the training data was used, unlike the partial information used in our approach.

3.7 Discussion

In this work we focus on the obligatory core event extraction task, Task 1. The optional Task 2, which involves the extraction of secondary arguments can also be tackled by building a classifier that would learn whether to associate an Entity with an Event or one of its Themes or Causes, similar to the approach used for Themes of Simple events. Since the Entities are not provided, they could be recognized using the techniques described in Chapter 2. However, the lack of Entity annotation deterred us from pursuing this option, since we would not be able to evaluate in an informative way. Also performances of the participants on Task 2 were observed to be highly dependent on the performances on Task 1 (Kim et al., 2009).

As mentioned in Section 3.2.1, Task 3 did not have text-bound annotation. Kilicoglu and Bergler (2009) showed that it is possible to identify event speculation and negation using a combination of a dictionary and rules, similar to the one described in Section 3.3.2, achieving the top performance in shared task (25% F-score). However, as pointed out by the organizers, the lack of text-bound annotation is most likely the cause of the low performances (the other participants achieved 12% F-score or less), despite the (relative) simplicity of the task. We argue that the lack of text-bound annotation not only renders system development more difficult, but it is likely to cause annotation inconsistencies as well, therefore the evaluation becoming more problematic. For these reasons, we avoided attempting this task.

Finally, echoing the observations of Buyko et al. (2009), we found that annotation inconsistency even for the core event extraction task was affecting our results significantly. In many cases the event triggers annotated in the development data were rather misleading, e.g. “negative” as a Gene_expression event trigger (abstract 8622883), “increase the stability” as a Positive_regulation event trigger (abstract 8626752), “disappearance” as a Binding event trigger (abstract 10455128). Furthermore, there were cases of events ignored by the annotation, such as “regulation of thymidine kinase” (abstract 8622883).

[19] We did use the annotated triggers to construct the annotated dictionary, but we argue that this could have been provided by domain experts, without requiring annotating abstracts with events.

An additional complication is that events that are annotated due to anaphoric linking can have disproportionate effect on the scores. In an example from abstract 9794375:

> CD3, **CD2**, and **CD28** are functionally distinct receptors on T lymphocytes. Engagement of any of these receptors induces the rapid tyrosine phosphorylation of a shared group of intracellular signaling proteins, including **Vav**, **Cbl**, p85 phosphoinositide 3-kinase, and the **Src** family kinases **Lck** and **Fyn**.

With respect to Task 1, the gold standard annotation for this is:

```
T4 Protein CD2
T5 Protein CD28
T6 Protein Vav
T7 Protein Cbl
T8 Protein Src
T9 Protein Lck
T10 Protein Fyn
T36 Binding Engagement
T37 Positive_regulation induces
T39 Phosphorylation phosphorylation
E3 Binding:T36 Theme:T4
E4 Binding:T36 Theme:T5
E5 Positive_regulation:T37 Theme:E15 Cause:E3
E6 Positive_regulation:T37 Theme:E14 Cause:E3
E7 Positive_regulation:T37 Theme:E13 Cause:E4
E8 Positive_regulation:T37 Theme:E16 Cause:E3
E9 Positive_regulation:T37 Theme:E16 Cause:E4
E10 Positive_regulation:T37 Theme:E15 Cause:E4
E11 Positive_regulation:T37 Theme:E13 Cause:E3
E12 Positive_regulation:T37 Theme:E14 Cause:E4
E13 Phosphorylation:T39 Theme:T10
E14 Phosphorylation:T39 Theme:T6
E15 Phosphorylation:T39 Theme:T7
E16 Phosphorylation:T39 Theme:T9
```

By failing to recognize the anaphoric Binding events `E3` and `E4`, an otherwise perfect system is going to receive (according to the approximate span matching/approximate recursive matching variant) 2 false negatives for the Binding events, 8 false negatives for the missing Positive_regulation events due to the missing Causes and 4 false positives for the incomplete Positive_regulation events extracted.

Despite this criticism, we believe that the BioNLP 2009 shared task on event extraction was a big step forward for biomedical information extraction and we are grateful to the organizers for the effort and resources provided, without which the research presented here would not have been possible. However, we argue that future work should look at improving the annotation in order to be able to assess the progress in the systems developed.

Apart from these annotation and evaluation issues, there are interesting directions for future work on the system development side. While this chapter (in-line with the goals of the thesis) explores the use of partial annotation for event extraction, future work should investigate methods that can take advantage of unlabeled data. Recent work by Daumé III (2009) showed promising results on using unlabeled data for dependency parsing. Furthermore, joint inference models, such as Markov Logic Networks (Riedel, 2008), can capture interactions between related decisions, e.g. the Theme assignment is able to affect the choice of Cause for a given trigger. Such models were applied to the BioNLP 2009 event extraction shared task by Riedel et al. (2009) and were ranked 4th in the event extraction task using the gold standard annotation of the training data. Therefore, it would be of interest to explore the use of partial annotation with these models.

3.8 Summary

In this chapter we discussed approaches to tackle event extraction using partially annotated data, developing two systems for this purpose. The first one relies on a dictionary of lemmas associated with event types and a set of simple rules for argument identification operating on top of the dependency output of a syntactic parser (Section 3.3). The second system combines the same dictionary with SVM classifiers trained on associations between event triggers recognized by the dictionary and their candidate arguments (Section 3.5). Both systems were evaluated in the context of the BioNLP 2009 shared task and they achieved performances competitive with systems using larger amounts of annotation. Apart from the reasonable performances achieved, domain-adapted syntactic parsing was found to be beneficial, without requiring annotation tailored to the particular task being considered.

Chapter 4

Biomedical semantic verb clustering

4.1 Introduction

Verbs are used in natural language in order to express actions, events, or states of being and therefore are central to the meaning of sentences. They have been the subject of numerous studies in the NLP community that have resulted in several manually constructed taxonomies, e.g. Levin (1993) and VerbNet (Kipper-Schuler, 2005). These taxonomies group verbs in classes according to their meaning in general English text (e.g. *MOTION* class for "travel", "walk" and "run"). Such classes can provide important support for other NLP tasks, such as word sense disambiguation (Prescher et al., 2000) and semantic role labeling (Swier and Stevenson, 2004). While manually curated taxonomies can be useful, their construction is difficult and requires substantial amount of human effort.

In the biomedical domain, verbs common in general English acquire domain-specific semantics, while new verbs emerge to convey specialized activities or states.[1] For example, the verb "express" is used to denote a particular biochemical event, while the verb "dephosphorylate" is unlikely to be found in general English text. These cases help illustrate the challenges the biomedical domain presents to existing manually constructed taxonomies which, due to their construction, lack the coverage and are inadequate for specific domains (Korhonen et al., 2006). Even though biomedical verb clustering is not a typical biomedical IE task, we believe it is relevant to the goals of the latter. In particular, verb clustering can assist the construction of dictionaries which are employed by IE systems such as those described in Chapter 3.

A remedy proposed for the issue of coverage is the use of supervised classification methods to assign a new verb to one of the classes of the taxonomy. This approach has been explored by a number of authors with good results for general English (Li and Brew, 2008; Joanis et al., 2008) as well as the biomedical domain (Korhonen et al., 2008). Such approaches though, apart from the requirement for labeled data for training, cannot handle

[1] For a detailed analysis of the role of verbs in the biomedical domain see Cohen et al. (2008).

adequately verbs whose meaning is not already included in the taxonomy. For example, using training data obtained from a general English taxonomy of verbs, a supervised classification method will not identify a new class for a biomedical verb but it will try to assign it to one of the existing classes, which is unlikely to be appropriate.

A more promising direction of research is the use of unsupervised clustering, which does not require labeled training data. Such approaches have been explored in both general language (Brew and Schulte im Walde, 2002; Korhonen et al., 2003) and the biomedical domain (Korhonen et al., 2006). However, the clustering algorithms applied so far require the number of clusters as input. This is problematic as we cannot know in advance how many classes exist in the data. Even if the number of classes for a task was known (e.g. in the context of a carefully controlled experiment), a particular dataset may not contain instances for all the classes. Moreover, each class is not necessarily contained in one cluster exclusively, since the target classes are defined manually without taking into account the feature representation used.

In order to provide a solution to this issue, in this chapter we explore the application of Dirichlet Process Mixture Models (DPMMs) to the problem of semantic verb clustering. These models have the attractive property that the number of components used to model the data is not fixed in advance but is actually determined by the model and the data. This property is particularly interesting for NLP where many tasks are aimed at discovering novel, previously unknown information in corpora. Recent work has applied Bayesian non-parametric mixture models to language modeling (Teh, 2006), anaphora resolution (Haghighi and Klein, 2007) and syntactic parsing (Cohn et al., 2009) with promising results.

Bayesian non-parametric mixture models are not the only approach to determine the number of clusters for a particular dataset. Various methods have been proposed for other clustering frameworks such as k-means (Hamerly and Elkan, 2003) and spectral clustering (Sanguinetti et al., 2005). However, these methods typically involve multiple runs for different numbers of clusters in order to find the optimal one. Furthermore, they rely on assumptions that are difficult to verify whether they apply or not to a particular dataset, thus limiting their applicability. For example, the approach presented by Sanguinetti et al. (2005) relies on the assumption of tight, widely separated clusters.

On the other hand, the Bayesian framework allows for modeling the parameters and the number of clusters simultaneously. This is achieved by using a suitable prior over cluster assignments and in this work we employ the Dirichlet Process prior for this purpose. The choice of prior is independent of the way the data is modeled in the clusters and therefore it can easily be extended to different tasks. Finally, the non-parametric Bayesian framework makes no additional assumptions compared to its parametric counterpart with respect to the nature of the data.

We begin this chapter by describing Dirichlet Process Mixture Models (Section 4.2) and

the technique used to infer their parameters (Section 4.3). Then, we review various clustering evaluation measures and propose a modified version of the newly introduced V-measure (Rosenberg and Hirschberg, 2007) (Section 4.4). Following this, we describe the dataset and the experiments conducted (Section 4.5). Finally, we discuss related work as well as directions for future research (Section 4.6).

4.2 Unsupervised clustering with DPMMs

In a Bayesian mixture model we assume that the parameters for each component θ are generated from a prior distribution G, and in turn each instance x_i is generated by its chosen component θ_i:

$$\begin{aligned} \theta_i|G &\sim G \\ x_i|\theta_i &\sim F(x_i|\theta_i) \end{aligned} \tag{4.1}$$

With respect to the task verb clustering, for each cluster a set of parameters is generated (i.e. a description of the features describing the class) and for each verb a cluster is chosen from which the features representing the verb are generated. In such a mixture model, the number of clusters needs to be determined in advance.

Dirichlet Process Mixture Models (Antoniak, 1974) belong to the class of Bayesian non-parametric mixture models which have received a lot of attention in the machine learning community. With DPMMs, as with other models of this class, the number of mixture components used to represent the data is not fixed in advance, but is determined by the model and the data. The prior G for the parameters of each component θ is generated by a Dirichlet Process (DP) which can be seen as a distribution over other distributions. In turn, each instance is generated by the chosen component given the parameters defined in the previous step. More formally:

$$\begin{aligned} G|\alpha, G_0 &\sim DP(\alpha, G_0) \\ \theta_i|G &\sim G \\ x_i|\theta_i &\sim F(x_i|\theta_i) \end{aligned} \tag{4.2}$$

In Equations 4.2, G_0 and G are probability distributions over the component parameters (θ), and $\alpha > 0$ is the concentration parameter which determines the variance of the Dirichlet process. We can think of G as a randomly drawn probability distribution with mean G_0. As a consequence, G and G_0 are of the same family but have different parameters. Intuitively, the larger α is, the more similar G will be to G_0. For instance x_i, the parameters of its chosen component are θ_i. The graphical model is depicted in Figure 4.1.

The prior probability for assigning instance x_i to either an existing component z or to a new one z_{new} conditioned on the other component assignments (denoted by z_{-i}) is given

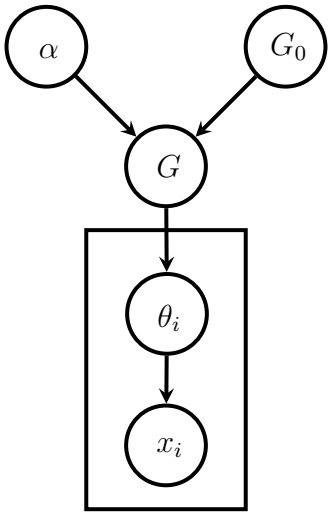

Figure 4.1: Graphical representation of DPMMs.

by:

$$p(z_i = z|z_{-i}) = \frac{n_{-i,z}}{N-1+\alpha} \qquad (4.3)$$
$$p(z_i = z_{new}|z_{-i}) = \frac{\alpha}{N-1+\alpha}$$

where $n_{-i,z}$ is the number of instances assigned to component z excluding instance x_i and N is the total number of instances. A clustering of the instances is generated by assigning more than one instance to the same mixture component. With respect to the application considered in this chapter, components are equivalent to clusters.

The prior in Equation 4.3, exemplifies two main properties of the DPMMs. Firstly, the probability of assigning an instance to a particular component is proportionate to the number of instances already assigned to it ($n_{-i,z}$). In other words, DPMMs exhibit the "rich get richer" property. Secondly, the probability that a new cluster is created is dependent on the concentration parameter α.

A popular metaphor to describe DPMMs is the Chinese Restaurant Process (CRP), a term also used to refer to the prior in Equation 4.3. Customers (instances) arrive at a Chinese restaurant which has an infinite number of tables (components). Each customer chooses to sit at one of the tables that is either occupied ($p(z_i = z|z_{-i})$) or vacant ($p(z_i = z_{new}|z_{-i})$). Popular tables attract more customers.

An alternative view of DPMMs is the stick-breaking construction (Sethuraman, 1994). In this construction, the mixing proportions of the components (π_k) are produced as follows:

$$\pi_k = \beta_k \prod_{j=1}^{k-1}(1-\beta_j) \qquad (4.4)$$
$$\beta_k \sim \mathcal{B}(1,\alpha)$$

where $\mathcal{B}$ is the Beta distribution. It can be verified that $\sum_{k=1}^{\infty} \pi_k = 1$. Intuitively, the mixing proportion of each component is obtained by successively breaking a stick of unit length. As a result, the mixing proportion of a new component gets progressively smaller. In order to generate an instance x_i, the component z_i is chosen using a multinomial distribution parameterized by the mixing proportions π_k, and the instance is generated as in Equation 4.2.

It important to note that in our discussion we did not need to make any assumptions on the probability distribution used to model the data assigned to each component (F in Equation 4.2), nor to the probability distributions used over its parameters (θ in Equation 4.2), namely the distribution G in the same equation. This is due to the fact that the Bayesian framework separates the component assignment prior from the distributions used to model the instances in each component, thus allowing the choice for the latter to depend exclusively on the task in question.

A distribution specified by a set of parameters and the distribution over these parameters are commonly referred to as the likelihood and the prior respectively, which is the relation that holds between $F(x|\theta)$ and $G(\theta|\lambda)$ in Equation 4.2. When the posterior distribution $P(\theta|x, \lambda)$ belongs to the same family as the prior, then it is called the conjugate prior for the likelihood. Conjugate priors have two important properties. First they allow for a closed-form expression for the product $F(x|\theta)G(\theta|\lambda)$ which makes computations easier. Second, they can provide clear intuitions in how they affect the likelihood.

The distribution used to model the components is the multinomial (F in Equation 4.2) and the prior used is its conjugate prior, the Dirichlet distribution (G and G_0 in Equation 4.2). The latter has the following form:

$$Dir(\theta|\lambda) = \frac{\Gamma(\sum_{d=1}^{D} \lambda_d)}{\prod_{d=1}^{D} \Gamma(\lambda_d)} \prod_{d=1}^{D} \theta_d^{\lambda_d - 1} \tag{4.5}$$

where $\theta_d > 0$, $\sum_{d=1}^{D} \theta_d = 1$, D is the dimensionality of the instances (number of features) and the Gamma function (denoted by Γ) is an extension of the factorial function to real and complex numbers. λ is a vector of positive real numbers that denote how many times we have observed the outcome associated parameter θ_d. The Dirichlet distribution, since it is the conjugate prior of the multinomial, it allows analytic integration over the parameters of the multinomial (θ), resulting in the Dirichlet compound multinomial distribution, also known as the Polya distribution:

$$\begin{aligned} Polya(x|\lambda) &= \int_{\theta} Mult(x|\theta) Dir(\theta|\lambda) \\ &= \frac{(\sum_{d=1}^{D} x_d)! \Gamma(\sum_{d=1}^{D} \lambda_d)}{\Gamma(\sum_{d=1}^{D} \lambda_d) + \sum_{d=1}^{D} x_d)} \prod_{d=1}^{D} \frac{\Gamma(\lambda_d + x_d)}{x_d! \Gamma(\lambda_d)} \end{aligned} \tag{4.6}$$

More intuitively, placing a Dirichlet prior over the parameters of the multinomial has a smoothing effect on the latter. Using this interpretation, the vector λ can be seen as

pseudo-counts added to the estimation of the parameters of the multinomial from the data. While the choice of a conjugate prior in this work is due to the computational convenience it provides, it is possible to use non-conjugate priors as well (Cohen and Smith, 2009).

4.3 Inference for DPMMs

Exact parameter estimation for DPMMs is intractable because the number of parameters of the model (θ's and z's in Equation 4.2) is unknown since the number of mixture components is not specified in advance. Research has concentrated on two approaches, Markov Chain Monte Carlo (MCMC) sampling (Neal, 1993) and variational inference (Ghahramani and Beal, 2000). Both approaches have their merits. In particular, sampling techniques are simpler to develop and apply, but assessing convergence is not always straightforward. On the other hand, variational inference is deterministic, providing a straightforward way to assess convergence, but harder to implement. In this work we focus on MCMC sampling methods.

More specifically the parameters of interest in the DPMM are the component assignments (z_i in Equation 4.2). Following Neal (2000), the probability distribution for each assignment conditioned on all the others is given by:

$$P(z_i = z|z_{-i}, x) \propto$$
$$p(z_i = z|z_{-i})Polya(x_i|z_i = z, x_{-i,z}, \lambda) \tag{4.7}$$

The first term is Dirichlet process prior (Equation 4.3) and the second term is the Polya distribution (Equation 4.6) that expresses the likelihood of instance x_i being generated by the component z_i conditioned on the instances already assigned to it and the parameters λ. The latter are the same for all the components. The expression of Equation 4.7 can be evaluated, and therefore be sampled from, relatively easily, since the prior depends only on the component assignments (but not the instances of the components), while the second term depends only on the instances assigned to the component considered. Therefore, it is natural to consider Gibbs sampling, which is one of the most commonly used MCMC sampling methods. In Gibbs sampling, given the conditional distribution for each parameter, we iterate repeatedly reassigning the assignments until convergence, i.e. until we sample assignments from the true distribution of the data. The following equations demonstrate (part of) the procedure to obtain sample $t+1$-th sample:

$$z_1^{t+1} \sim P(z_1^t|z^t, x)$$
$$z_2^{t+1} \sim P(z_2^t|z_{-1}^t, z_1^{t+1}, x)$$
$$z_3^{t+1} \sim P(z_3^t|z_{-1,2}^t, z_{1,2}^{t+1}, x)... \tag{4.8}$$

This sampling scheme is possible due to the fact that the instances in the model are exchangeable, i.e. the order in which they are generated does not affect the overall model. In terms of the CRP metaphor, we consider each instance x_i as the last customer to arrive and he chooses to sit together with other customers at an existing table or to sit at a new table.

In the first iteration of Gibbs sampling (as with other MCMC sampling methods) the component assignments are initialized arbitrarily. As a result, the samples obtained in the first iterations are unlikely to be useful and therefore they are discarded. These iterations are commonly referred to as burn-in period. Furthermore, successive samples obtained by Gibbs sampling are correlated, since a new sample is obtained using the assignments of the previous one. For this reason, it is common practice to keep samples with a lag of few iterations in order to minimize this effect.

The remaining parameter of the DPMM that needs to be estimated is the concentration parameter (α in Equation 4.2) which affects the number of components used to model the data. Antoniak (1974) showed that the posterior distribution for α in the model depends only on the number of components but not on their instances. Following Navarro et al. (2006) we update it with Gibbs sampling using the following equations:

$$\begin{aligned} \alpha|\eta, K, N &\sim Gamma(K + a - 1, b - \ln \eta) \\ \eta|\alpha, K, N &\sim Beta(\alpha, N) \end{aligned} \tag{4.9}$$

In Equation 4.9 *Beta* and *Gamma* are the homonymous distributions, η is an auxiliary variable, a and b are parameters of the *Gamma* distribution, N is the number of instances in the data and K is the number of components. α and η updated each time all the component assignments have been updated.

4.4 Clustering evaluation

The evaluation of unsupervised clustering against a gold standard is not straightforward because the clusters found are not explicitly labeled. Formally defined, an unsupervised clustering algorithm partitions a set of instances $X = \{x_i|i = 1, ..., N\}$ into a set of clusters $K = \{k_j|j = 1, ..., |K|\}$. The standard approach to evaluate the quality of the clusters is to use an external gold standard in which the instances are partitioned into a set of classes $C = \{c_l|l = 1, ..., |C|\}$. Given this, the goal is to find a partitioning of the instances K that is as close as possible to the gold standard C.

Most work on clustering has used either the F-measure (Fung et al., 2003) or the Rand Index (RI) (Rand, 1971) for evaluation, which rely on counting pairwise links between instances. However, Rosenberg and Hirschberg (2007) pointed out that F-measure assumes

(the missing) mapping between c_l and k_j. Also, Meilă (2007) showed that in practice, RI values concentrate in a small interval near 100%. Alternative measures used are Purity and Entropy (Zhao and Karypis, 2001), whose scores improve monotonically with the number of clusters found (Rosenberg and Hirschberg, 2007).

In this work we focus on information theoretic evaluation measures which were shown to conform to the desirable properties defined by Rosenberg and Hirschberg (2007). These measures attempt to assess and balance the two desirable properties that a clustering should have with respect to a gold standard: homogeneity and completeness. Homogeneity is the degree to which each cluster k_j contains instances from a single class c_l. Completeness is the degree to which each class c_l is contained in a single cluster k_j. While an ideal clustering should have both properties, naively improving one of them can be harmful for the other. For example, one can achieve better homogeneity by simply increasing the number of clusters discovered but this is likely to reduce completeness.

The variation of information (VI), introduced by Meilă (2007), assesses homogeneity and completeness using the quantities $H(C|K)$ (the conditional entropy of the class distribution in the gold standard given the clustering) and $H(K|C)$ (the conditional entropy of clustering given the class distribution in the gold standard). The lower these quantities are, the better the clustering is with respect to the gold standard. The final score is obtained by summing them, which means that lower values are better.

However, as Gao and Johnson (2008) pointed out, VI is biased towards clusterings with a small number of clusters. It can be observed that if $|C|$ and $|K|$ are very different then the terms $H(C|K)$ and $H(K|C)$ will not necessarily be in the same range. In particular, if $|K| \ll |C|$ then $H(K|C)$ (and VI) will be low. Also, it is worth mentioning that VI scores are not normalized and therefore they are difficult to interpret. Meilă (2007) presented two normalizations, acknowledging the potential disadvantages they have. The first one normalizes VI by $2\log(max(|K|, |C|))$, which is inappropriate to use for comparisons when the number of clusters discovered $|K|$ changes between experiments. The second normalization involves the quantity $\log N$ which is appropriate when comparing different algorithms on the same dataset (N is the number of instances). However, this quantity depends exclusively on the size of the dataset and hence if the dataset is very large it will result in a normalized VI score misleadingly close to 100%. This does not affect rankings, i.e. a better VI score will also be translated into a better normalized VI score.

In their comparative study of clustering evaluation methods, Rosenberg and Hirschberg (2007) proposed V-measure as an alternative information-theoretic metric. In V-measure homogeneity is computed as the conditional entropy of the class distribution of the gold standard given the clustering discovered by the algorithm, $H(C|K)$, normalized by the entropy of the class distribution in the gold standard, $H(C)$. Completeness is computed as the conditional entropy of the cluster distribution discovered by the algorithm given the class, $H(K|C)$, normalized by the entropy of the cluster distribution, $H(K)$. In both

cases, the resulting ratios are subtracted from 1 to associate higher scores with better solutions and their harmonic mean is the final score:

$$hom = 1 - \frac{H(C|K)}{H(C)}$$
$$comp = 1 - \frac{H(K|C)}{H(K)}$$
$$V_\beta = \frac{(1+\beta) * hom * comp}{(\beta * hom) + comp} \tag{4.10}$$

The parameter β in Equation 4.10 regulates the balance between homogeneity and completeness. Rosenberg and Hirschberg (2007) set it to 1 in order to obtain the harmonic mean of these qualities. They also note that V-measure favors clustering solutions with a large number of clusters (large $|K|$), since such solutions can achieve very high homogeneity while maintaining reasonable completeness. This effect is more prominent in datasets with a small number of instances and a relatively large number of gold standard classes. Unlike Purity and Entropy, increasing $|K|$ does not guarantee an increase in V-measure (splitting homogeneous clusters would reduce completeness without improving homogeneity), however it is easier to achieve higher scores when more clusters are produced.

Both the V-measure and VI have important advantages over the RI and F-measure: they do not assume a mapping between classes and clusters and their scores depend only on the relative sizes of the clusters. However, the V-measure and VI can be misleading if the number of clusters found ($|K|$) is substantially different from the number of gold standard classes ($|C|$). In order to ameliorate this, we propose a variation of the V-measure, V-beta, that takes advantage of the β parameter in Equation 4.10 in order to balance homogeneity and completeness. More specifically, setting $\beta = |K|/|C|$ assigns more weight to completeness than to homogeneity in case $|K| > |C|$ since the former is harder to achieve and the latter is easier when the clustering solution has more clusters than the gold standard has classes. The opposite occurs when $|K| < |C|$. In case $|K| = |C|$ the score is the same as the original V-measure. Achieving a perfect score according to any of these measures requires correct prediction of the number of clusters.

4.5 Experiments

In our experiments we use a dataset consisting of 193 medium to high frequency verbs obtained from a corpus of 2230 full-text articles from 3 biomedical journals, which was first introduced by Korhonen et al. (2006). The features for each verb are its subcategorization frames (SCFs) and associated frequencies in corpus data, which capture the syntactic context in which the verb occurs. Previous research has shown these features to be indicative of verb semantics, since they provide us with generalizations over their

gold standard	*bio-16*	*bio-34*	*bio-50*
homogeneity	75.9%	69.89%	67.92%
completeness	62.57%	77.86%	85.23%
V-measure	68.59%	73.65%	75.59%
V-beta	66.42%	73.55%	73.80%
VI	2.4984	2.4281	2.3991

Table 4.1: Performance of the DPMMs on biomedical verb clustering.

linguistic properties (Levin, 1993) and have been applied successfully to semantic verb clustering in general English (Korhonen et al., 2003). Given their nature though, they are unlikely to be able to distinguish between verbs whose semantics differ but not their syntax. Such cases occur when verb semantics is altered through the use of particles, e.g. "activate" vs. "disactivate". SCFs were extracted automatically using the domain-independent statistical parsing toolkit RASP (Briscoe and Carroll, 2002) and a classifier which identifies verbal SCFs (Briscoe and Carroll, 1997; Korhonen, 2002) . As a consequence, some noise was introduced due to standard text processing and parsing errors and due to the subtlety of the argument-adjunct distinction.

The feature sets based on verbal SCFs are very sparse and the counts vary over a large range of values. This can be problematic for generative models like DPMMs, which attempt to identify the process that generated the instances instead of distinctions among them and a few dominant features are likely to influence inappropriately the clustering discovered. To reduce the sparsity and the variance of the feature set, we applied non-negative matrix factorization (NMF) (Lin, 2007) which decomposes the dataset in two dense matrices (of lower dimensionality) with non-negative values. It has proven useful in a variety of tasks, e.g. information retrieval (Xu et al., 2003) and image processing (Lee and Seung, 1999). The new features derived are linear combinations of the original ones.

A team of domain experts and linguists were involved in creating a gold standard for this dataset. The former analyzed the verbs requiring domain-knowledge and the latter the general English and/or scientific ones. This effort resulted in a hierarchical gold standard with three levels of increasing granularity: 16, 34 and 50 classes.

In our experiments the parameters of the Dirichlet prior on the component parameters (λ vector in Equation 4.7) were set to 1. The number of dimensions obtained using NMF was 35. We initialize the Gibbs sampler 5 times, using 100 iterations for burn-in and draw 20 samples from each run with 5 iterations lag between samples. Table 4.1 shows the average performances against each version of the gold standard using the three information-theoretic evaluation measures discussed in Section 4.4, VI, V-measure and V-beta, as well as the individual homogeneity and completeness scores as computed in Equation 4.10.

The DPMM discovers 32.38 verb clusters on average, which is closer to the version of the gold standard of medium granularity (*bio-34*). As we move from the coarsest version of the gold standard (*bio-16*) towards the finest one (*bio-50*) we can observe that completeness improves while homogeneity deteriorates. This is expected, since smaller classes can be contained more easily in a single cluster, while clusters are more likely to be homogeneous when larger classes are considered. Furthermore, we can observe that VI scores correlate with completeness, while V-measure scores are higher when finer versions of the gold standard are considered. V-beta, by balancing homogeneity and completeness with respect to the number of classes in the gold standard, results in lower scores than the V-measure, since it downweighs the high completeness achieved against *bio-16* and the high homogeneity achieved against *bio-50*.

As explained earlier, in our experiments we use multiple runs and samples from them in order to obtain more reliable estimates of our performance by reporting averaged scores. This process results also in multiple clustering samples and it would be convenient to be able to average over them in order to obtain a single clustering solution from the model. However this is not possible, since the clusters in a particular sample cannot be identified with any of the clusters in a different sample. Another option would be to identify a particular sample as representative of all the samples, by measuring the average distance of each sample to all the others. But such a sample could contain clusters that appeared only in that sample and it would be hard to tell which of the clusters contained were responsible for it being representative and which are not.

In order to obtain a single clustering from the model we represent each clustering sample as a linking matrix between the instances of the dataset and measure the frequency of each pair of instances occurring in the same cluster. We then construct a partial clustering of the instances using only those links that occur with frequency higher than a threshold *prob_link*. Singleton clusters are formed by considering instances that are not linked with any other instances more frequently than a threshold *prob_single*. The lower the *prob_link* threshold, the larger the clusters will be, since more instances get linked. Note that including more links in the solution can either increase the number of clusters when instances involved were not linked otherwise, or decrease it when linking instances that already belong to other clusters. The higher the *prob_single* threshold, the more instances will end up as singletons. By adjusting these two thresholds we can affect the coverage of the analysis since it alters the number verbs that are allowed to appear either linked in a cluster or as singletons. Such post-processing can be used to conduct qualitative analysis of data that is relevant to most clustering samples and irrespective of individual samples, as well as to use the output of the clustering algorithm as a component in a pipeline which requires a single result rather than multiple samples.

We applied this analysis to the samples obtained from the DPMM in our experiments and we present some of the clusters and the singletons recovered in Table 4.2. The thresholds used were 0.9 for the *prob_link* and 0.1 for the *prob_single*, i.e., the resulting links between

	verb	*bio-16*	*bio-34*	*bio-50*
Cluster1	analyze	9	9.1	9.1.1
	assess	9	9.1	9.1.1
	evaluate	9	9.1	9.1.1
	measure	10	10.1	10.1.1
	monitor	10	10.1	10.1.1
	quantify	10	10.1	10.1.1
	quantitate	10	10.1	10.1.1
Cluster2	establish	9	9.1	9.1.2
	investigate	9	9.1	9.1.2
	test	9	9.1	9.1.2
Cluster3	hypothesize	9	9.2	9.2.1
	imply	9	9.3	9.3
	note	9	9.2	9.2.1
	reason	9	9.2	9.2.1
	speculate	9	9.2	9.2.1
	suggest	9	9.3	9.3
Cluster4	determine	9	9.1	9.1.3
Cluster5	cotransfect	6	6	6
	inject	6	6	6
	microinject	6	6	6
	transfect	6	6	6
Cluster6	centrifuge	7	7.2	7.2
	incubate	4	4.1	4.1.4
Cluster7	increase	1	1.3	1.3
	decrease	1	1.3	1.3

Table 4.2: Aggregate sample of the DPMM on the biomedical verb dataset. For each verb we add the class it is found with in each version of the gold standard.

verbs exist in at least 90% of the samples obtained. The DPMM recovers some coarse distinctions (e.g. Cluster5 vs. Cluster3) as well as some of the finer ones (e.g. Cluster2 vs. Cluster3). Also, it must be noted that it fails to capture some distinctions that exist in the coarser version of the gold standard. These failures can be attributed either to (relative) semantic proximity (e.g. “assess” vs. “measure” in Cluster1), or to inadequacy of the features to capture the distinction desired (e.g. “centrifuge” vs. “incubate”).

4.6 Discussion - Related work

Previous work on unsupervised verb clustering used algorithms that require the number of clusters as input, e.g. pairwise clustering (Puzicha et al., 2000), the information bottleneck method (Tishby et al., 1999) and spectral clustering (von Luxburg, 2006). In particular, Korhonen et al. (2006) who introduced the dataset used in our experiments obtained their best performances using the information bottleneck method and adjusting the number of clusters discovered by the algorithm taking into account the version of the gold standard used for evaluation. While the actual performances are not comparable with ours due to the use of different evaluation measures and differences in post-processing of the features, the performances achieved in the previous section are in the same range. In particular, when fixing the number of clusters discovered to 33, Korhonen et al. (2006) evaluating against *bio-16*, *bio-34* and *bio-50* report performances of 65%, 77% and 77% in F-score respectively. We argue though that, independently of the performance achieved, when performing clustering on an unknown dataset it is impossible to know the number of clusters in advance and this is the main advantage of the DPMMs over other clustering methods. It can be argued that one could use clustering algorithms that require the number of clusters to be known in advance to discover interesting sub-classes such as those discovered by the DPMMs. However, this would normally require multiple runs and manual inspection of the results, while DPMMs discover them automatically. If the goal is to discover a clustering that follows some prior intuition on its structure, this can be achieved through the use of constraints which is the focus of the next chapter.

On the issue of clustering evaluation methods, it is important to point out that evaluation against gold standard classes is not always the best way to assess the quality of the clustering solution returned by an algorithm. Depending on the application, extrinsic evaluation can provide useful insights and more reliable results. Performing such an evaluation is challenging because unsupervised methods do not return meaningful labels in their output. For example, DPMMs (like any other clustering algorithm) return cluster identifiers instead of meaningful labels such as *MOTION* or *SPECULATION*. Such per cluster labels can be obtained if needed by a human expert, in which case the clustering discovered in an unsupervised fashion would reduce the amount of work needed since such labels would apply to all the verbs of each cluster and the clusters.

Semantic clustering bears resemblance to automatic term recognition (Nenadic et al., 2004; Wermter and Hahn, 2005; Korkontzelos et al., 2008) whose aim is the identification of domain-specific terms in corpora. In this respect, semantic clustering produces a richer output by dividing the terms further into semantic classes. It is worth pointing out that in the gold standard of the biomedical verb dataset used in our experiments the scientific verbs were in separate clusters from the general English ones.

The verb clusters found by the DPMM could be incorporated in aspects of the event extraction task. For example, Cluster3 in Table 4.2 could be used to detect speculative

clauses which would be useful in Task 3 of the BioNLP 2009 shared task (Kim et al., 2009). Thus, it becomes possible to build a event extraction system without requiring a dictionary provided by domain experts. While verb clustering does not assign semantically meaningful labels, we argue that these can be obtained from the clusters themselves, for example Cluster7 in Table 4.2 could be identified as containing verbs denoting REGULATION class events.

It is important to note though that in unsupervised clustering, we rely exclusively on the feature set in order to learn the desired distinctions. In our experiments we used subcategorization frames which, while indicative of semantics, cannot distinguish between pairs of verbs such as "increase" and "decrease" (Cluster7 in Table 4.2), which typically denote different event types in the same shared task. Nevertheless, studying the behaviour of the words in a domain in an unsupervised way can provide useful insights upon which simple yet effective IE systems can be developed, such as the rule-based event extraction system developed in Chapter 3 which used the findings of Cohen et al. (2008). In a parallel example from a different and less explored domain, Goldberg et al. (2009) used Latent Dirichlet Allocation (Blei et al., 2003) in order to explore a corpus of wishes and built a system that extracts them automatically.

Recent work has explored a variety of features for supervised classification of biomedical verbs (Korhonen et al., 2008) and it would be of interest to assess them in the context of unsupervised clustering. Furthermore, we did not attempt to capture the phenomenon of verb polysemy which was explored in Korhonen et al. (2003). Also, it would be of interest to apply the method presented by Fritsch and Ickstadt (2009) to aggregate clustering samples from DPMMs.

Finally, the development of a multi-level gold standard for the dataset used indicates that hierarchical clustering is needed to capture such structure. Bayesian hierarchical clustering has been the subject of ongoing studies in the machine learning community (Heller and Ghahramani, 2005; Teh et al., 2007) and it would be of interest to evaluate the methods proposed in linguistic applications. For this purpose, it is important to develop suitable evaluation methods that can take into account multiple levels of hierarchy.

4.7 Summary

In this chapter we explored the application of Dirichlet Process Mixture Models to the task of biomedical semantic verb clustering. The main advantage of this method is that, unlike previously applied algorithms, it discovers the number of clusters in the data instead of requiring it as input. In order to evaluate the performance, we modified V-measure (Rosenberg and Hirschberg, 2007) to deal more appropriately with the varying number of clusters discovered by DPMMs and presented a method of aggregating the samples generated which allows for qualitative evaluation.

Chapter 5

Constrained semantic verb clustering

5.1 Introduction

Clustering using Dirichlet process mixture models (DPMMs) is an effective method of grouping biomedical verbs according to their semantics that does not require manual annotation or the number of clusters to be known in advance. While this is attractive, in many cases it is also desirable to influence the solution with respect to some prior intuition or consideration relevant to the application in mind. Evidence for this is provided by the three-level gold standard constructed by human experts for the biomedical verb dataset used in Chapter 4. Furthermore, while some applications might require coarse-grained classes, others may benefit from a fine clustering or a clustering that reveals a specific aspect of the dataset. For example, "phosphorylate" and "cleave" both denote a biochemical modification (class 2.2 in *bio-34* of the gold standard used in Chapter 4), but in the context of the BioNLP 2009 event extraction shared task (Kim et al., 2009) they denote different event types.

For this purpose, we introduce a constrained version of DPMMs (CDPMMs), that enables human supervision to guide the clustering solution. Following Wagstaff and Cardie (2000), the human supervision is modelled as pairwise constraints over instances: given a pair of instances, either they should be clustered together (*must-link*) or not (*cannot-link*). This information can be obtained either from a human expert, or by appropriate manipulation of extant resources, such as ontologies. Specifying the relations between the instances results in an indirect labeling of the instances. Such labeling is likely to be re-usable, since it defines relations between the instances rather than explicit labels, and relations are more likely to hold across multiple tasks.

From the point of view of the verb clustering task, the incorporation of constraints in the DPMM results in a semi-supervised clustering model. Therefore it is natural to consider active learning (AL) in order to select the supervision added to the model. To this end, we propose a simple yet effective method of selecting pairwise constraints for inclusion.

We begin this chapter by describing how we incorporate pairwise constraints in the DPMM (Section 5.2). We proceed with a brief overview of active learning (Section 5.3) and how we apply it to the selection of pairwise constraints for clustering with CDPMMs (Section 5.4). We demonstrate the potential of the method presented on the biomedical verb dataset of Korhonen et al. (2006) (Section 5.5) and conduct some further experiments on batch selection (Section 5.6). We conclude with a discussion of related approaches and future work (Section 5.7).

5.2 Constrained DPMMs

In order to incorporate constraints in the DPMM, we modify the underlying generative process to take them into account. In particular, *must-linked* instances are generated always by the same component and *cannot-linked* instances always by different ones. In terms of the Chinese restaurant metaphor, customers connected with *must-links* arrive at the restaurant together and choose a table jointly, respecting their *cannot-links* with other customers. They get seated at the same table successively one after the other. Customers without *must-links* with others choose tables avoiding their *cannot-links*. In the discussion and the experiments that follow, we assume that all links are consistent with each other.

In order to sample the component assignments according to this model, we restrict the Gibbs sampler to take them into account using the sampling scheme of Algorithm 2. First we identify *must-linked* groups of instances, taking into account transitivity.[1] We then sample the component assignments only from distributions that respect the links provided. More specifically, for each instance that does not belong to a *must-linked* group, we restrict the sampler to choose components that do not contain instances *cannot-linked* with it. For instances in a *must-linked* group, we sample their assignment jointly, again taking into account their *cannot-links*. This is performed by adding each instance of the *must-linked* group successively to the same component. In Algorithm 2, $\mathcal{C}_i$ are the *cannot-links* for instance(s) i, ℓ are the indices of the instances in a *must-linked* group, and $z_{<i}$ and $x_{<i}$ are the assignments and the instances of a *must-linked* group that have been assigned to a component before instance i.

It is important to note that random selection of pairwise constraints to incorporate can be very ineffective since a large number of binary links are likely to have been captured by the unsupervised DPMM and therefore will not have any effect on the clustering obtained if added. E.g., in the experiments of Section 4.5 the DPMM predicts more than 90% of the binary links correctly. For this reason, we develop an active constraint selection method in the following sections. A comparison between random and active selection is presented in Section 5.5.

[1] If A is linked to B and B to C, then A is linked to C.

Algorithm 2 Gibbs sampler incorporating *must-links* and *cannot-links*.

data $\mathcal{X}$, *must-links* $\mathcal{M}$, *cannot-links* $\mathcal{C}$
must_linked = find_must_linked_groups($\mathcal{X}$, $\mathcal{M}$)
Initialize Z according to $\mathcal{M}$, $\mathcal{C}$, *must_linked*
for i **not in** *must_linked* **do**
  for $z = 1$ **to** $|Z| + 1$ **do**
    if $x_{-i,z} \cap \mathcal{C}_i = \emptyset$ **then**
      $P(z_i = z|z_{-i}, x_i)$ (Eq. 4.7)
    else
      $P(z_i = z|z_{-i}, x_i) = 0$
  Normalize and Sample from $P(z_i)$
for ℓ **in** *must_linked* **do**
  for $z = 1$ **to** $|Z| + 1$ **do**
    if $x_{-\ell,z} \cap \mathcal{C}_\ell = \emptyset$ **then**
      Set $P(z_\ell = z|z_{-\ell}, x_\ell) = 1$
      for i **in** ℓ **do**
        $P(z_\ell = z|z_{-\ell}, x_\ell) * = P(z_i = z|z_{-\ell}, x_{-\ell,z}, z_{<i}, x_{<i})$
    else
      $P(z_\ell = z|z_{-\ell}, x_\ell) = 0$
  Normalize and Sample from $P(z_\ell)$

5.3 Active learning

In the active learning framework, the statistical learning model iteratively selects the instances on which it is going to be trained. In the widely used pool-based approach,[2] we start with a small labeled training set L and a large pool of unlabeled data U. In each round, a model is trained on L and it is used in order to select a batch B of instances from U which are considered to be informative. These are annotated by a human, added to L and the loop is repeated.

The main point of differentiation among the various active learning algorithms is the method of assessing the informativity of an instance. The two most popular active learning methods used in NLP are uncertainty-based sampling (Lewis and Gale, 1994) and query by committee (Seung et al., 1992). In uncertainty-based learning, the instances selected to be annotated are those on which the classifier is least certain of their classification. The assumption is that instances which are harder to classify are more useful to train the classifier on. The uncertainty of the classifier is commonly estimated using the entropy of its output in the case of probabilistic models. For non-probabilistic ones, the classification margin is used, as in the case of support vector machines (Tong and Koller, 2002).

[2]Settles (2009) contains a detailed overview of various active learning approaches.

Algorithm 3 Active Learning for constrained DPMMs.

1: **Input:** data X, batch size B, seed constraints $\mathcal{SC}$
2: Generate samples S using the DPMM from X constrained by $\mathcal{SC}$
3: **while** budget not exhausted and/or stopping criterion not reached **do**
4: Decompose S into binary links L
5: Rank L according to their uncertainty
6: Obtain labels for the top-B links and add the to $\mathcal{SC}$
7: Generate a new set of samples S using the DPMM constrained by $\mathcal{SC}$

In query by committee, a committee of classifiers is trained on L, then applied to the instances of U and those which result in the highest disagreement among the classifiers are considered to be the most informative. Common ways of estimating the disagreement are the vote-entropy (Argamon-Engelson and Dagan, 1999) and the Kullback-Leibler divergence (Pereira et al., 1993). In this work, we focus on uncertainty-based sampling since we are interested in exploring the potential of the CDPMM presented in Section 5.2.

5.4 Active constraint selection

In this work we employ the simple but effective idea of uncertainty based sampling in order to perform active learning with constrained DPMMs. We consider the most informative links as those on which the model is most uncertain. In order to pick the most uncertain links, we identify pairs of instances on which the model is highly uncertain whether they should be generated by the same component or not. During the sampling process used for parameter inference, component assignments vary from sample to sample and the components themselves are not identifiable, i.e. one cannot match the components of one sample with those of another. Therefore, we resort to generating a set of samples from the (possibly constrained) DPMM and we pick the link the samples maximally disagree on. If we consider clustering as binary classification of links into *must-links* (M) and *cannot-links* (C), this is identical to picking the link l' with the highest entropy:

$$l' = argmax_{l \in \mathcal{L}} \{-p(l \in \mathcal{M}) log(p(l \in \mathcal{M})) - p(l \in \mathcal{C}) log(p(l \in \mathcal{C}))\} \tag{5.1}$$

While obtaining multiple samples from the model can be awkward, in practice it is necessary in order to obtain a more reliable estimation of the model's performance given the stochastic inference used in the experiments of Section 4.5. Also, using multiple samples instead of a single sample has been found beneficial when the output of the model is an intermediate stage in a pipeline (Wei and Croft, 2006). The active constraint selection is described step-by-step in Algorithm 3.

Compared to the standard pool-based AL scenario, in the case of clustering with constraints the number of possible links between two instances (ignoring transitivity) are

$C(N, 2) = N(N-1)/2$ and there is an equal number of candidate queries to be considered (N is the size of the dataset), as opposed to N queries in the case of a supervised classification task. A potential issue that can arise from this is that in step 4 of Algorithm 3 the memory space requirements are quadratic in the size of the dataset, which can be prohibitive in the case of large datasets. In our experiments, the modest size of the dataset used (193 instances) did not lead to any excessive requirements, but for larger datasets, randomized data structures could be considered (Bloom, 1970). Alternatively, if a full ranking of the links is not required, one could keep in memory only the $|B|$ most uncertain links.

Another interesting difference is that the the AL process can be initiated without any supervision, since the DPMM is unsupervised. This is an interesting difference compared to the standard AL scenario where a (usually small) labeled seed set is always used. A consequence is that, assuming we use no seed constraints, we rely exclusively on the model and the features to guide the constraint selection process. If the model combined with the features is not appropriate for the task then the constraints chosen by the model are not likely to be useful. This can provide insights to the appropriateness of the features and the model with respect to clustering desired.

5.5 Active learning experiments

To investigate the effectiveness of the AL method presented in Section 5.4 we conduct experiments in which we compare the performance using the constraints selected actively versus the performance achieved when the constraints are selected randomly. The random selection is repeated 3 times and the results are averaged. In these experiments we use the biomedical verb dataset and features from Section 4.5, keeping all experimental conditions constant. In each AL round, we run the Gibbs sampler for the (constrained) DPMM 5 times, using 100 iterations for burn-in, draw 20 samples from each run with 5 iterations lag between samples and select the most uncertain link to be labeled. The performances were averaged across the collected samples. Each AL experiment is initialized without any constraints, i.e. the unsupervised DPMM is the starting point in each of the learning curves. Each of the three levels of the biomedical gold standard is used independently, resulting in 3 experimental setups in total and the results are shown in the graphs of Table 5.1. In each case, we extract links from the version of the gold standard we evaluate against. We present results only using V-beta, noting that the observations that follow hold across all evaluation measures used in Section 4.5. Also, we report independent homogeneity and completeness scores for the active learning runs.

In all the graphs of Table 5.1, the unsupervised performance achieved using the DPMM in the previous chapter is represented by the leftmost point in each of the curves, i.e. when no links have been added. A first observation is that incorporating constraints

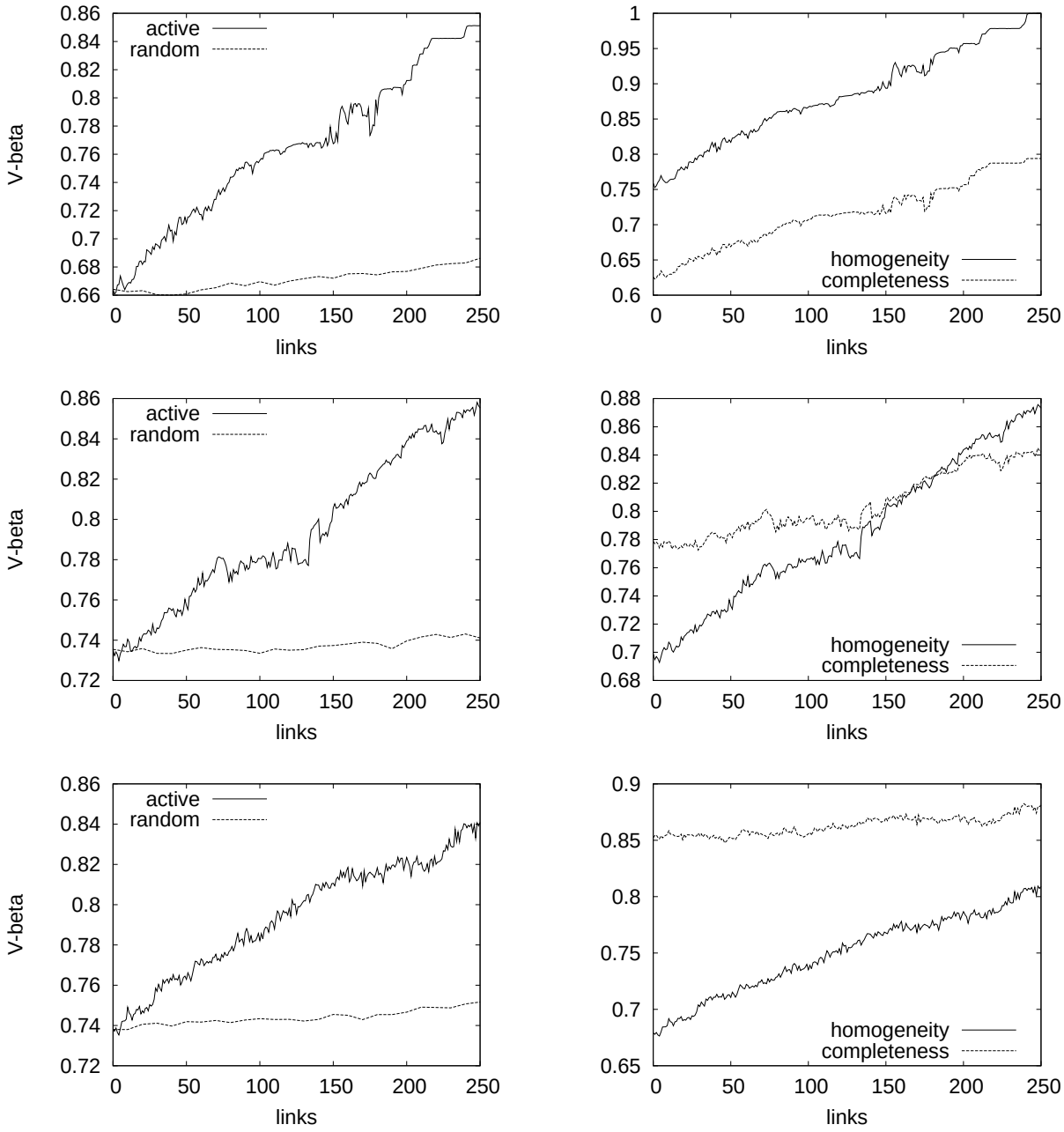

Table 5.1: Constrained DPMM learning curves. From top to bottom: *bio-16*, *bio-34*, *bio-50*. On the left, comparison between AL and random selection, on the right homogeneity and completeness curves during AL.

always improved performance, even though random selection of links improves it rather slowly. On the other hand, constraints selected via AL on the other hand improve the performance quite rapidly. Indicatively, the performance reached using 1000 randomly chosen constraints is obtained using only 200, 150 and 110 actively selected ones in *bio-16*, *bio-34*, *bio-50* datasets respectively. Another interesting observation is that as the AL process progressed, in all experiments homogeneity increased faster than completeness. This suggests that the features lead the model towards finer-grained clusters, which is

confirmed by the fact that the highest scores are achieved when comparing against *bio-50*. In particular, when extracting constraints from the latter version of the gold standard completeness does not increase (it is already above 85%) since the model queries links that are mostly *cannot-links*. While it would be possible to add constraints to the model that would force it towards the gold standard, we argue that by incorporating constraints that are informative according to the model we can obtain insights on the model and the features used that would otherwise be obscured.

With respect to practical applications that would build upon a clustering produced in this fashion, we expect that the very good homogeneity achieved will be more important than the completeness, since it is conceptually easier to label two homogeneous clusters as belonging to the same class, rather than splitting heterogeneous clusters into multiple homogeneous ones. This was confirmed in the experiments of Van Gael et al. (2009) who demonstrated that, when evaluating unsupervised part-of-speech (PoS) tagging, homogeneity rather than completeness correlated with the performance of a shallow parser using the PoS tags learned as features.

5.6 Batch selection

A common consideration in active learning experiments is the size of the batch of instances to annotate that is presented to the human annotators in every round. The steepest learning curves are commonly achieved when a single instance is annotated, which allows the model to update the ranking of the unlabeled instances most frequently, as in the experiments of Section 5.5. However, this implies that a human annotator annotates one instance (pairwise constraint in our experiments) and then needs to wait for the parameters of the model to be inferred again. Therefore, using larger batches would be a more realistic scenario.

In Table 5.2 we present learning curves using batches of 10 in which the learning rate, while higher than random selection, is lower than that of active learning with a single constraint selected in each round. This phenomenon has been commonly observed to varying degrees in many applications of AL. In our experiments, after some manual inspection of the links chosen at each round by the model, we noticed that the batch selected in each round very often contained links involving the same instances. This can be expected, given that an instance linked uncertainly with another instance, is likely to be uncertainly linked with a third one too.

In order to ameliorate this issue we modify Algorithm 3 in the following way. After obtaining the label of the most uncertain link, the samples that disagree with it are removed and the link uncertainties are re-calculated given the remaining samples. This process is repeated until the intended batch size is reached. By removing samples in this

fashion, we avoid selecting links involving the same instance, unless the uncertainty over its links is not reduced by the constraints already added.

A consideration that arises is that by reducing the number of samples used for uncertainty estimation, progressively we are left with very few samples to rank the remaining links. Each link reduces the number of samples approximately by half since the most uncertain link is likely to be a *must-link* in half the samples and a *cannot-lnk* in the remaining half. As a result, for a batch with size $|B|$ the uncertainty of the last link will be estimated using $|S|/2^{|B|-1}$ samples. A crude solution would be to generate enough samples for the batch size we intend to obtain. However, obtaining a very large number of samples is computationally expensive. Therefore, we set a threshold for the minimum number of samples to be used to estimate the link uncertainty and when there are fewer samples, more samples are generated with the Gibbs sampler using these remaining samples for assignment initialization. This results in requiring fewer samples to be generated. In the experiments of Figure 5.2 when 20 (or fewer) samples were left to estimate uncertainty, each of them was used to generate new samples so that 100 samples could be used. As it can be observed, this batch selection process improves the learning rate over the standard batch selection of the same size, and it is competitive with selecting only one constraint in each active learning round.

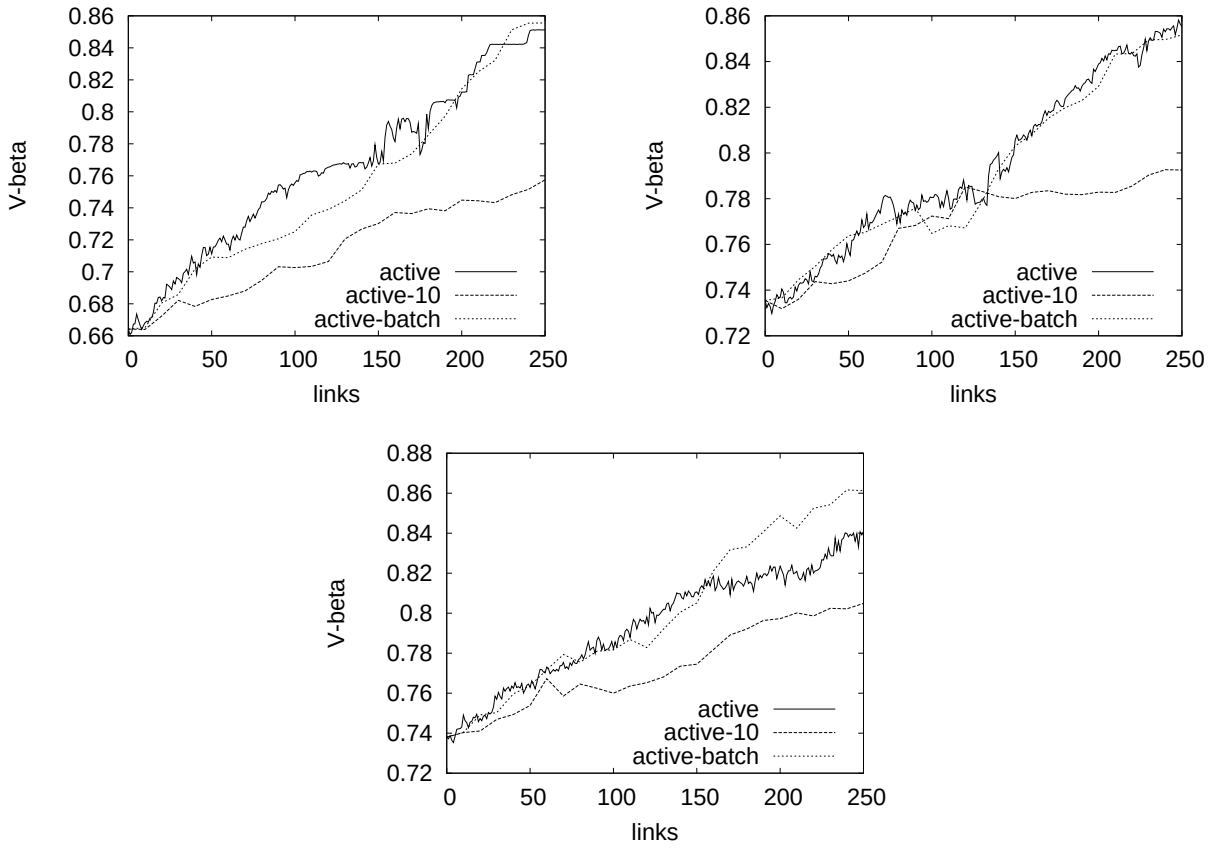

Table 5.2: Experiments with batch selection: *bio-16* (top-left) , *bio-34* (top-right) and *bio-50* (bottom).

5.7 Discussion - Related work

There is a large body of work on semi-supervised learning (SSL), mainly for classification and sequence labeling tasks, but relatively little work has been done on incorporating some form of supervision in clustering. It is important to note that the pairwise links used in this work constitute a weak form of supervision since they cannot be used to infer class labels as is required for standard SSL. However, the opposite can be done. Wagstaff and Cardie (2000) employed *must-links* and *cannot-links* to constrain the COBWEB algorithm, while Klein et al. (2002) applied them to complete-link hierarchical agglomerative clustering. The latter also studied how the added links affect instances not directly involved in them. Apart from the fact that fixing the number of clusters in advance restricts the discovery of novel information in the data, such algorithms cannot take full advantage of the pairwise constraints, since the latter are likely to change the number of clusters. In related work, Andrzejewski et al. (2009) used *must-links* and *cannot-links* to guide the discovery of topics by Latent Dirichlet Allocation (Blei et al., 2003).

There is a substantial amount of literature on AL for various NLP tasks, most of it involving supervised learning methods for classification or sequential tagging. In some cases, biomedical NLP tasks were used in order to assess the effectiveness of the methods presented (Settles and Craven, 2008; Tomanek and Hahn, 2009). The closest task to semantic verb clustering is word sense disambiguation (WSD) for which semi-supervised learning methods were recently explored in the biomedical domain by Stevenson et al. (2008).

However, AL for clustering is a relatively under-explored area. To our knowledge, Basu et al. (2006) and Klein et al. (2002) have applied AL to obtain *must-links* and *cannot-links* and used them for semi-supervised clustering. However, the clustering frameworks they used (hierarchical agglomerative clustering and hidden Markov random fields respectively) require the number of clusters to be known in advance. As explained already, this restricts counter-intuitively the clustering solutions that are discovered. Finally, since semi-supervised clustering is a form of semi-supervised learning, our approach is related to those of Zhu et al. (2003) and Liang et al. (2009) who combined AL with semi-supervised learning for classification and sequential tagging respectively with good results.

With respect to the practical application of the AL method suggested, an obvious extension is to apply it to other datasets and clustering tasks in the biomedical domain. In particular, the idea of combining multiple samples with uncertainty based sampling, while explored in the context of clustering with the constrained DPMM, is likely to be applicable in other models that can incorporate *must-links* and *cannot-links* and where during parameter learning multiple samples are obtained.

An interesting extension would be to incorporate it in an interactive curation-assistance environment which would allow its users to modify the grouping of biomolecular events

interactively based on the constraints they provide the system with. Thus, an IE system using the clusters discovered would be able to adapt its results on-the-fly according to the users' preferences. As described in Section 4.6, such clusters can be used in event extraction to replace dictionaries. For example, a user might want to cluster together Transcription and Gene_expression events (as defined in Chapter 3) and it would be enough to provide a few must-linked pairs of verbs to achieve this, since the constrained DPMM would take them into account. This would be more straightforward than having to change feature representations or fix the number of clusters to achieve a similar result.

It is worth noting that in all our experiments the constraints were obtained from the respective gold standard of the dataset in question and consequently they are all consistent with each other. However, this assumption might not hold if human experts are employed for the same purpose. Inconsistencies can arise either due to human error or due to genuine disagreement between annotators. In order to use such feedback in the framework suggested, it is necessary to filter the constraints provided in order to maintain a consistent subset of them. This aspect might be crucial if one considers crowd sourcing as a method to obtain supervision, such as the Amazon Mechanical Turk (Snow et al., 2008).

Other directions for future work could include applying AL methods to non-parametric Bayesian models whose parameters are estimated using variational inference. This would allow taking advantage of "soft" assignments of instances to components. Also, it would be interesting to investigate the potential of using "soft" constraints, i.e. constraints that are provided with relative instead of absolute confidence. This could provide an elegant approach to remove the constraint consistency requirement that the framework presented has and it is likely to be useful in realistic application scenarios.

5.8 Summary

In this chapter we introduced a constrained version of the Dirichlet Process Mixture Models (CDPMMs) that allow us to incorporate of *must-link* and *cannot-link* constraints to adapt the clustering solution towards a user's prior intuition with minimal supervision. Furthermore, we presented an active learning method for CDPMMs based uncertainty based sampling, that selects links that adapt the model faster to the desired clustering. We demonstrated the potential of the methods proposed on a biomedical verb dataset with a three-level gold standard.

Chapter 6

Conclusions - Future Work

As stated in Chapter 1, the goal of this thesis was to develop semi-supervised approaches for biomedical information extraction tasks. We believe that we have succeeded in this goal and that during this process we have made some contributions that would be applicable in other domains.

In Chapter 2 we tackled biomedical named entity recognition. For this purpose, we implemented a previously suggested approach to generate training material using a dictionary of gene names combined with raw abstracts. In replicating these experiments we were able to confirm the benefits of this approach, as well as observe some annotation issues that we found to have substantial impact on the evaluation. Our contributions begin by developing new guidelines and datasets that helped evaluate the system built with greater confidence and identify its strengths and weaknesses. Unlike the majority of the literature in biomedical named entity recognition, we evaluated our performance on abstracts and full papers. This analysis prompted us to develop a new system combining conditional random fields with syntactic parsing which improved performance on full papers. Our final contribution is that we were able to improve it further by obtaining feedback from users in the context of a real working environment. Overall, the performances achieved were competitive with systems that use manually labeled data for training, which we believe that it makes a strong case for exploring and adapting such methods to other tasks and domains.

In Chapter 3 we focused on bio-molecular event extraction in the context of the BioNLP 2009 shared task (Kim et al., 2009). Initially we developed a simple rule-based system which, combined with a dictionary of lemmas associated with event types and the output of a syntactic parser, performed competitively against machine learning based systems that used fully annotated training data. We believe this to be a two-fold contribution. On the one hand, it demonstrated that machine learning methods are not guaranteed to achieve better performance than the rule-based ones, even when a substantial amount of training data is available. On the other hand, it revealed the aspects of the task for which a rule-based approach would become rather complex and awkward to implement

in order to achieve good performance. This led to an improved system that uses Support Vector Machines trained on partial event annotation which would achieve the second best performance in the context of the shared task. Apart from achieving reasonable performances with little manual annotation, our experiments highlighted the value of domain-adapted syntactic parsing for the task, as well as some annotation and evaluation issues.

In Chapter 4 we performed biomedical semantic verb clustering which allows us to explore the behaviour of verbs in the domain without any manual annotation. For this purpose, we used Dirichlet Process Mixture Models which, unlike clustering algorithms previously applied to this task, discover the number of clusters in the data instead of requiring it as input. Nevertheless, the performance achieved was competitive with existing approaches. Such methods can be used to explore the semantics of other datasets and domains without restricting the space of possible solutions. Another contribution is that we introduced V-beta, a variant of an existing clustering evaluation measure that is able to deal appropriately with the varying number of clusters discovered when evaluating the performance against a manually compiled gold standard.

While discovering clusters in a completely unsupervised way is useful, in Chapter 5 we introduced constrained Dirichlet Process Mixture Models (CDPMMs) that allow us to incorporate *must-link* and *cannot-link* constraints to adapt the clustering solution towards a user's prior intuition. Thus, applications that rely on these clusters can take users' needs into account. Furthermore, we presented an active learning method for CDPMMs based on uncertainty based sampling, which minimizes the number of links that need to be provided.

Each of these chapters concluded with suggestions for future work that are specific to the particular task explored. In what follows, we outline directions for future work with a broader scope.

An important aspect that was not explored in this thesis is the combination of various techniques presented in an integrated pipeline. The main reason is that there are no integrated resources that would allow us to evaluate these techniques simultaneously. Furthermore, building such a pipeline would distract our focus from the individual tasks, which we believe was very informative since in each of them our investigation brought up evaluation issues. Nevertheless, we argue that the connections between the tasks explored were made clear so that future work could move in that direction.

Future work should also investigate joint learning for the various BioIE tasks. Recent work has demonstrated that improvements can be obtained by learning simultaneously related tasks, such as named entity recognition and syntactic parsing (Finkel and Manning, 2009). Most importantly, the approaches developed should be able to take advantage of existing resources and partial annotation such as those used in this thesis, so that they are applicable to new domains and task definitions. In addition, further research is needed in order

to combine them with active learning and integrate them in a real working environment. Such research is likely to benefit from ideas from machine learning and human computer interaction.

Among the tasks explored, we believe that biomedical event extraction presents the greatest potential for future work. This is due to its complexity which led to relatively low performances as well as its reliance on other IE and NLP components, to which it provides an extrinsic evaluation. Future work should consider extracting events from full papers which, as observed in the experiments in Chapter 2, they are likely to present a different challenge. Furthermore, it would be interesting to explore ways of adapting an event extraction system with user feedback, as performed in Chapters 2 and 5. In this direction, it would be useful to consider the lessons learnt from the recent success of crowd-sourcing methods in obtaining annotated material.

Finally, our work on biomedical verb clustering demonstrated the potential of non-parametric Bayesian techniques that are able to adapt the complexity of the learnt models to the data. Such techniques are likely to become increasingly useful to biomedical information extraction and future work should look into integrating supervision of various forms beyond pairwise constraints that were explored in this thesis.

Bibliography

Eugene Agichtein and Luis Gravano. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, pages 85–94, San Antonio, TX, USA, June 2000. Association for Computing Machinery.

Sophia Ananiadou and John Mcnaught. *Text Mining for Biology And Biomedicine*. Artech House, Inc., Norwood, MA, USA, 2005.

Rie Kubota Ando. BioCreAtIvE II Gene Mention Tagging System at IBM Watson. In *Proceedings of the Second BioCreAtIvE Challenge Evaluation Workshop*, 2007.

David Andrzejewski, Xiaojin Zhu, and Mark Craven. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–32, 2009.

Charles E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

Shlomo Argamon-Engelson and Ido Dagan. Committee-based sample selection for probabilistic classifiers. *Journal of Artificial Intelligence Research*, 11:335–360, 1999.

Jason Baldridge and Miles Osborne. Active learning and the total cost of annotation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 9–16, July 2004.

Sugato Basu, Mikhail Bilenko, Arindam Banerjee, and Raymond J. Mooney. Probabilistic semi-supervised clustering with constraints. In O. Chapelle, B. Schoelkopf, and A. Zien, editors, *Semi-Supervised Learning*, pages 73–102. MIT Press, 2006.

Ann Beis, Mark Ferguson, Karen Katz, and Robert MacIntyre. Bracketing guidelines for treebank II style: Penn TreeBank Project. Technical report, University of Pennsylvania, 1995.

Daniel M. Bikel. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4): 479–511, 2004.

Jari Bjorne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. Extracting complex biological events with rich graph-based feature sets. In *Proceedings of the BioNLP'09 Shared Task on Event Extraction*, pages 10–18, 2009.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 120–128, Sydney, Australia, July 2006. Association for Computational Linguistics.

Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13:422–426, 1970.

Chris Brew and Sabine Schulte im Walde. Spectral Clustering for German Verbs. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 117–124, 2002.

Ted Briscoe and John Carroll. Automatic extraction of subcategorization from corpora. In *Proceedings of the 5th conference on Applied Natural Language Processing*, pages 356–363, 1997.

Ted Briscoe and John Carroll. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1499–1504, 2002.

Ted Briscoe, John Carroll, and Rebecca Watson. The second release of the RASP system. In *Proceedings of the COLING/ACL Interactive presentation sessions*, pages 77–80, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

Markus Bundschus, Mathaeus Dejori, Martin Stetter, Volker Tresp, and Hans-Peter Kriegel. Extraction of semantic biomedical relations from text using conditional random fields. *BMC Bioinformatics*, 9(1):207, 2008.

Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun K. Ramani, and Yuk Wah Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 33(2):139 – 155, 2005.

Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

Ekaterina Buyko, Joachim Wermter, Michael Poprat, and Udo Hahn. Automatically Adapting an NLP Core Engine to the Biology Domain. In *Proceedings of the Joint BioLINK-Bio-Ontologies Meeting.*, pages 65–68, 2006.

Ekaterina Buyko, Erik Faessler, Joachim Wermter, and Udo Hahn. Event extraction from trimmed dependency graphs. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 19–27, June 2009.

Jean Carletta. Assessing Agreement on Classification Tasks: The Kappa Statistic. *Computational Linguistics*, 22(2):249–254, 1996.

Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 173–180, 2005.

Nancy A. Chinchor. Overview of MUC-7. In *Proceedings of the 7th Message Understanding Conference*, 1998.

Andrew Clegg and Adrian Shepherd. Benchmarking natural-language parsers for biological applications using dependency graphs. *BMC Bioinformatics*, 8(1):24+, January 2007.

Kevin B. Cohen, Martha Palmer, and Lawrence Hunter. Nominalization and alternations in biomedical language. *PLoS ONE*, 3(9), 2008.

Shay B. Cohen and Noah A. Smith. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 74–82, 2009.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145, 1996.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. Inducing compact but accurate tree-substitution grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, June 2009.

Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, 1999.

Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

Mark Craven and Johan Kumlien. Constructing biological knowledge-bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86, 1999.

Jeremiah Crim, Ryan McDonald, and Fernando Pereira. Automatically annotating documents with normalized gene lists. *BMC Bioinformatics*, 6 (Suppl 1):S13, 2005.

Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 423–429, 2004.

Aron Culotta, Trausti Kristjansson, Andrew McCallum, and Paul Viola. Corrective feedback and persistent learning for information extraction. *Artificial Intelligence*, 170(14): 1101–1122, 2006.

James R. Curran and Stephen Clark. Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the 11th Annual Meeting of the European Chapter of the Association for Computational Linguistics*, pages 91–98, 2003.

James R. Curran, Stephen Clark, and Johan Bos. Linguistically Motivated Large-Scale NLP with C&C and Boxer. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 33–36, June 2007.

Van B. Dang and Akiko N. Aizawa. Multi-class named entity recognition via bootstrapping with dependency tree-based patterns. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 76–87, 2008.

Hal Daumé III. Unsupervised search-based structured prediction. In *International Conference on Machine Learning*, pages 209–216, 2009.

Marie-Catherine de Marneffe and Christopher D. Manning. The Stanford typed dependencies representation. In *Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8, 2008.

Shipra Dingare, Malvina Nissim, Jenny Finkel, Christopher Manning, and Claire Grover. A system for identifying named entities in biomedical text: how results from two evaluations reflect on both the system and the evaluations: Conference papers. *Computational Functional Genomics*, 6(1-2):77–85, 2005.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. The Automatic Content Extraction (ACE) Program – Tasks, Data, and Evaluation. In *Proceedings of the Fourth Conference on Language Resources and Evaluation*, 2004.

Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st Annual International ACM SIGIR Conference on research and development in Information Retrieval*, pages 595–602, 2008.

Jenny Finkel, Shipra Dingare, Huy Nguyen, Malvina Nissim, Christopher Manning, and Gail Sinclair. Exploiting context for biomedical entity recognition: from syntax to the web. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 88–91, 2004.

Jenny R. Finkel and Christopher D. Manning. Joint parsing and named entity recognition. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 326–334, June 2009.

Arno Fritsch and Katja Ickstadt. Improved criteria for clustering based on the posterior similarity matrix. *Bayesian analysis*, 4(2):191–412, 2009.

Katrin Fundel, Robert Kuffner, and Ralf Zimmer. RelEx–Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371, 2007.

Benjamin C. M. Fung, Ke Wang, and Martin Ester. Hierarchical document clustering using frequent itemsets. In *Proceedings of SIAM International Conference on Data Mining*, pages 59–70, 2003.

Jianfeng Gao and Mark Johnson. A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 344–352, 2008.

Caroline Gasperin. Semi-supervised anaphora resolution in biomedical texts. In *Proceedings of the BioNLP Workshop on Linking Natural Language and Biology*, pages 96–103, June 2006.

Caroline Gasperin, Nikiforos Karamanis, and Ruth Seal. Annotation of anaphoric relations in biomedical full-text articles using a domain-relevant scheme. In *Proceedings of the 6th Discourse Anaphora and Anaphor Resolution Colloquium*, 2007.

Zoubin Ghahramani and Matthew J. Beal. Graphical models and variational methods. In Saad and Opper, editors, *Advanced Mean Field Methods—Theory and Practice*. MIT Press, 2000.

Andrew B. Goldberg, Nathanael Fillmore, David Andrzejewski, Zhiting Xu, Bryan Gibson, and Xiaojin Zhu. May all your wishes come true: A study of wishes and how to recognize them. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 263–271, June 2009.

Ben Hachey, Beatrice Alex, and Markus Becker. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 144–151, June 2005.

Aria Haghighi and Dan Klein. Prototype-driven learning for sequence models. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 320–327, 2006.

Aria Haghighi and Dan Klein. Unsupervised Coreference Resolution in a Nonparametric Bayesian Model. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 848–855, June 2007.

Greg Hamerly and Charles Elkan. Learning the K in K-means. In *Proceedings of the Seventeenth Annual Conference on Neural Information Processing Systems*, pages 281–288, 2003.

Katherine A. Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 297–304, 2005.

Lynette Hirschman, Marc Colosimo, Alexander Morgan, and Alexander Yeh. Overview of BioCreAtIvE task 1B: normalized gene lists. *BMC Bioinformatics*, 6 (Suppl 1):S11, May 2005a.

Lynette Hirschman, Alexander Yeh, Christian Blaschke, and Alfonso Valencia. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6 (Suppl 1):S1, 2005b.

Jerry R. Hobbs. Information extraction from biomedical text. *Journal of Biomedical Informatics*, 35(4):260–264, 2002.

Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

Kevin Humphreys, George Demetriou, and Robert Gaizauskas. Bioinformatics applications of information extraction from scientific journal articles. *Journal of Information Science*, 26(2):75–85, April 2000.

Thorsten Joachims. Making large-scale support vector machine learning practical. In A. Smola B. Schölkopf, C. Burges, editor, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998a.

Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, Lecture Notes in Computer Science, pages 137–142. Springer, April 1998b.

Eric Joanis, Suzanne Stevenson, and David James. A general feature space for automatic verb classification. *Natural Language Engineering*, 14(3):337–367, 2008.

Yoshinobu Kano, William Baumgartner, Luke McCrohon, Sophia Ananiadou, Kevin Cohen, Larry Hunter, and Jun'ichi Tsujii. U-compare: share and compare text mining tools with UIMA. *Bioinformatics*, 25(15):1997–1998, 2009.

Nikiforos Karamanis, Ian Lewin, Ruth Seal, Rachel Drysdale, and Ted Briscoe. Integrating natural language processing with FlyBase curation. In *Proceedings of the Pacific Symposium in Biocomputing*, pages 245–256, 2007.

Nikiforos Karamanis, Ruth Seal, Ian Lewin, Peter McQuilton, Andreas Vlachos, Caroline Gasperin, Rachel Drysdale, and Ted Briscoe. Natural language processing in aid of FlyBase curators. *BMC Bioinformatics*, 9:193, 2008.

S. Sathiya Keerthi and Chih-Jen Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 15(7):1667–1689, 2003.

Halil Kilicoglu and Sabine Bergler. Syntactic dependency based heuristics for biological event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 119–127, June 2009.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. GENIA corpus - a semantically annotated corpus for bio-textmining. In *Bioinformatics*, volume 19, Suppl. 1, pages 180–182, 2003.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier, editors. *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, 2004.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(1):10, 2008.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, June 2009.

Karin Kipper-Schuler. *VerbNet: A broad-coverage, comprehensive verb lexicon.* PhD thesis, University of Pennsylvania, 2005.

Dan Klein, Sepandar D. Kamvar, and Christopher D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 307–314, 2002.

Anna Korhonen. *Subcategorization acqusition.* PhD thesis, University of Cambridge, 2002.

Anna Korhonen, Yuval Krymolowski, and Zvika Marx. Clustering polysemic subcategorization frame distributions semantically. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 64–71, 2003.

Anna Korhonen, Yuval Krymolowski, and Nigel Collier. Automatic classification of verbs in biomedical texts. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 345–352, 2006.

Anna Korhonen, Yuval Krymolowski, and Nigel Collier. The choice of features for classification of verbs in biomedical texts. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 449–456, 2008.

Ioannis Korkontzelos, Andreas Vlachos, and Ian Lewin. From gene names to actual genes. In *Proceedings of BioLink at ISMB*, 2007.

Ioannis Korkontzelos, Ioannis Klapaftis, and Suresh Manandhar. Reviewing and evaluating automatic term recognition techniques. In *Proceedings of the 6th International Conference on Natural Language Processing, GoTAL 2008*, pages 248–259, August 2008.

Martin Krallinger and Lynette Hirschman, editors. *Proceedings of the Second BioCreAtIvE Challenge Evaluation Workshop*, 2007.

Martin Krallinger, Florian Leitner, Carlos Rodriguez-Penagos, and Alfonso Valencia. Overview of the protein-protein interaction annotation extraction task of BioCreative II. *Genome Biology*, 9(Suppl 2):S4, 2008.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of 18th International Conference in Machine Learning*, pages 282–289, 2001.

Matthew Lease and Eugene Charniak. Parsing biomedical literature. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing*, pages 58 – 69, October 2005.

Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In F. Fogelman and P. Gallinari, editors, *Proceedings of the International Conference on Artificial Neural Networks*, pages 53–60, 1995.

Daniel D. Lee and Sebastian H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999.

Beth Levin. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago, 1993.

Ian Lewin. BaseNPs that contain gene names: domain specificity and genericity. In *Proceedings of BioNLP: Biological, translational, and clinical language processing*, pages 163–170, 2007.

David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In W. Bruce Croft and Cornelis J. van Rijsbergen, editors, *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.

Jianguo Li and Chris Brew. Which are the best features for automatic verb classification. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies*, pages 434–442, June 2008.

Percy Liang, Michael I. Jordan, and Dan Klein. Learning from measurements in exponential families. In *Proceedings of the 26th International Conference on Machine Learning*, pages 641–648, 2009.

Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Compuation*, 19(10):2756–2779, 2007.

Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Seventh Conference on Natural Language Learning*, pages 188–191. Edmonton, Canada, 2003.

Andrew McCallum and Ben Wellner. Conditional models of identity uncertainty with application to noun coreference. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Proceedings of the Eighteenth Annual Conference on Neural Information Processing Systems*, pages 905–912, Cambridge, MA, 2004. MIT Press.

Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

David McClosky and Eugene Charniak. Self-training for biomedical parsing. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies*, pages 101–104, 2008.

Marina Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007.

Guido Minnen, John Carroll, and Darren Pearce. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223, 2001.

Yusuke Miyao, Rune Sætre, Kenji Sagae, Takuya Matsuzaki, and Jun'ichi Tsujii. Task-oriented evaluation of syntactic parsers and their representations. In *Proceedings of the*

46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies, pages 46–54, June 2008.

Alexander A. Morgan, Lynette Hirschman, Marc E. Colosimo, Alexander S. Yeh, and Jeffrey B. Colombe. Gene name identification and normalization using a model organism database. *Journal of Biomedical Informatics*, 37(6):396–410, 2004.

Daniel J. Navarro, Thomas L. Griffiths, Mark Steyvers, and Michael D. Lee. Modeling individual differences using Dirichlet processes. *Journal of Mathematical Psychology*, 50(2):101–122, April 2006.

Radford M. Neal. Probabilistic Inference Using Markov Chain Monte Carlo Methods. Technical report, University of Toronto, Department of Computer Science, September 1993.

Radford M. Neal. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, June 2000.

Claire Nédellec. Learning Language in Logic - Genic Interaction Extraction Challenge. In *Proceedings of the Learning Language in Logic 2005 Workshop at ICML*, 2005.

Goran Nenadic, Sophia Ananiadou, and John McNaught. Enhancing automatic term recognition through recognition of variation. In *COLING '04: Proceedings of the 20th international conference on Computational Linguistics*, pages 604–610, 2004.

Vincent Ng. Semantic class induction and coreference resolution. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 536–543, June 2007.

Tomoko Ohta, Jin-Dong Kim, Sampo Pyysalo, Yue Wang, and Jun'ichi Tsujii. Incorporating GENETAG-style annotation to GENIA corpus. In *Proceedings of the BioNLP 2009 Workshop*, pages 106–107, June 2009.

Fernando Pereira, Naftali Tishby, and Lillian Lee. Distributional clustering of English words. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 183–190, June 1993.

John Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

Detlef Prescher, Stefan Riezler, and Mats Rooth. Using a probabilistic class-based lexicon for lexical ambiguity resolution. In *In Proceedings of the 18th International Conference on Computational Linguistics*, pages 649–655, 2000.

Jan Puzicha, Thomas Hofmann, and Joachim Buhmann. A theory of proximity based clustering: Structure detection by optimization. *Pattern Recognition*, 33(4):617–634, 2000.

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Bjorne, Jorma Boberg, Jouni Jarvinen, and Tapio Salakoski. BioInfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(1):50+, 2007.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected apllications in speech recognition. In A. Waibel and K.-F. Lee, editors, *Readings in Speech Recognition*, pages 267–296. Kaufmann, San Mateo, CA, 1990.

William M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

Jason D. M. Rennie and Ryan Rifkin. Improving multiclass text classification with the Support Vector Machine. Technical Report AIM-2001-026, Massachusetts Insititute of Technology, Artificial Intelligence Laboratory, 2001.

Sebastian Riedel. Improving the Accuracy and Efficiency of MAP Inference for Markov Logic. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 468–475, 2008.

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. A Markov Logic Approach to Bio-Molecular Event Extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 41–49, 2009.

Laura Rimell and Stephen Clark. Porting a lexicalized-grammar parser to the biomedical domain. *Journal of Biomedical Informatics*, 42(5), October 2009.

Angus Roberts, Robert Gaizauskas, Mark Hepple, and Yikun Guo. Mining clinical relationships from patient narratives. *BMC Bioinformatics*, 9(Suppl 11):S3, 2008.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420, June 2007.

Kenji Sagae and Jun'ichi Tsujii. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 1044–1050, 2007.

Guido. Sanguinetti, Jonathan Laidler, and Neil D. Lawrence. Automatic determination of the number of clusters using spectral algorithms. In *IEEE Workshop on Machine Learning for Signal Processing*, 2005.

Jonathan Schuman and Sabine Bergler. Postnominal prepositional phrase attachment in proteomics. In *Proceedings of the Workshop on Linking Natural Language Processing and Biology*, pages 82–89, 2006.

Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4: 639–650, 1994.

B. Settles and M. Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1069–1078, 2008.

Burr Settles. Biomedical Named Entity Recognition Using Conditional Random Fields and Novel Feature Sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, 2004.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual ACM Conference on Computational Learning Theory*, pages 287–294, July 1992.

Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 134–141, 2003.

Christian Siefkes. A comparison of tagging strategies for statistical information extraction. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 149–152, June 2006.

Larry H. Smith and W. John Wilbur. The value of parsing as feature generation for gene mention recognition. *Journal of Biomedical Informatics*, 42(5):895 – 904, 2009.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. Cheap and Fast – But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 254–263, Honolulu, Hawaii, October 2008.

Mark Stevenson, Yikun Guo, Robert Gaizauskas, and David Martinez. Disambiguation of biomedical text using diverse sources of information. *BMC Bioinformatics*, 9(Suppl 11):S7, 2008.

Beth M. Sundheim. Overview of results of the MUC-6 evaluation. In *MUC6 '95: Proceedings of the 6th conference on Message understanding*, pages 13–31, 1995.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006.

Robert S. Swier and Suzanne Stevenson. Unsupervised semantic role labelling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 95–102, 2004.

Yuka Tateisi and Jun'ichi Tsujii. Part-of-speech annotation of biology research abstracts. In *Proceedings of the the 4th International Conference on Language Resources*, pages 1267–1270, 2004.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. Syntax annotation for the GENIA corpus. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing, Companion Volume*, pages 222–227, October 2005.

Yee Whye Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992, Sydney, Australia, July 2006.

Yee Whye Teh, Hal Daumé III, and Daniel Roy. Bayesian agglomerative clustering with coalescents. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, 2007.

J. Thomas, D. Milward, C. Ouzounis, and S. Pulman. Automatic extraction of protein interactions from scientific abstracts. In *Proceedings of the Pacific Symposium in Biocomputing*, pages 538–549, 2000.

Naftali Tishby, Fernando Pereira, and William Bialek. The information bottleneck method. In *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

Erik F. Tjong Kim Sang. Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition. In Dan Roth and Antal van den Bosch, editors, *Proceedings of the Sixth Workshop on Computational Language Learning*, pages 155–158, 2002.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of the Seventh Conference on Computational Natural Language Learning*, pages 142–147. Edmonton, Canada, 2003.

Katrin Tomanek and Udo Hahn. Semi-supervised active learning for sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th*

International Joint Conference on Natural Language Processing of the AFNLP, pages 1039–1047, August 2009.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2002.

Özlem Uzuner, Tawanda C. Sibanda, Yuan Luo, and Peter Szolovits. A de-identifier for medical discharge summaries. *Artificial Intelligence in Medicine*, 42(1):13–35, 2008.

Jurgen Van Gael, Andreas Vlachos, and Zoubin Ghahramani. The infinite HMM for unsupervised PoS tagging. In *Proceedings of 2009 Conference on Empirical Methods in Natural Language Processing*, pages 678–687, Singapore, 2009.

Keith Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition edition, 1979.

Vladimir Vapnik. *The Nature of Statistical Learning Theory*. Springer, N.Y., 1995.

Andreas Vlachos. Tackling the BioCreAtIvE Gene Mention task with Conditional Random Fields and Syntactic Parsing. In *Proceedings of the Second BioCreative Challenge Evaluation Workshop*, 2007.

Andreas Vlachos, Paula Buttery, Diarmuid Ó Séaghdha, and Ted Briscoe. Biomedical event extraction without training data. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 37–40, June 2009.

Ulrike von Luxburg. A tutorial on spectral clustering. Technical Report TR-149, Max Plank Institute for Biological Cybernetics, August 2006.

Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–1110, 2000.

Xing Wei and W. Bruce Croft. LDA-based document models for ad-hoc retrieval. In *Proceedings of the 29th ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006.

Ben Wellner. Weakly supervised learning methods for improving the quality of gene name normalization data. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases, Mining Biological Semantics*, pages 1–8, 2005.

Joachim Wermter and Udo Hahn. Massive biomedical term discovery. In *Discovery Science*, pages 281–293, 2005.

Wei Xu, Xin Liu, and Yihong Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 267–273, 2003.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106, 2003.

Ying Zhao and George Karypis. Criterion functions for document clustering: Experiments and analysis. Technical Report TR 01-40, Department of Computer Science, University of Minnesota, 2001.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining Active Learning and Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *ICML workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 58–65, 2003.

Xiaojin Zhu and Andrew B. Goldberg. *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, 2009.