Coordination Resolution in Biomedical Texts

by

Philip Victor Ogren

B.S., Harding University, 1997

M.S., University of Colorado at Boulder, 2005

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2011

This thesis entitled:
Coordination Resolution in Biomedical Texts
written by Philip Victor Ogren
has been approved for the Department of Computer Science

_____

Prof. Lawrence Hunter

_____

Prof. James Martin

_____

Prof. Martha Palmer

_____

Rodney Nielsen, Ph.D.

_____

Prof. Wayne Ward

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Ogren, Philip Victor (Ph.D., Computer Science)

Coordination Resolution in Biomedical Texts

Thesis directed by Prof. Prof. Lawrence Hunter

One of the most difficult and least studied sources of structural ambiguity in text is syntactic coordination. Coordination resolution, for this dissertation, is the task of determining the correct conjuncts of the coordinating conjunctions "and" and "or" and is explored here for biomedical scientific literature. It is a challenging problem because conjunctions are highly promiscuous with respect to the kinds of words and phrases that they are willing to coordinate. For example, a conjunct may consist of a single word such as an adjective or a much longer verb phrase. The main contribution of this work is an efficient and accurate coordination resolution algorithm that outperforms the previous state-of-the-art on this task and a state-of-the-art syntactic parser when applied to this task. The algorithm uses binary classifiers to predict conjunct boundaries. One of the more interesting features that improved the performance of these classifiers leverages probabilities generated by a language model which is built using large quantities of readily available unlabeled data. The language model derived features exploit the intuition that sentences containing coordinating conjunctions can often be rephrased as two or more smaller sentences derived from the coordination structure. Candidate sentences corresponding to different possible coordination structures are generated and compared using the language model to help determine which coordination structure is best. Performance is further improved by first predicting the syntactic type of the coordination structure and using this type prediction to help train and classify conjunct boundaries. Finally, a system that integrates the new approach with a syntactic parser is shown to outperform either approach in isolation.

Dedication

To Jana, Marlow, Jonah, and Rowan.

Acknowledgements

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

One of the most difficult and least studied sources of structural ambiguity in biomedical text is syntactic coordination. Coordination in this dissertation will refer to the coordinating conjunctions ***and*** and ***or*** and the conjuncts that they coordinate. Coordination resolution is the task of identifying the correct conjuncts of one of these two coordinating conjunctions. The following sentence contains the coordinating conjunction ***and*** twice:

- Examination of [*embryonic* **and** *adult*] [*germ cells* **and** *gonads*] in Dppa3-deficient animals did not reveal any defects.

The two conjunctions in this example sentence are shown in bold font while their corresponding conjuncts are shown in italics. Biologists have little difficulty reading and understanding this sentence because they would know that germ cells and gonads can both be modified by "embryonic" and "adult" and so the most likely interpretation is that the following subjects of study were examined: embryonic germ cells, adult germ cells, embryonic gonads, and adult gonads. Machines, in contrast, have difficulty performing coordination resolution on examples like these because they are confronted with a large number of possible conjuncts, many of which look structurally similar from a syntactic point of view. Table 1.1 shows all of the possible conjuncts for the second conjunction found in the example sentence above. In total, there are seven possible left conjuncts and nine possible right conjuncts which gives a total of sixty-three possible coordination structures for that conjunction. Such a large number of candidate coordination structures is quite common and

represents a search space through which a machine must effectively navigate to perform well on this task.

Table 1.1: A listing of all possible conjuncts for the second conjunction in the sentence "Examination of embryonic and adult germ cells and gonads in Dppa3-deficient animals did not reveal any defects." An asterisk (*) indicates a correct conjunct.

Conjunct	Possible Values
left	cells
left	germ cells*
left	adult germ cells
left	and adult germ cells
left	embryonic and adult germ cells
left	of embryonic and adult germ cells
left	Examination of embryonic and adult germ cells
right	gonads*
right	gonads in
right	gonads in Dppa3-deficient
right	gonads in Dppa3-deficient animals
right	gonads in Dppa3-deficient animals did
right	gonads in Dppa3-deficient animals did not
right	gonads in Dppa3-deficient animals did not reveal
right	gonads in Dppa3-deficient animals did not reveal any
right	gonads in Dppa3-deficient animals did not reveal any defects

Automatic coordination resolution is an interesting challenge because there exists a large disparity between how well modern syntactic parsers and humans perform on this task. This is particularly unfortunate in the biomedical domain because coordinating conjunctions are very common in biomedical text. In fact, Tateisi et al.[51] found that coordinating conjunctions occur nearly twice as often in biomedical abstracts as in newswire text. The prevalence of coordinating conjunctions in biomedical text presents a challenge to syntactic parsers and information extractions systems which require successful coordination resolution in order to perform well.

This dissertation focuses on automatic coordination resolution in biomedical texts. Two approaches are compared: a syntactic parser based approach and a "parser-free" approach which performs coordination resolution as an isolated task from syntactic parsing. The parser-free approach is shown to outperform the parser-based approach on several corpora at a much faster speed.

The two approaches are shown to be complementary and when integrated produce the highest performing coordination resolution system. The main contribution of this work is the presentation of a simple algorithm that the parser-free approach employs and the use of novel features that exploit language models built from large quantities of readily available unlabeled data. A secondary contribution of this work is a new corpus that consists of coordination structures for 5,000 randomly selected sentences from the biomedical domain and inter-annotator agreement numbers for the test set of the corpus. Also, a method for deriving coordination structures from the output of a syntactic parser or Penn Treebank style annotations is presented.

The remainder of this chapter presents related work. Chapter 2 describes the corpora that contain gold-standard coordination structures used for training and evaluating coordination resolution systems. Chapter 3 details features used by a machine learning classifier for the coordination resolution system described in Chapters 4 and 5. Chapters 4 and 5 introduce the novel "parser-free" coordination resolution system and compare its performance with a state-of-the-art syntactic parser on this task. Chapter 6 presents results of the coordination resolution system on three different corpora and provides detailed error analysis. Chapter 7 provides a summary and conclusions of this work and discusses future research directions for coordination resolution.

1.1 Related Work

There have been two main approaches for performing automatic coordination resolution. One approach considers coordination resolution within the broader task of syntactic parsing. In this approach evaluation of coordination resolution per se is seldom done. Rather, the evaluation criteria of interest is usually the overall syntactic parsing performance and the effect of coordination-specific considerations on it. The other main approach considers coordination resolution as an isolated task in which a system is built specifically to perform coordination resolution. In this approach, coordination resolution is typically narrowly defined and evaluation is conducted on small data sets. For the sake of brevity I have excluded a literature review of work done in the early 1990's, the 1980's, and earlier. Suffice it to say that the papers are few in number, are heavily influenced

by linguistic formalisms, describe systems that can generally be characterized as heuristic-based search, and report results on a small number of examples. One interesting anecdote about several of the papers reviewed in this chapter is that they emphasize in their introductions how little studied coordination resolution is despite the fact that it is a major source of system errors.

1.1.1 Coordination Resolution in the Context of Syntactic Parsing

Any syntactic parser that produces a full syntactic analysis of a sentence must perform coordination resolution as a part of that syntactic analysis because coordination resolution is typically considered primarily a syntactic phenomena. Typically, a syntactic parser will have a single, central algorithm that is used to determine all syntactic categories and relationships contained in a sentence. However, this does not preclude parsers from giving special attention to coordination resolution by adding coordination resolution specific rules and features.

Nivre and McDonald[37] show that the MSTParser[32] and MaltParser[35] perform worse on dependencies related to conjunctions (both coordinating and subordinating) than for most other parts-of-speech by a wide margin (as much as 10% worse compared to verbs and nouns.) To address this disparity Nilsson et al.[34] show that giving special attention to the structure of coordination dependencies improves overall parsing performance on the Prague Dependency Treebank (PDT). The PDT makes conjunctions the head of their conjuncts. Nilsson et al. automatically transform the training data from this structure to one that chains coordination dependencies from the first conjunct and then train on this transformed data. The resulting parse of this model is transformed back into the structure used by the PDT to provide an overall performance gain when evaluated on the test data in its original form.

The constituent parser described in Charniak and Johnson [9] is a reranking parser that uses two features which focus on coordination when mapping each generated parse to a set of features. The first measures parallelism in the labels of conjunct constituents and their children and the second measures the lengths of the conjunct constituents. However, they do not report what effect these two features have on overall parsing performance or on coordination resolution. The work

done by Hogan[21] focuses directly on coordination of noun phrases in the context of the Collins parser[13]. After the parser has determined the structure of the first conjunct, features derived from that conjunct are used when building the structure of the second conjunct. The overall results of this work resulted in an increase of performance of 69.9% to 73.8% in F-score (a 13% error reduction) for coordinate noun phrases. This resulted in a small but statistically significant performance gain for overall parser performance from 89.4% to 89.6%. The parser-free coordination resolution system introduced in Section §5.3 similarly uses features from existing conjuncts to help determine the remaining conjuncts.

McClosky et al.[31] report improvements in syntactic parsing performance using self-training on unlabeled data with their two-phase reranking parser. One of the surprising results from this study is that sentences in the test set for which parsing performance improved as a result of the self-training were more likely to have conjunctions than sentences for which there was no improvement, i.e. the self-training improved coordination resolution of the parser. The authors point out that they had expected improvement to correspond to the presence of unknown words or prepositional phrases and were reluctant to make conclusions about this counter-intuitive result. They conjecture that the self-training helps the parser learn what sentences and verb phrases look like which helps for parsing conjunctions that coordinate these two constituent types. One of the telling quotes from this paper is "Conjunctions are about the hardest things in parsing, and we have no grip on exactly what it takes to help parse them."

1.1.2 Coordination Resolution as an Isolated Task

An example of coordination resolution approached as an isolated task is that of Chantree et al.[8] who make use of word distribution information to disambiguate modifiers involved with coordination. They focus on conjunctions that have only two conjuncts (typically two noun phrases) and a single ambiguous modifier (typically a noun, adjective, or preposition) as in the phrase "old boots and shoes". Their system judged such phrases to be coordination-first (i.e. "old" modifies "boots and shoes"), coordination-last (i.e. the conjuncts are "old boots" and "shoes"), or "ambigu-

ous so that it might lead to misunderstanding." They collected 138 such coordination examples from requirements specifications documents that were syntactically ambiguous and had them reviewed by human judges who rated them as one of the three categories. Their system makes use of three types of word distribution information derived from the British National Corpus: "relative frequency of the coordination ..., the distributional similarity of the coordinated words, and the collocation frequency between the coordinated words and a modifier." Each of these were used effectively by their system to produce a significant performance gain.

In related work, Nakov and Hearst[33] use the web to disambiguate coordinations of the form "noun conjunction noun head-noun". Their task corresponds directly to the coordination-first and coordination-last classification described above. They leveraged web data to create paraphrase features and surface features that involve use of punctuation. The surface features using punctuation had little effect, but using paraphrase features was found to be effective. For example, the phrase "car and truck production" would more likely be classified as coordination-first if there are many paraphrases from the web such as "truck and car production" and "car production and truck production." Paraphrases such as "truck production and car" would push the classifier towards a coordination-last classification. Their overall accuracy improved from a baseline (always choose coordination-first) of 56.5% to the best system which used, among other things, paraphrase features and achieved 80.6% accuracy. In section §3.2, language model features that echo the paraphrase strategies used here and the word distribution information used in [8] described above are introduced.

In another study, Resnik[44] makes use of semantic similarity using WordNet for resolving the ambiguity found in coordinating conjunctions with the patterns "noun conjunction noun noun" and "noun noun conjunction noun noun." Instead of measuring semantic similarity between two concepts using a distance metric based on taxonomical structure, similarity is measured using the information content of the "most informative subsumer" of two concepts. This measure relies heavily on the probabilities associated with each concept in the taxonomy of seeing a mention of a concept in text. While this similarity measure is theoretically elegant this particular study is crippled by the

very small amount of data in the training and test sets (200 and 178 phrases, respectively, for the two patterns examined) and the extent to which WordNet provided concept-recognition coverage for words in the data sets. Even still, there were encouraging results that indicated that the use of semantic similarity shows promise for coordination resolution. In section §4.2.3, the use of related semantic similarity measures for coordination resolution is explored.

An interesting observation from the work summarized in this subsection is that the authors do not attempt to compare their results with previous results other than in the broadest and most-qualified way. This is because, according to Resnik[44], there is no common data set or task definition for comparative evaluation. A decade later, this remains true for coordination resolution. In this work, I provide a data set that can be shared and a coordination resolution task definition along with evaluation metrics that will hopefully promote further research that is directly comparable to what is presented in this dissertation.

1.1.3 Biomedical Coordination Resolution in the Context of Syntactic Parsing

Very little attention has been given to coordination resolution in the context of syntactic parsing of biomedical texts. One study performed by Clegg and Shepherd [10] looked at the performance of two constituent-based syntactic parsers on coordination resolution evaluated on GENIA treebank data[47]: the Bikel parser (version 0.9.8)[4] and the Charniak-Lease parser[26]. They converted the GENIA treebank data and the output of these parsers into dependency graphs using the Stanford Parser's constituent-to-dependency conversion tool which is based on the work of Marneffe et al.[29]. Coordination resolution was evaluated as F-score by evaluating the dependencies in the subgraphs of the dependency trees whose roots are at either end of a conjunction dependency. They argue that simply measuring accuracy of the dependencies corresponding to conjunctions and conjuncts) does not take into account the overall effect of errors that are caused by erroneous conjunction and conjunct dependencies. By measuring performance on these subgraphs they are attempting to show the overall effect of error propagation caused by mistakes on the conjunction dependencies. The two parsers performed at 75.5 and 75.0 F-score, respectively, on the conjunction

subgraphs compared with an overall performance of 77.0 on all dependencies (including the conjunction subgraphs.) While the the performance on the conjunction subgraphs is not dramatically lower than overall performance it does indicate that coordination structures are more difficult to parse. However, the rather low over-all performance of the dependency graphs generated from the output of these parsers may reflect underlying fundamental problems with this study. For example, both parsers were trained on newswire text rather than on biomedical text. Furthermore, the process of converting constituent-based trees to dependency-based trees is error prone and so the true performance of the parsers may not be well reflected in the dependency-based evaluation.

1.1.4 Biomedical Coordination Resolution as an Isolated Task

Buyko et al.[5] study the large number of named entities in the GENIA corpus that are coordinated in such a way that one of the named entities has elided material. For example, the words *dependent transcription* are elided from the named entity *NFAT-dependent transcription* in the phrase *NFAT- and interleukin-2-dependent transcription.* They define an elliptical entity expression (EEE) to be those named entities in GENIA in which such coordinations are explicitly captured. The task they performed was to identify the correct conjuncts for a given EEE and then correctly identify the locations of ellipses and their corresponding antecedents (i.e. the ellided words.) For the EEE given above, the correct conjuncts would be *NFAT-* and *interleukin-2-* with an ellipsis occurring between *NFAT-* and *and* whose corresponding antecedent is *dependent transcription.* The training and test sets were drawn from the 1,578 EEEs found in GENIA. These EEEs account for 3.5% of the named entities annotated in GENIA. The F-score for finding the correct conjuncts was 0.93 using a linear-chain Conditional Random Field. The accuracy for overall EEE resolution (getting the correct conjuncts and identifying the location of the ellipses and the antecedents that satisfy them) was 86%. This very high performance may be due to a task definition that is unrealistically easy because gold-standard EEEs are provided. That is, the system is given an EEE such as *NFAT- and interleukin-2-dependent transcription* and what remains to be done is determine the conjuncts and the antecedents. Unfortunately, their system's ability to identify EEEs

automatically was a disappointing 0.55 as F-score (personal communication.) One interesting result of this study is that the performance numbers suggest that resolution of ellipses can be done with very high accuracy if the conjuncts are correctly identified. To illustrate, a naïve interpretation of their results is to assume that conjunct errors are evenly distributed throughout the EEEs and that there are two conjuncts per EEE (a conservative estimate). Using the conjunct-level performance of 0.93, the overall performance of EEE resolution would be approximately $.93^2 = .865$ if the ellipsis resolution performance is 100%. This result is very close to their reported overall accuracy of 86%. This dissertation does not address recovering elided materials found in coordination structures. However, this result suggests that it can be done with high accuracy.

Shimbo and Hara[45] apply several algorithms to the task of conjunct bracketing and coordination bracketing. They look at sentences that have one instance of the word *and* that is coordinating noun phrases. Only one instance of a conjunction is allowed because their system does not handle nested coordination structures and they assume an oracle that provides the systems with only sentences that contain coordinated noun phrases and gold-standard part-of-speech tags. They apply several baselines to the task for comparison with their own algorithm: the Bikel parser version 0.9.9c, the Bikel parser using the Collins (Bikel/Collins) parser emulation mode, Charniak and Johnson's (CJ) reranking parser, and a chunker based on linear-chain Conditional Random Fields (CRF) using a Begin-Inside-Outside (BIO) labeling scheme. Their algorithm is based on sequence alignment modified for coordination resolution in which an edit graph is employed to assign the highest score to a set of edit operations that translates one conjunct into another. This approach exploits the intuition that coordinated conjuncts often exhibit structural similarities.

In a follow-up study, Hara et al.[18] provide a much more comprehensive treatment to the problem of coordination resolution on biomedical texts. The work by Hara et al. is the most similar to the work presented in this dissertation and a direct comparison between their work and the approaches presented here is reported in detail in Section §6.3. Their approach involves a grammar tailored for coordination structures that is coupled with a sequence alignment algorithm

that uses perceptrons for learning feature weights of an edit graph. Again, this approach exploits the intuition that coordinated conjuncts often exhibit structural similarities. In contrast to the Shimbo and Hara study, here coordinations are not confined to be non-nested coordinated noun-phrases but rather all instances of the conjunctions *and* and *or* and the disjunction *but* are handled by their system regardless of the types of constituents that are coordinated or whether or not the coordination structure is nested inside another coordination structure. Similarly, in this work all coordination structures involving the conjunctions *and* and *or* are analyzed.

1.2 UIMA and ClearTK

All of the software created to support the research performed in this dissertation was built on top of the Unstructured Information Management Architecture (UIMA)[15]. Briefly, UIMA provides a set of interfaces for defining components for analyzing unstructured information and provides infrastructure for creating, configuring, running, debugging, and visualizing these components. In the context of natural language processing (NLP), the focus is on UIMA's ability to process textual data. All UIMA components are organized around a type system which defines the structure of the analysis data associated with documents.[1] This information is instantiated in a data structure called the Common Analysis Structure (CAS). There is one CAS per document that UIMA components can access and update. I chose UIMA for a number of reasons including its open source license, wide spread community adoption, strong developer community, elegant APIs that encourage reusability and interoperability, helpful development tools, and extensive documentation.

While UIMA provides a solid foundation for processing text, it does not directly support machine learning based NLP. ClearTK[39, 40] aims to fill this gap. ClearTK provides a framework for creating UIMA components that use statistical learning as a foundation for decision making and annotation creation. This framework provides a flexible and extensible feature extraction library and wrappers for several popular machine learning libraries. One of the goals of the ClearTK framework is to facilitate the creation of NLP components that make use of classifiers in such

[1] The term *document* is used to refer to any unit of text that is analyzed

a way that the details particular to a specific machine learning classifier library are abstracted away. This promotes best-of-breed solutions in which multiple classifier libraries can be directly compared when used for the same component. As an example, results for a parser-free coordination resolution approach, which is described in Section §5.3 and was built using ClearTK, are reported in Table 5.3 for five different classifier libraries. Additionally, ClearTK provides a number of useful components that were used in this dissertation including a sentence segmenter, tokenizer, stemmer, and part-of-speech tagger.

Chapter 2

Corpora

To train and evaluate the coordination resolution approaches described in subsequent chapters, gold-standard coordination structures are required. Each coordination structure should consist of a conjunction and the conjuncts it coordinates. Optionally, a coordination type may also be assigned to a coordination structure which corresponds, roughly, to the syntactic categories of the conjuncts that are coordinated. Such structures can be obtained from texts annotated with syntactic structures. Two corpora annotated with the Penn Treebank (PTB) format[2] are used in this dissertation: CRAFT and GENIA. Coordination structures can also be directly annotated with high consistency without annotating a full syntactic analysis of each sentence. A new corpus annotated in this way called BIOCC is introduced.

Because coordinating conjunctions are very promiscuous with respect to the constituents, phrases, or words they are willing to coordinate, coordination structures are very diverse. In addition to describing the corpora for the purposes of explaining the experimental setup for training and evaluation used in subsequent chapters, this chapter highlights the diversity of coordination structures by providing many examples and by quantifying differences across several characteristics. Coordination structures are shown to differ with respect to the number of conjuncts, length of conjuncts, and type. These data along with the provided examples underscore the difficulty of accurately performing coordination resolution. However, the results reported in Section §2.2.3 show that humans, despite the complexity of the task, can perform coordination resolution with high agreement.

Basic counts for CRAFT, BIOCC, and GENIA are shown in Table 2.1 along with a subset of the GENIA corpus referred to here as the Hara corpus which is described in Section §2.3. It is interesting to note that the total number of conjunctions in CRAFT and GENIA is 23,300 which is more than the PTB[28] which has 22,888 conjunctions even though the 31,014 sentences in CRAFT and GENIA is far less than the 49,208 sentences in the PTB. It is not surprising then that the percentage of sentences that contain a coordinating conjunction is much higher for CRAFT (49.7%) and GENIA (54.8%) than for PTB (36.6%). Similarly, the ratio of conjunctions to tokens is much higher for the biomedical corpora (2.86% and 2.93% for CRAFT and GENIA, respectively) than for the PTB (1.95%.) These facts suggest that coordination resolution may be more important for biomedical texts than for newswire texts simply because coordinating conjuncts occur more frequently in the former. Incidentally, the word "and" is the third most frequent non-punctuation token in the full-text open-access articles used in Section §2.2.1 behind the words "the" and "of". The word "or" ranks nineteenth.

Table 2.1: Summary of corpora used in this dissertation. The columns labeled 'Sent.' and 'Conj.' correspond to the total number of sentences and conjunctions, respectively. The number of conjunctions equals the number of coordination structures. The column labeled 'S%' is the percentage of sentences that have at least one coordinating conjunction. The column labeled 'T%' is the ratio of conjunction counts to token counts as a percentage. The values for the last two columns are not shown for BIOCC because sentences not containing conjunctions were excluded from BIOCC.

Corpus	Genre	Sent.	Tokens	Conj.	S%	T%
CRAFT	full-text articles	12,473	316,294	9,045	49.7%	2.86%
BIOCC	sentences	5,000	156,746	7,384		
GENIA	abstracts	18,541	486,489	14,255	54.8%	2.93%
Hara	abstracts	4,529	123,170	3,598	55.4%	2.92%
PTB	newswire	49,208	1,173,766	22,888	36.6%	1.95%

2.1 CRAFT

The Colorado Richly Annotated Full-Text (CRAFT) corpus is being developed at the University of Colorado Denver. The corpus consists of ninety-seven full-text open-access scientific articles

that have been annotated by the Mouse Genome Institute[1] with concepts from the Gene Ontology[2] and Mammalian Phenotype Ontology[3] . Each article is annotated with PTB-style syntactic trees, biomedical named entities, and coreference chains. At the time this research was conducted forty-six articles had been syntactically annotated. Thus, the CRAFT corpus consists of only these forty-six articles for the purposes of this dissertation.

2.1.1 CRAFT Training and Test Sets

The CRAFT corpus was split into a training set consisting of thirty-six articles and a test set consisting of the remaining ten articles. The training set was further split into ten folds consisting of three or four articles each. These folds were used for cross-validation during development of the coordination resolution approaches described in subsequent chapters. Sentences within a given article may contain repeated words, phrases, or sentence fragments with other sentences within the same article at a higher rate than with sentences from a different article. This follows from the intuition that the discourse structure of a typical scientific article lends itself to repetition in order to emphasize crucial facts or observations or simply because the objects of study (e.g. specific genes) appear in many sentences. If this is true, then it follows that the coordination structures within an article are more similar to each other than with coordination structures in other articles. For this reason, it is important to avoid training and testing a coordination resolution system on different portions of the same article because this may inflate performance results. This is why the training and test sets and the folds within the training set are delimited at the article level.

One problem with delimiting training and test sets at the article level is that it removes any hope that the conjunctions within the two sets are independent and identically distributed (i.i.d.) In fact, under the assumption that coordination structures within the same article are more similar than those that are not, grouping the training and test sets (as well as the folds) at the article-level guarantees that the coordination structures are not i.i.d. across both sets. For this

[1] http://www.informatics.jax.org/
[2] http://geneontology.org/
[3] http://www.informatics.jax.org/searches/MP_form.shtml

reason, it is not safe to assume that a statistically significant difference between the performance of two coordination resolution systems when evaluated on the training set will hold when the same two systems are evaluated against the test set. Because most significance tests assume that the data are i.i.d. it is problematic to report statistical significance for results reported on the CRAFT corpus. As a proxy for statistical significance, performance results reported on the training set (via cross-validation) or the test set are given with the standard deviation of the performance of the individual articles within the set.

2.1.2 Coordination Structure Production Algorithm

One might hope that given a PTB-style syntactic tree for a sentence, that it would be straightforward to determine the coordination structures found in that sentence. For example, consider the following noun phrase in CRAFT marked up with PTB-style syntactic structure:

```
(NP (UCP (NN brain) (CC and) (JJ striatal)) (NN volume))
```

A simple algorithm that will recover the correct coordination structure for the conjunction represented by the node "(CC and)" is to identify the sibling nodes "(NN brain)" and "(JJ striatal)" and choose the corresponding texts for these two nodes "brain" and "striatal" as the conjuncts. Unfortunately, this simple approach is incorrect in many cases. For example, consider the following verb phrase and noun phrase:

```
(VP (VB map) (CC and) (VB characterize)
  (NP (NP (NNS members))
  (PP (IN of) (NP (NP (DT the) (NN subset))))))

(NP (NP (DT the) (NN production) (CC and) (NN survival))
  (PP (IN of) (NP (JJ striatal) (NNS neurons))))
```

The coordination structure produced by simply choosing siblings of the conjunction node would produce the following conjuncts for the conjunction in the verb phrase: "map", "characterize",

and “members of the subset”. This is a poor choice of conjuncts compared with choosing only “map” and “characterize”. Similarly, the conjuncts for the conjunction node in the noun phrase would be “the”, “production”, and “survival” which is a poor choice compared with choosing only “production” and “survival”. Fortunately, both of these examples can be handled by adding simple rules to the proposed approach. Unfortunately, these represent some of the easier cases to handle. An example of a constituent that is a bit trickier is the following list node:

```
(LST (-LRB- -LRB-)
  (LS B) (, ,) (LS E) (, ,) (LS H) (, ,) (CC and) (LS K)
  (-RRB- -RRB-))
```

For this example, the desired conjuncts are “B”, “E”, “H”, and “K”. To obtain these conjuncts, two nodes corresponding to parentheses and three nodes corresponding to commas must be pruned.

A coordination structure production algorithm (CSPA) for producing coordination structures from syntactic parse trees that handles both easy and difficult cases was developed as follows. First a conjunction node such as “(CC and)” or “(CC or)” or a conjunction phrase such as “(CONJP (CC and) (RB so))” is identified with priority given to the latter (i.e. conjunction nodes nested inside a conjunction phrase are not considered separately.) Next, the parent of the conjunction node (or conjunction phrase) is obtained by traversing one node up the tree. The type of the parent node is used as the type of the resulting coordination structure. Next, the children of the parent node are considered as conjunct candidates. The candidates are then subjected to a series of rules which determine the correct conjuncts by either removing and/or merging candidate nodes. Once the conjuncts are identified they are added to the coordination structure and it is complete.

The rules for determining the correct conjuncts were developed in parallel with a suite of unit tests in an iterative fashion. First, a number of unit tests were created to make sure that easy examples such as the first example given above are handled correctly. Next, the CRAFT corpus was passed through the CSPA with the current set of rules (beginning on the first iteration with no rules.) The results were used to produce an HTML rendering of the training corpus which was

then manually examined. In cases where the coordination structure appeared to be incorrect, the corresponding syntactic structure was examined and a new rule was added to the algorithm (if possible). Coordination structures were evaluated as correct or incorrect according the the manual annotation guidelines discussed in Section §2.2.2. New unit tests were then added to the test suite to ensure that the updated CSPA worked correctly and continued to work correctly as the set of rules expanded for the examples tested. The CRAFT corpus was passed through the updated CSPA and the process was repeated until no more fixable errors were found. All rules that were added address a class of phenomena rather than one specific coordination structure, i.e. there were no rules added that were specific to a single coordination structure instance. In some cases, coordination structures that were incorrect were not fixable by adding a rule. This happened in cases where two rules conflicted or when the treebank data appeared to have mistakes in it. Therefore, there exist known errors in the final set of coordination structures that was used though no attempt was made to quantify them. Likewise, the entire corpus was not manually inspected on each iteration of the development of the rules or even after the last iteration, so it may be possible that this algorithm could be improved with further effort.

While care was taken to make sure the resulting corpus of coordination structures looks to be of high quality using manual inspection, no attempt was made to measure the accuracy of the CSPA. However, the quality of the resulting coordination structures can be inferred indirectly in two ways. In Section §6.3 results of the Berkeley Parser applied to the task of coordination resolution are given and are shown to be better than the previous state-of-the-art on the Hara corpus (described in Section §2.3.) The output of the Berkeley Parser is syntactic trees in PTB format and the CSPA is applied to this output to give the coordination structures which are then evaluated against the coordination structures in the Hara corpus. The high performance of this system suggests that the CSPA is consistent with the approach taken by Hara et al. to determine coordination structures from PTB-style syntactic structures. A second way of indirectly measuring the quality of the CSPA is to consider results given in Section §6.2 which gives coordination resolution results for several systems on the BIOCC corpus (described in Section §2.2.) One of the highest performing systems

was trained on the coordination structures provided by CRAFT via the CSPA. This again suggests that the CSPA produces coordination structures that are consistent with yet another approach for obtaining coordination structures that was employed for the BIOCC corpus.

2.1.3 Coordination Types

As described in the previous section, the node type of the parent node of each conjunction node was designated as the coordination type for the corresponding coordination structure. Table 2.2 gives an example of a coordination structure for each type. These examples demonstrate the diversity of the coordination structures found in the CRAFT corpus. Table 2.3 shows the distribution of the different coordination types in CRAFT. The most frequent coordination type, "NP" accounts for 41.4% of all coordination structures. The next most frequent type, "NML" which is also closely related syntactically to the "NP" type, accounts for 19.5% and together they account for 60.9% of all coordination structures. In contrast, the eleven least frequent types, grouped together under "OTHER", account for only 3.4%. These types correspond to citations (89 occurrences), subordinate clauses (66), quantifier phrases (54), list markers (37), adverb phrases (34), parentheticals (19), fragments (9), captions (1), particles (1), yes/no questions (1), and unknown constructions (1). The final column of Table 2.3 shows the average length, in words, of the conjuncts of the coordination structures for each type. The conjuncts of coordination structures of type "ADJP" are, on average, 1.7 words long. In constrast, the conjuncts of coordination structures of type "S" are 14.1 words long. This suggests that coordination structures of certain types can be found within a comparatively smaller context than others.

2.2 BIOCC

A new corpus consisting of coordinating conjunctions and their respective conjuncts were manually annotated for training and evaluation of the coordination resolution methods presented in this dissertation. The corpus is named BIOCC because it contains biomedical coordinating conjunctions and their respective conjuncts. This corpus was built to demonstrate how coordination

Table 2.2: Examples of coordination structures in CRAFT for each type. Square brackets are used to clarify conjunct boundaries when two conjunctions appear (see the "VP" row) or extra context is provided (see the "ADJP" row). The example given for the "OTHER" category is of type "SBAR" which corresponds to subordinate clauses.

Type	Description	Example
NP	noun	motor activity and higher cognitive function
NML	nominal	Chr 10 and Chr 19
VP	verb	[has already been processed and regenotyped] and is now part of the Mouse Brain Library
S	sentence	Our comparisons are based on five strains, and one consequence of this modest sample size ...
ADJP	adjective	the optic chiasm are [malformed] or [absent] in Vax1 knockout mice.
PP	preposition	in the striatum and in many other regions ...
UCP	unlike constituents	pigmented or albino
OTHER	subordinate clause, quantifiers, etc.	that CLN2 is a tripeptidase, and that this inhibitor it represents the ...

Table 2.3: The distribution of coordination types in CRAFT. Each row gives for each coordination type the name, the number of coordination structures of the type, the percent of all coordination types that have the type, and the average length in words of the conjuncts of the coordination structures of the type. For example, the first row indicates that 3,746 (41.4%) coordination structures have the type "NP" and the average length of the conjuncts of those coordination structures is four words.

Type	Count	%	Len.
NP	3746	41.4%	4.0
NML	1760	19.5%	2.0
VP	1448	16.0%	8.6
S	657	7.3%	14.1
ADJP	598	6.6%	1.7
PP	266	2.9%	8.4
UCP	258	2.9%	2.7
OTHER	312	3.4%	4.5
Total	9045	100.0%	5.0

resolution techniques perform on randomly selected sentences from the biomedical literature. In contrast, the sentences in CRAFT and GENIA belong to articles that were carefully selected to meet specific criteria. Therefore, performance of coordination resolution systems measured using these corpora may not be as generalizable. The BIOCC corpus also provides insight into how coordination

resolution can be accomplished without full-syntactic markup. In Section §5.3, a parser-free method (i.e. a method that does not utilize a syntactic parser) is described. The annotation of BIOCC described here along with the parser-free approach, presents the possibility of quickly porting this coordination resolution approach to a new domain (e.g. clinical data or legal documents) without requiring availability of full-syntactic markup. Finally, the BIOCC corpus provides inter-annotator agreement results which allows comparison of coordination resolution between humans and machines.

BIOCC consists of 5,000 sentences which contain a total of 7,384 coordination structures. Four thousand sentences were annotated once by myself and make up a training data set which was used for algorithm development and error analysis. The remaining 1,000 sentences were annotated twice and differences were adjudicated. These sentences make up a hold-out evaluation data set. The annotation scheme used is very simple and consists of two annotation types: "conjunction" and "conjunct". The conjunction annotation type has a single one-to-many relationship with the conjunct annotation type so that conjunction annotations are unambiguously linked to their respective conjunct annotations. Both conjunction and conjunct annotations were restricted to covering a single contiguous span of text.

2.2.1 Sentence Selection and Pre-Annotation

The sentences used for BIOCC were randomly selected from full-text open-access scientific articles provided by a bulk download available from PubMed Central.[4] The articles are provided in XML format and so a simple script was created to strip out XML tags as well as sections that are of no interest here such as references, authors, glossaries, etc. The resulting corpus consists of 83,128 plain text files. Next, the OpenNLP sentence segmenter[5] was run over the entire corpus with each generated sentence added as a line to a single file which consisted of 12.9 million sentences. From this file 5,000 sentences containing the word "and" or "or" were randomly selected for annotation. In a

[4] `http://www.ncbi.nlm.nih.gov/pmc/about/ftp.html`. Downloaded on September 9th, 2008
[5] `http://opennlp.sourceforge.net/`

few cases the acronym "OR" was found instead of an actual conjunction. As such, twelve sentences in the corpus do not have a conjunction in them. The 5,000 sentences were automatically pre-annotated such that all occurrences of the tokens "and", "or", and "and/or" were annotated with the conjunction annotation type. Additionally, if the two words on either side of the conjunction had matching part-of-speech assignment (using the part-of-speech tagger described in Section §3.1.1), then the two words were annotated as two conjunct annotations of the conjunction annotation. About 37% (or 2,709) of all the conjunctions were annotated this way. Of these, 73.9% proved to be the correct conjuncts of the conjunction. More sophisticated automatic pre-annotation was avoided to limit the amount of bias towards specific coordination resolution approaches in the data set and to limit the amount of noisy (i.e. incorrect) annotations presented to the human annotator.

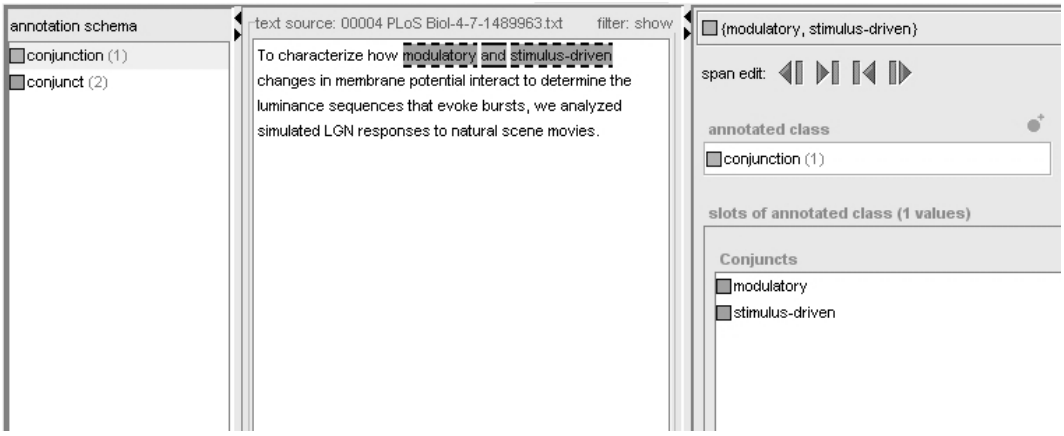

Figure 2.1: A screenshot of Knowtator as employed for the coordination annotation task. In this example, a single conjunction annotation is shown with its corresponding conjunct annotations.

2.2.2 Manual Annotation

Manual annotation was performed using Knowtator[38] as shown in Figure 2.1. All 5,000 sentences were annotated by myself. While annotating the first 1,000 sentences, annotation guidelines were developed for the annotation task. An underlying assumption of these guidelines is that humans have a good intuitive sense of what the correct coordination structure is and that a rich linguistic formalism is not required to understand and consistently perform the task. As such, the

guidelines are mostly procedural in nature and consist of specific rules, helpful strategies, and many examples. The first rule in the guidelines is heuristic in nature and is given here:

> Maximize parallel structure rule. The over-arching goal of selecting the spans of a conjunct is to maximize the "parallelness" of the conjuncts. The rules below do not cover all possible situations that you will encounter. So, when in doubt, try to choose conjuncts that look like each other or exhibit parallel structure.

The remaining rules cover such details as whether to include conjunct delimiting punctuation (e.g. commas and semi-colons) in the span of a conjunct (answer is "no"), how to handle parenthetical material, and whether conjunct boundaries must correspond to token boundaries (answer is "no"). The strategies provided give suggestions for effective annotation such as restating a sentence with a candidate conjunct removed. For example, a sentence that starts with:

- To characterize how modulatory and stimulus-driven changes in ...

can be restated as:

- To characterize how stimulus-driven changes in ...

to test the plausibility of assigning "modulatory" as the first conjunct. Other strategies discussed include annotating nested conjunctions first (conjunctions which appear in the conjunct of another conjunction), annotating the conjunct that follows the conjunction first (as it is sometimes easier to identify), and looking for words such as "between", "either", and "both" which often precede the first word of a conjunct. The guidelines allow for referring to the source document from which the sentence was extracted as well as any other resources on the Internet such as Wikipedia,[6] search results, and PubMed.[7] In many cases, the choice of the annotated coordination structure was directly influenced by seeing the surrounding context of the sentence being annotated. For example, sentences corresponding to a figure or table caption were much easier to interpret after examining the table column headings or the figure's labels. Similarly, consulting with domain

[6] http://www.wikipedia.org/
[7] http://www.ncbi.nlm.nih.gov/pubmed/

experts was also allowed for. In particular, because I have very little formal training in Biology, there were many sentences which were structurally ambiguous to me with respect to the contained coordination structures. When I came across a sentence that I could not confidently interpret after consulting various Internet resources I would consult a Biology domain expert for help. This often brought clarity to understanding the sentence. After completing the annotation guidelines, which were about 3,100 words long, the first 1,000 sentences were reviewed to ensure they were annotated consistently with the guidelines. These sentences were added to the training set.

The remaining 4,000 sentences were annotated as follows. I annotated 1,000 sentences which were used for the evaluation set. I then annotated the remaining 3,000 sentences which were added to the 1,000 sentences annotated during annotation guideline construction. These 4,000 sentences became the training set. After completing the training set, I then re-annotated the first 500 sentences in the evaluation set in an effort to increase the consistency of the annotations. The intra-annotator agreement of this effort is shown in Table 2.4. Approximately six months transpired between the two annotation efforts for these 500 sentences. The second half of the evaluation set was annotated by a Computer Science graduate student with education background in both Natural Language Processing and Molecular Biology. She is referred to as "Elizabeth" in Table 2.4 below. Her training for the annotation task consisted of two one-hour meetings and several hours of annotation practice. In the first meeting, the task was introduced, the annotation guidelines provided and summarized, and annotation exercises provided from the first 1,000 annotated sentences. At the second meeting, the annotation exercises were reviewed and questions were answered. After the second meeting she annotated the 500 sentences in the second half of the evaluation set with minimal interaction with me. Thus, all 1,000 sentences in the evaluation set were annotated twice. All differences between the two sets of annotations for the respective sets of 500 sentences were adjudicated by myself so that all differences were resolved.

After all annotation was complete a script was run to examine the annotation data for likely errors. For example, the script checked that all conjunction annotations were annotated with at least two conjunct annotations: one of which precedes and one that follows the conjunction. It

checked for matching parentheses and quotation marks (i.e. if a conjunct annotation contains a left parenthesis, then it should probably contain a right parenthesis) and looked for inconsistent nested conjunctions (i.e. if a conjunct contains a conjunction, then the conjuncts of the nested conjunction should be constrained by the extent of the nesting conjunct.) This script uncovered dozens of errors in the data that were manually fixed.

2.2.3 Annotator Agreement

The 1,000 sentences that comprise the evaluation data set were all double annotated to increase the consistency of the data and Inter-annotator agreement (IAA) and Intra-annotator agreement were calculated. Inter- and Intra-annotator agreement was calculated as positive-specific agreement[22] at the conjunct and conjunction levels. Conjunct level agreement measures agreement of conjuncts in two annotated coordination structures. Two conjunct annotations agree if they have the same spans (i.e. they have the same begin and end offsets in the text) and they are attached to the same conjunction. Conjunction level agreement is much stricter and measures the complete agreement of two coordination structures. Agreement at the conjunction level requires that both coordination structures have the same number of conjuncts and the respective conjunct annotations agree. Measuring agreement on both levels was simplified by the fact that there was perfect agreement as to what words were conjunctions so that agreement was always measured by comparing two coordination structures, one from each annotator.

Table 2.4 provides the IAA results for BIOCC. The IAA between Elizabeth and myself (labeled as "Philip") was 83.88 which measures how consistently two individuals can perform this annotation task when working independently. It is interesting to note that the agreement between Elizabeth and the final adjudicated annotations is higher (94.26) than that of Philip (89.34). This is presumably because Elizabeth, who holds a bachelor's degree in Molecular Biology, was better able to interpret the sentences correctly. This result is somewhat suspect, however, since I performed the adjudication and it may reflect my own willingness to accept her judgment in the face of my own uncertainty. Even still, there were many cases in which I came to understand why my annotation

was incorrect after conversing with Elizabeth. Intra-annotator agreement was measured for the second half of the evaluation data set and the agreement numbers are quite comparable to that of the inter-annotator agreement.

Table 2.4: Inter- and intra-annotator agreement on the hold-out evaluation set are shown. IAA for 500 sentences annotated individually by both Philip (myself) and Elizabeth and then adjudicated by myself are shown in the first three rows. Intra-annotator agreement for the 500 sentences of the evaluation set that were annotated twice by myself and then adjudicated by myself are shown in the next three rows. "Philip I" refers to annotations created the first time I annotated these 500 sentences while "Philip II" refers to the annotations created during re-annotation.

Annotator 1	Annotator 2	Conjunct Level	Conjunction Level
Philip	Elizabeth	91.81	83.61
Philip	Adjudicated	94.89	89.34
Elizabeth	Adjudicated	96.93	94.26
Philip I	Philip II	91.99	85.34
Philip I	Adjudicated	94.98	90.87
Philip II	Adjudicated	96.83	93.50

It is informative to examine IAA in greater detail across a number of features characterizing conjunctions. For example, Table 2.5 shows IAA for conjunctions that have two, three, and four or more conjuncts. The data show that when there are four or more conjuncts agreement is higher. This is presumably because when there are more conjuncts it is easier to identify them because they generally read as a listing of clearly related and enumerated items that are delimited by punctuation (usually commas.)

Table 2.6 shows conjunction level IAA for differing sizes of the largest conjunct measured by number of words. Not surprisingly, coordination structures with shorter conjuncts have higher agreement than those with longer conjuncts. Agreement is even higher (94.69) for conjunction structures that have only two one-word conjuncts. Resolving such one-word conjunct pairs is the subject of Chapter 4.

Table 2.7 shows conjunct level IAA for different part-of-speech tags for the first word of a conjunct. BIOCC was not annotated with coordination types. This is an important deficiency of the BIOCC corpus as it will be shown that coordination type in the CRAFT corpus will be effectively

Table 2.5: Conjunction level IAA is shown for the 500 sentences of the evaluation data set annotated by Philip and Elizabeth for conjunctions grouped by the number of conjuncts they have. The number of conjuncts for a given pair of coordination structures was chosen (arbitrarily) to be what was annotated by Philip. The first row of this table can be read as "The agreement of conjunctions annotated with two conjuncts was 83.28 for 634 coordination structures.

Conjuncts	IAA	Total
2	83.28	634
3	82.61	69
4+	93.10	29
Total	83.61	732

Table 2.6: Conjunction level IAA is shown for the 500 sentences of the evaluation data set annotated by Philip and Elizabeth for conjunctions grouped by the number of words of the longest conjunct. The number of words for the longest conjunct for a given pair of coordination structures was chosen (arbitrarily) to be what was annotated by Philip. The first row of this table can be read as "The agreement of conjunctions whose longest conjunct is one word was 91.39 for 267 coordination structures."

words	IAA	Total
1	91.39	267
2	87.50	80
$3-4$	85.06	87
$5-9$	78.17	142
10+	73.72	156
Total	83.61	732

exploited in Section §5.3.4. However, the part-of-speech tag of the first word of a conjunct can give a rough approximation of the coordination types. Table 2.7 shows that conjuncts that begin with a "NN" (for noun) and "CD" (for cardinal number) are annotated more consistently than conjuncts that begin with other part-of-speech tags. This may be because conjuncts beginning with these two tags may tend to be shorter than conjuncts that begin with, for example, "VB" (for verb.) These data do not give any intuition about the annotation consistency of conjuncts for one coordination type that is shown to be quite difficult to correctly resolve: the sentence type.

Table 2.7: Conjunct level IAA is shown for the 500 sentences of the evaluation data set annotated by Philip and Elizabeth for conjuncts grouped by the part-of-speech tag of the first word of each conjunct. The tags are grouped together into families such that, for example, all part-of-speech tags beginning with "NN" are counted together. The part-of-speech tag for the first word of the conjunct for compared conjuncts was chosen (arbitrarily) to be what was annotated by Philip. The first row of this table can be read as "The agreement of conjuncts whose first word had a part-of-speech tag of 'NN' was 94.91 for 687 coordination structures."

POS	IAA	Total
NN	94.91	687
JJ	90.61	277
VB	88.84	242
DT	85.82	134
CD	96.70	91
IN	89.53	86
OTHER	86.17	94
Total	91.81	1,611

2.3 Hara Corpus

The work described in Section §1.1.4 by Hara et al. was performed on the beta version of the GENIA corpus[24]. This version of GENIA consists of five hundred syntactically annotated abstracts in a format similar to that of PTB. The full corpus consists of 1,999 abstracts. One important modification to the GENIA annotation schema from the PTB annotation schema of relevance here is that coordination structures are explicitly annotated with a "COOD" label. Nodes with this label that surrounded "and", "or", and "but" were converted into coordination structures by Hara et al. for their experimentation. This data set was given to me for use in this research by Kazuo Hara via email and is therefore referred to as the Hara Corpus in order to distinguish it from GENIA. This corpus allows direct comparison with their work to be reported (see Section §6.3.) The corpus consists of 4,529 sentences that contain 3,598 coordination structures and is split into five folds to facilitate cross-validation.

2.4 Evaluation

The automatic coordination resolution systems explored in this dissertation are evaluated such that the results are directly comparable with the IAA metrics described in Section §2.2.3. Evaluation results are reported either at the conjunct or conjunction levels. A conjunct is counted as correct if it has the same span as a conjunct in the gold-standard data and it is attached to the same conjunction. Conjunct level evaluation is reported as F-measure because the number of conjuncts produced by a coordination resolution system will vary from the number of conjuncts in the gold-standard data and thus precision and recall will differ. A coordination structure produced by a coordination resolution system is counted as correct if it has the same number of conjuncts as the gold-standard coordination structure and all of the conjuncts exactly match. This is a very exacting criteria because it is possible to have a system-generated coordination structure that is very close to the gold-standard coordination structure and no partial credit is given for having most of it correct. Conjunction level evaluation measures how many coordination structures are correct as accuracy. Because the coordination resolution system will always generate (or should be expected to generate) the same number of coordination structures as the gold-standard data it is not useful to distinguish incorrect coordination structures as being either false positives or false negatives. That is, precision and recall are identical for conjunction level evaluation. All results presented in this dissertation should be assumed to be for the conjunction level unless otherwise noted.

In most of the experimental settings found in this dissertation gold-standard conjunctions are provided as input to the coordination resolution system. This is justified because "and" and "or" can be reliably identified as coordinating conjunctions. This was empirically shown to be true during the annotation of the BIOCC corpus. As discussed above, the simple sentence selection mechanism which selected only sentences containing the words "and" and "or" did this without case sensitivity. This process generated only twelve out of 5,000 sentences which did not contain a conjunction. Had sentence selection used case sensitive matching it would not have selected any of

these twelve sentences. Furthermore, subsequent experimentation with a conjunction finder on the BIOCC corpus was 100% accurate.

Providing gold-standard conjunctions simplifies evaluation because accuracy can be used instead of F-measure. Accuracy is simply the total number of system generated instances that are correct divided by the total number of gold-standard coordination structures. This is equivalent to recall and is directly comparable to how evaluation was performed by Hara et al. The CRAFT corpus contains a total of 9,045 coordination structures, 7,176 of which are in the training set and 1,869 the test set. Thus the denominator for accuracy when measured against CRAFT will always be 7,176 and 1,869 when evaluated on the training and test sets, respectively. When cross-validation is performed, the results are micro-averaged across the folds.

In some cases, precision, recall, and F-measure will be calculated. For example, in Chapter 4, the gold-standard conjunctions are not provided because only one-word conjunct pairs are considered, i.e. coordination structures consisting of exactly two conjuncts, each of which consist of one word. Precision is different than recall in cases where a coordination resolution approach does not generate a coordination structure for each provided gold-standard conjunction. For example, the parser-based approach discussed in Section §5.1 does not generate coordination structures for sentences that it fails to parse.

2.5 Discussion

CRAFT was chosen as the primary corpus used for driving research and development of the coordination resolution approaches explored in this dissertation. This choice was based on a number of factors. Having the type of coordination structure turns out to be very important (see Section §5.3.4.) Unfortunately, BIOCC does not provide coordination types. Furthermore, having full syntactic annotation of the data allows for a fair comparison of the traditional approach of applying a syntactic parser to the task of coordination resolution with an alternate approach introduced in Section §5.3. CRAFT was chosen in preference to GENIA or the Hara subset somewhat arbitrarily, though the latter is considerably smaller than the CRAFT corpus. However, because

of an increasing interest in the Biomedical Natural Language Processing community in analyzing full-text articles as evidenced in recent competitive evaluations such as the TREC Genomics[19] and BioCreative[20, 25] competitions, the CRAFT corpus is attractive because it consists of full-text articles. Though CRAFT was the primary corpus utilized in this dissertation, all three corpora were used to some extent to develop and evaluate the coordination resolution systems described here.

Chapter 3

Features

In Chapters 4 and 5 three coordination resolution approaches are introduced and analyzed. Each approach leverages statistical classifiers that use a variety of features described in this chapter. Most of the features used can be considered as lexical or "shallow" features that are derived from surrounding words. These are described in Section §3.1. A major contribution of this work is a novel way of employing language models to improve coordination resolution. In Section §3.2, a technique for creating features from language model probabilities is described. Careful attention to how the language model is built and used is crucial to its performance. A metric for directly measuring the fitness of a language model for coordination resolution is introduced and used to determine the features chosen for building and using the language model. Other classifier features, such as the word-level orthographical and semantic similarity features described in Section §4.2, will be introduced and discussed in the contexts where they are used.

3.1 Lexical Features

The lexical features used throughout this dissertation are summarized in Table 3.1. These features have been widely used for a variety of natural language processing tasks. In fact, most of these features have been employed for coordination resolution by others (c.f. [45].) All of these features require either preprocessing or presence of gold-standard labeled data. For example, the "word" feature requires either gold-standard tokens or the output of an automatic tokenizer. Throughout this dissertation, the following gold-standard data is used unless otherwise noted: sentence bound-

aries, token boundaries, and conjunctions. Sentence and token boundaries are provided under the assumption that both tasks can be done with very high accuracy for the biomedical domain if given enough attention. However, in Section §6.2 BIOCC is evaluated with no gold-standard data provided except conjunctions. Gold-standard conjunctions can be found automatically with near perfect accuracy and so they are provided simply for processing convenience. Part-of-speech tags may be provided or computed depending on the context. All other features are computed. Stems, for example, are computed using the English Snowball Stemmer.[1] Most of the features in Table 3.1 are simple binary features. For example, the feature type "numeric type" is used to describe a string (usually a word) with respect to whether or not the string contains digits. Similarly, "contains hyphen" is a simple binary feature that determines whether a word contains a hyphen or not.

Table 3.1: This table summarizes the word-level features that are common to many of the classifiers used in this dissertation.

feature type	feature description
word	the spanned text of a token
lower case word	a word lowercased
POS_{gold} tag	gold-standard part-of-speech tags
$\text{POS}_{\text{craft}}$ tag	CRAFT-trained part-of-speech tagger tags
$\text{POS}_{\text{genia}}$ tag	GENIA-trained part-of-speech tagger tags
stem	the stem of the word using the English Snowball stemmer
feature n-grams	the concatenation of features from n contiguous words
n-char prefix	the first n characters of a word
n-char suffix	the last n characters of a word
capital type	all uppercase, all lowercase, initial uppercase, mixed case, or N/A
contains hyphen	indicates if word contains a hyphen
numeric type	digits, year digits (e.g. 2010), alphanumeric, some digits*, roman numeral, or N/A
distance	the distance in words between two words one of which may be the conjunction of focus

*The feature "alphanumeric" means that all characters are either digits or letters while "some digits" includes any token containing a digit character.

[1] `http://snowball.tartarus.org/`

3.1.1 Part-of-Speech Tags

As mentioned above, part-of-speech tags can either be computed using an automatic part-of-speech tagger or provided directly from gold-standard data. Both the CRAFT and GENIA corpus provide gold-standard part-of-speech tags which can be used for training a part-of-speech tagger. For both corpora it may be advantageous to use a part-of-speech tagger that is trained on either corpus when using part-of-speech tags as features for coordination resolution. However, it is important to avoid training and then later running a part-of-speech tagger on the same data. When evaluating coordination resolution approaches using the Hara corpus, the part-of-speech tagger that is used will be trained on all of GENIA excluding the Hara corpus. $\text{POS}_{\text{genia}}$ refers to GENIA-trained tagger tags. The CRAFT corpus is somewhat more problematic for performing cross-validation of coordination resolution when using the training data set. To address this issue, leave-one-file-out training for each of the 36 files in the training data set (i.e. a tagging model for each file was trained on the other 35 files) was performed. Each file was then tagged using the part-of-speech model that was built excluding the tagged file. The predicted part-of-speech tags were stored for subsequent use. Because the part-of-speech tags were stored, it was not necessary to invoke thirty-six separate part-of-speech taggers each time CRAFT-trained tagger tags, i.e. $\text{POS}_{\text{craft}}$ tags, were used in a cross-validation experiment on the CRAFT corpus. For the CRAFT test data set, a part-of-speech tagging model trained using the entire training data set is used. When gold-standard part-of-speech tags are used the label POS_{gold} is used. However, POS_{gold} may refer to GENIA tags or CRAFT tags depending on which corpus is being used.

The part-of-speech tagger used was one that I created and is available in ClearTK and is described in [39]. The part-of-speech tagging models that were built make use of OpenNLP's maximum entropy library and perform at 96.3% and 98.3% accuracy for CRAFT and GENIA, respectively. Because this tagger was built with ClearTK, several other classifier libraries could be experimented with which may find a better performing tagger model. However, this part-of-speech tagger compares favorably with state-of-the-art performance on the GENIA corpus which

is 98.6% as reported in [48] and is tolerably fast for tagging large amounts of data. For example, part-of-speech tagging the nearly 13 million sentences for the language model described below was completed in less than seven hours at a rate of about 10,000 words per second.

3.2 Language Model Features

As discussed in Section §2.2.2, a simple sentence re-write strategy was found to be useful for consistently determining the conjuncts of a conjunction. For a candidate conjunct simply remove the candidate conjunct along with the conjunction and examine the fitness of the resulting sentence. For example, consider the following sentence:

- Subsequent measurements of *striatal volume* **and** *neuron packing density* were corrected for volumetric shrinkage.

When applying the rewrite strategy to the correct conjunct "striatal volume" the resulting sentence is:

- Subsequent measurements of neuron packing density were corrected for volumetric shrinkage.

This sentence looks much better than a re-write corresponding to the incorrect conjunct candidate "measurements of striatal volume" which results in the following sentence:

- Subsequent neuron packing density were corrected for volumetric shrinkage.

This sentence is much less coherent than the result of the rewrite corresponding to the correct conjunct given above. Because this heuristic is useful for reliably determining coordination structures, the hypothesis that it could be exploited to improve automated coordination resolution is explored. The remainder of this section describes how this heuristic was exploited to improve coordination resolution and reports experimental results that support the approach taken.

For each conjunction there is always a fixed number of candidate left conjuncts corresponding to the number of tokens to the left of the conjunction. For the sentence given above each of the

possible left conjuncts are given in Table 3.2 along with their corresponding sentence rewrites. A set of candidate right-hand-side conjuncts along with corresponding sentence rewrites can also be generated in a very similar way. The relative "fitness" of these sentences can be compared using a language model. Ideally, the sentence rewrite corresponding to the correct conjunct candidate will have the highest probability returned from a language model than the other candidate sentences. If so, then this strategy could be employed directly to the task of coordination resolution by simply choosing the conjunct corresponding to the highest probability sentence rewrite. Unfortunately, as described below, experimentation did not demonstrate this to be a feasible approach. Even so, if the candidate sentence corresponding to the correct conjunct has a higher probability than most of the other candidate sentences, then this information can be used to create an effective feature for a classifier. Experimentation described in later chapters shows this to be true.

Table 3.2: All five possible left conjuncts are given along with the sentence rewrites for the following sentence: "Subsequent measurements of *striatal volume* **and** *neuron packing density* were corrected for volumetric shrinkage." Each row a candidate conjunct (labeled e.g. "1c") and the corresponding sentence rewrite for that conjunct (labeled e.g. "1s").

	Candidate Conjunct and Sentence
1c	Subsequent measurements of striatal volume
1s	Neuron packing density were corrected for volumetric shrinkage.
2c	measurements of striatal volume
2s	Subsequent neuron packing density were corrected for volumetric shrinkage.
3c	of striatal volume
3s	Subsequent measurements neuron packing density were corrected for volumetric shrinkage.
4c	striatal volume
4s	Subsequent measurements of neuron packing density were corrected for volumetric shrinkage.
5c	volume
5s	Subsequent measurements of striatal neuron packing density were corrected for volumetric shrinkage.

One difficulty with simply passing each sentence rewrite to a language model and comparing the resulting probabilities is that the sentences differ in length. Comparing sentences of different length will give undesired preference towards shorter sentences (i.e. shorter sentences generally have

higher probability than longer ones.) One way to normalize for the differing sentence length is to divide the resulting probabilities by the respective length (in words) of the candidate sentence. While this normalization technique is better than none at all, a better approach is to incorporate the probability of the candidate conjunct (as a phrase) in addition to the probability of the sentence rewrite. Note that the number of words in the candidate conjunct plus the number of words in the corresponding candidate sentence will be the same for each candidate conjunct for a given conjunction and laterality (i.e. left or right). For the working example above, the number of words in the candidate conjunct plus the number of words in the sentence rewrite is always thirteen (excluding the period at the end of the sentence.) The probability of a candidate conjunct is calculated with Equation (3.1):

$$p(c) = e^{lm_{sentence}(c_s)+lm_{phrase}(c_c)} \tag{3.1}$$

Here, c_s refers to the candidate sentence and c_c refers to the candidate conjunct. The function lm_{phrase} returns a probability (as a logprob) from the language model for a phrase. Similarly, the function $lm_{sentence}$ returns a probability (as a logprob) from the language model for a sentence. The distinction between these two functions is that the latter incorporates beginning-of-sentence and end-of-sentence markers which help determine how likely the beginning and end of the sentence is, respectively.

The probabilities produced by Equation (3.1) tend to be vanishingly small and difficult to interpret as proper probabilities per se. They are only meaningful in relationship to each other. As such, the probability of each candidate is calculated using this simple metric and then rank ordered.[2] Because the number of candidate conjuncts varies from one sentence to the next (as determined by the token index of the conjunction) the absolute rank of a correct conjunct candidate is rather meaningless (e.g. a rank of 3 out of 6 candidates is much different than a rank of 3 out of 30 candidates.) For this reason, it is useful to translate the rank into a percentile such that the highest rank of 1 is translated to 100 and the lowest rank (the total number of candidates) is

[2] In fact, only the sum of the logprobs returned from lm functions is needed here.

translated to 0. This is accomplished by Equation (3.2):

$$rp(r, c) = (1 - \frac{r-1}{c-1}) \times 100 \tag{3.2}$$

Where rp stands for rank percentile given r (the rank) and c (the count of candidates.) Subtracting one from the numerator and the denominator accounts for the fact that the rank and count are indexed from 1 instead of 0 and allows for a rank of 1 to correspond to a rank percentile of 100. To illustrate, the rank percentile of $rp(1, 6) = 100$, $rp(2, 6) = 80$, $rp(3, 6) = 60$, $rp(4, 6) = 40$, $rp(5, 6) = 20$, and $rp(6, 6) = 0$. The value for $rp(1, 1)$ is defined to be 100. The rank percentiles of the candidate conjuncts will be applied to the task of coordination resolution by adding them as a features. However, it is informative to directly evaluate how good the rank percentile scores of the correct conjuncts are.

3.2.1 Description of Language Models

One reason that leveraging a language model is attractive is because there exists large amounts of readily available unlabeled biomedical text. The language models that were built here utilize a corpus of more than 80,000 full-text open-access scientific articles that were obtained from PubMed Central.[3] The articles are provided in a simple XML format which was parsed to produce plain text documents using only sections of the articles containing contentful prose (i.e. by excluding sections such as e.g. *acknowledgments* and *references*.) The plain text documents were automatically sentence segmented and tokenized and then optionally stemmed and part-of-speech tagged using either $\text{POS}_{\text{craft}}$ or $\text{POS}_{\text{genia}}$ tags. This resulted in six sets of sentences consisting of either words, stems, $\text{POS}_{\text{craft}}$ tagged words, $\text{POS}_{\text{genia}}$ tagged words, $\text{POS}_{\text{craft}}$ tagged stems, or $\text{POS}_{\text{genia}}$ tagged stems. Each set of sentences consists of nearly 13 million sentences and over 250 million words. A language model for each set of sentences was then built with the SRILM toolkit[46]. Default options were used for creating the language model except that the order of the model was set to four. Thus, a 4-gram model with Good-Turing discounting and Katz backoff

[3] `http://www.ncbi.nlm.nih.gov/pmc/about/ftp.html`. The corpus was downloaded in September of 2008.

for smoothing was built. A language model can be built with or without the "-tagged" option depending on whether the input sentences were part-of-speech tagged or not.

3.2.2 Evaluation

For each token to the left of a conjunction a candidate conjunct and sentence rewrite pair is derived. The generated conjunct candidates and sentence rewrites are produced to be consistent with the target language model. For example, if the language model was trained with sentences consisting of $\text{POS}_{\text{craft}}$ tagged stems, then the candidate conjunct will also consist of $\text{POS}_{\text{craft}}$ tagged stems as will the corresponding sentence rewrite. For each pair, a probability is calculated, and a rank percentile score is assigned to it relative to the other candidates. The same is done for tokens on the right-hand-side of the conjunction. Because multiple conjuncts can appear on the left-hand-side of the conjunction, the left border of the leftmost conjunct is considered here. All of the rank percentile scores are then averaged to give a single metric that can be used to approximate the "fitness" of a language model's use for the task of coordination resolution. Table 3.3 shows the average rank percentile for the correct left and right conjunct boundaries for all coordination structures in CRAFT and BIOCC for the six language models that were built. The worst performing language model is the one trained on sentences consisting of words only and gives overall average rank percentiles between 74.06 and 75.50. The remaining language models perform at roughly the same level giving overall average rank percentiles between 80.06 and 82.77 with a trend towards better performance when part-of-speech tags are used regardless of whether words or stems are used. The average rank percentile is calculated excluding trivial cases that occur when the conjunction is the penultimate word on either side of the sentence. Such cases will always have a rank percentile of one hundred because there is only one possible conjunct. Adding these cases inflates the average rank percentile by a small amount. For example, the score of 82.77 given for left conjuncts in CRAFT when using the language model trained with sentences consisting of $\text{POS}_{\text{craft}}$ tagged words is improved to 83.16 when the trivial cases are counted.

Figure 3.1 shows a histogram of the percent of correct left conjunct candidates that occur

Table 3.3: Language model performance using different features. Each row gives the performance of a language model as the average rank percentile of the correct conjuncts as measured by the language model using Equation (3.2). The first two columns characterize the sentences that were used to build the language model as discussed in Section §3.2.1. The letters "L" and "R" in the headers for the remaining columns specify whether left or right conjuncts are being evaluated for the named corpus. Thus, the third column of the fourth row says that the average rank percentile for left conjuncts in the CRAFT corpus was 82.77 when a language model was built using $\text{POS}_{\text{craft}}$ words.

Tokens	POS tags	CRAFT L	CRAFT R	BIOCC L	BIOCC R
words	none	74.74	74.06	75.50	74.84
stems	none	81.03	81.07	80.09	81.04
words	$\text{POS}_{\text{craft}}$	**82.77**	82.19	81.96	82.22
stems	$\text{POS}_{\text{craft}}$	82.00	82.23	80.84	**82.34**
words	$\text{POS}_{\text{genia}}$	82.69	82.00	**82.00**	82.01
stems	$\text{POS}_{\text{genia}}$	81.85	**82.25**	80.86	82.30

in a given rank percentile range when using the language model trained with sentences consisting of $\text{POS}_{\text{craft}}$ tagged words. Most notably 49.7% of the correct conjunct candidates have a rank percentile of 90 or higher. This includes 32.1% of correct left conjunct candidates that have a rank percentile of 100 (i.e. the model predicted the correct conjunct.) By adding the last five columns together, 89.6% of the correct left conjunct candidates have a rank percentile of 50 or higher. The overall average rank percentile for all of the correct left conjunct candidates was 82.77%. The median number of candidates (c) in Equation (3.2) on the left-hand-side is 17 (i.e. the median token index of the conjunction is 17). Very similar results were obtained for the right-hand-side data but were withheld for space considerations. The overall average rank percentile for right-hand-side conjuncts was 82.19% with a median number of 12 candidates. These data suggest that the rank percentile of the candidate conjuncts calculated as described above could be an effective feature to use for coordination resolution.

One could imagine many different variations of the approach described above for leveraging a language model for this task. Several alternate approaches to measure the fitness of a candidate conjunct boundary using a language model were attempted. In particular, instead of comparing entire sentence rewrites along with their corresponding conjunct candidates, comparison of n-gram

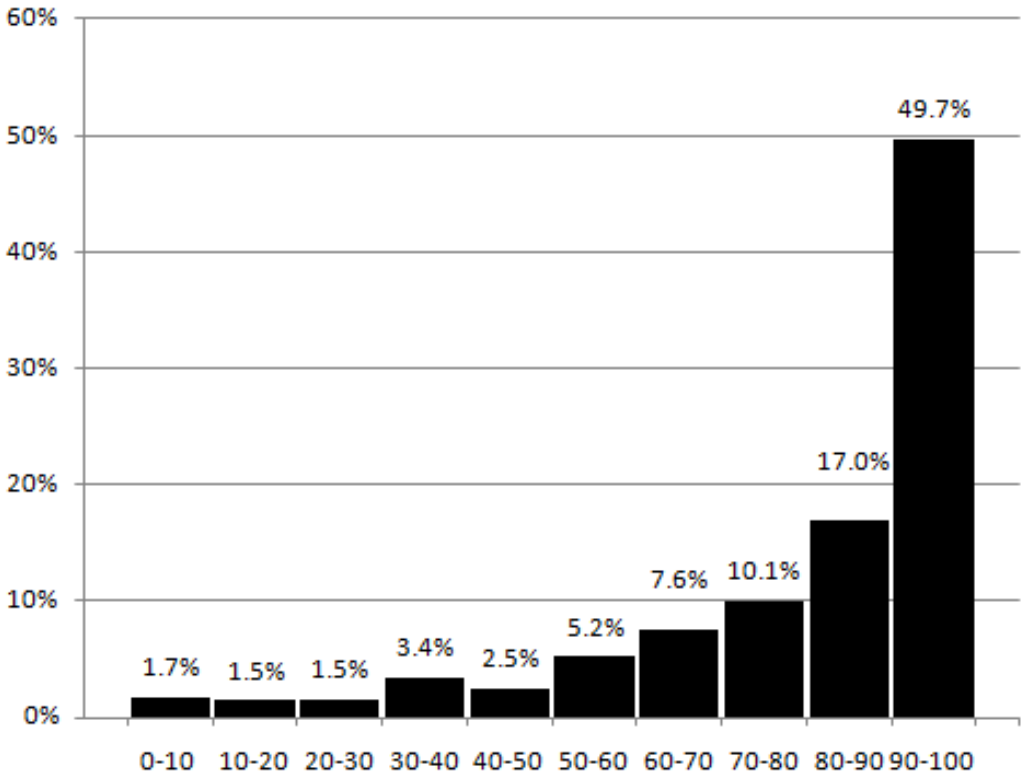

Figure 3.1: Percentage (y-axis) of correct left conjunct candidates at different rank percentiles (x-axis). The first column can be read as "The correct conjunct candidate had a rank percentile between 0 and 10 1.7% of the time." The columns add to one.

phrases consisting of words immediately preceding the candidate left conjunct boundary and words following the conjunction with differing values of n was conducted. Also, rather than adding the probability of the candidate conjunct as a phrase to the probability of the sentence rewrite, the sentence rewrite probabilities were instead divided by the number of words. Finally, experimentation with not using sentence boundary markers was also conducted. All of these alternate configurations resulted in worse system performance than the approach described and used above.

3.2.3 Error Analysis

Informal error analysis was performed on cases where the language model gave the correct candidate conjunct a rank percentile of 30 or lower. The cases examined fell under three broad categories. The first class of "errors" are those in which smarter candidate sentence construction could help. For example, consider the sentence:

- The MBL is both a *physical* **and** *Internet* resource.

The corresponding candidate sentence corresponding to the correct left conjunct is "The MBL is both a Internet resource." The sentence received a rank of 6 out of 6 (i.e. it ranked the worst out of the six possible left conjunct candidates.) It is likely that the phrase "both a Internet" is giving the sentence a low probability. This leads to two possible modifications to candidate sentence construction: remove "both" and "either" from candidate sentences and make instances of the words "a" and "an" consistent with the word to the right of the conjunction (when the left conjunct candidate begins immediately after one of these determiners.) A similar change resulted from noticing that when the left boundary of the left conjunct was the beginning of the sentence, then the candidate sentence for that conjunct candidate was not capitalized. So, candidate sentence construction was modified to make sure that all candidate sentences begin with a capitalized word. Table 3.4 summarizes the effect of these changes on overall rank percentile performance on the language model trained with sentences consisting of $\text{POS}_{\text{craft}}$ tagged words for the left conjuncts in CRAFT. Of the three changes, making sure the first word is capitalized had the largest positive

Table 3.4:

Modification	Average Rank Percentile	Δ
none	81.37	
"a" or "an" agreement	81.38	.01
remove "both" and "either"	81.61	.24
capitalize first	82.52	1.15
all modifications	82.77	1.4

impact.

Further error analysis might reveal more ways to preprocess candidate sentences so that the language model does a better job of predicting the correct candidate. However, the remaining errors that were examined fall into one of two broad categories: the coordination structure in the gold-standard data is incorrect or the language model approach described above appears to have failed. An example of the incorrect coordination structure is:

- The authors *thank Sandeep Raghow and Aldan Shank for technical assistance with the behavioral studies* **and** *Robert Lane for useful discussions.*

To correct this example, the word "thank" should be removed from the left conjunct. It is not surprising that the corresponding candidate sentence for the left conjunct is given a rank of 15 out of 15 possible left conjuncts:

- The authors Robert Lane for useful discussions.

The rank produced by the language model for the corrected left conjunct was 2 out of 15. Examining the treebank annotation for this sentence reveals that it is annotated correctly but the translation from treebank structures to coordination structures does not correctly handle this sentence. The gold-standard sentence is annotated as follows:

```
( (S (NP-SBJ (DT The) (NNS authors)) (VP
    (VP (VBD thank) (NP=1 (NP (NNP Sandeep) (NNP Raghow)) ...
```

```
      (PP=2 (IN for) (NP (NP (JJ technical) (NN assistance)) ...
    (CC and)
    (VP (NP=1 (NNP Robert) (NNP Lane))
      (PP=2 (IN for) (NP (JJ useful) (NNS discussions)))))
(. .)) )
```

In this example, the verb phrases are coordinated rather than the noun phrases because the verb is modified by a prepositional phrase once for each respective noun phrase. However, the coordination structure production algorithm described in Section §2.1.2 that creates coordination structures from treebank structures did not handle this case correctly.

The other broad category of errors produced by the language model can be described as failures of the approach. One such coordination structure that was poorly ranked is given in the following sentence:

- Add and Dom are estimates of the *additive* **and** *dominance* effects of genetic variation.

Here the candidate sentence corresponding with the correct left conjunct seems like a perfectly fine sentence but was ranked 6 out of 8.

3.3 Conclusions

In this chapter, a number of lexical features were itemized and described. In particular, special attention was given to part-of-speech tag features because they will prove to be important features to consider when comparing the coordination resolution approaches evaluated in subsequent chapters. A novel feature was described for performing coordination resolution that leverages a sentence rewriting heuristic by applying a language model to large amounts of unlabeled biomedical text. The language model feature was evaluated directly by measuring the average rank percentile for the correct conjuncts for both CRAFT and GENIA. This metric was shown to be useful for comparing different language models and for comparing different strategies for using the language model. Similarly, the rank percentiles given by the language model feature provides support to the

intuition that this feature will be of utility for coordination resolution. For example, as mentioned above, 89.6% of the correct left conjunct candidates were shown to have a rank percentile of 50 or higher. This suggests that for all but 10.4% of the examples, half of the candidate conjuncts can be safely excepted from consideration. In subsequent chapters the direct impact of the language model feature will be directly measured and shown to have a positive impact on performance.

Chapter 4

One-Word Conjunct Pairs

A large percentage of coordination structures can be described by a pair of one-word conjuncts. The following sentence contains two examples that illustrate this phenomenon:

- *Add* **and** *Dom* are estimates of the *additive* **and** *dominance* effects of genetic variation.

In the CRAFT corpus 2,528 (27.9%) out of the 9,045 coordination structures consist of two one-word conjuncts which will be referred to as one-word conjunct pairs (OWCPs). Correctly identifying coordination structures that are described by OWCPs is an important sub-task within the broader task of coordination resolution because there are a large number of them and because they are comparatively easy to correctly resolve than other coordination structures. By tackling OWCPs separately, overall coordination resolution performance is shown to improve. This is presumably because a large number of easy coordinations no longer need to be considered by downstream classification models allowing them to concentrate on the harder coordinations. In this chapter, an OWCP classifier is introduced and evaluated. Adding features that use orthographic similarity are shown to provide a small increase in classifier performance while adding the language model features described in Section §3.2 give a larger, though still modest, increase in performance. The best performing OWCP classifier is shown to outperform the Berkeley Parser when it is employed for this task though is still far below human inter-annotator agreement on the task.

An additional benefit of concentrating on OWCPs separately is that it provides a focused subtask for exploring the effectiveness of competing classifiers with various learning parameters and

measuring the effectiveness of different features. For example, in Section §4.2.3 semantic similarity features are explored and shown to be ineffective on correct classification of OWCPs which suggests they are not likely to be effective on the broader task of full coordination resolution. For this task each conjunction corresponds to either a positive or negative instance in the training data which means that there are only 9,045 training instances divided among the 10-fold cross-validation and the holdout evaluation. Such small training sets allow for rapid experimentation. A typical OWCP experiment takes a few minutes to complete rather than an hour or more for a typical experiment for the full coordination resolution system described in subsequent chapters.

4.1 One-Word Conjunct Pairs Classifier

The OWCP classifier takes as input for each conjunction the word to its left, w_L, and the word to its right, w_R. If the token immediately to the left of a conjunction is a common conjunct delimiter, i.e. a comma or semi-colon, then w_L will be the token to the left of the conjunct delimiter. Similarly, if the token immediately to the right of a conjunction belongs to a set of "trailing skip words", then w_R is the next token to the right. The set of trailing skip words was empirically derived using the most common words found in treebank nodes of type CONJP and includes the following words: *then*, *therefore*, *thus*, *also*, *even*, *hence*, *so*, *not*, *rather*, *only*, and *instead*. Throughout this dissertation this set of skip words was used to identify words to "skip" over when determining the left boundary of the right of the conjunction (i.e. they were not included as the first word of the right-hand conjunct.) For example, the word "not" is excluded from the second conjunct in this sentence:

- The angle recess . . . was *open* **and** not *occluded*.

Feature extraction is performed on w_L and w_R and a boolean classifier predicts "true" or "false" using the resulting features. The classifier returns a boolean prediction that says whether or not the coordination structure for the conjunction is a OWCP corresponding to w_L and w_R.

4.2 Features

4.2.1 Lexical Features

Table 4.1 lists the lexical features used by the OWCP classifier. While many different sets of lexical features were experimented with, thorough feature selection experimentation was not conducted for the OWCP classifier and the final set used here may be somewhat arbitrary. Intuitively, these simple lexical features should go a long way towards adequately characterizing OWCPs for classification because there are many local surface cues that can be exploited. For example, knowing whether a delimiter sits between w_L and the conjunction will likely be an important feature for negative examples (i.e. coordinations that are not OWCPs) and is captured by the word feature for w_{L+1}. To illustrate, the comma in the following example may be indicative of a coordination structure that is not a OWCP.

- ... were taken from the Mouse Brain Library, **and** point counting was ...

Table 4.2 shows the results of the OWCP classifier performance for 10-fold cross-validation on the training set of the CRAFT corpus. Classification models were trained using a number of different classifiers including three classifiers provided by Mallet[30] (Naïve Bayes, MCMaxEnt, and MaxEnt), the OpenNLP implementation of maximum entropy,[1] SVM^{light}[23], and LIBSVM[7]. These classifiers were chosen because ClearTK supports them. For the SVM^{light} and LIBSVM classifiers models were trained using linear and polynomial kernels. Grid search for learning parameter selection for both the polynomial (for degree three) and radial-basis function kernels was performed and the highest performing combination found (polynomial kernel with c=10, g=1, and r=1) is reported. Not surprisingly, the polynomial models outperform the corresponding linear models by 1.23 points and 1.24 points for LIBSVM and SVM^{light}, respectively. It is interesting to note that the three implementations of maximum entropy differ by as much as 3.76 points (between MCMaxEnt and the Mallent MaxEnt).

[1] `http://maxent.sourceforge.net/`

The final column of Table 4.2 shows the sample standard deviation of file-level performance as previously discussed Section §2.1.1. For the CRAFT training set, which is used here, there are thirty six data points used to calculate the standard deviation corresponding to the F-measure performance for each file. The standard deviations shown are relatively high when compared with the absolute F-measure differences between the different classifiers. In fact, the largest performance difference of 4.13 points (between the third row and first row) is smaller than the smallest standard deviation of 5.38. This indicates that while overall performance can fluctuate considerably from one classifier to another, overall performance is not a good indicator of which classifier will work the best for any given file. This conclusion is further reinforced by examining the standard deviation of the difference of two systems' performances on the thirty six files. For example, the standard deviation of the file-level performance differences (e.g. Naïve Bayes gives an F-measure of 80 for the first file and Mallet's MaxEnt gives an F-measure of 85, then the file-level performance difference is five points) for the first and third rows is 5.87.

Table 4.1: Baseline features for the one-word pairs classifier. The letters w_L and w_R refer to the words to the left and right of the conjunction, respectively.

feature name	feature taken from
word	w_{L-2}, w_{L-1}, w_L, w_{L+1}, w_{L+2}, w_{R-2}, w_{R-1}, w_R, w_{R+1}, w_{R+2}
lower case word	w_L, w_R
POS_{craft} tag	w_{L-2}, w_{L-1}, w_L, w_{L+1}, w_{L+2}, w_{R-2}, w_{R-1}, w_R, w_{R+1}, w_{R+2}
3 character suffix	w_L, w_R
3 character prefix	w_L, w_R
capital type	w_L, w_R
contains hyphen	w_L, w_R
numeric type	w_L, w_R
word bigram	w_Lw_R, $w_{L-1}w_L$, $w_{L-2}w_{L-1}$, w_Lw_{L+1}, $w_{L+1}w_{L+2}$, $w_{R-1}w_R$, $w_{R-2}w_{R-1}$, w_Rw_{R+1}, $w_{R+1}w_{R+2}$
part-of-speech bigram	w_Lw_R, $w_{L-1}w_L$, $w_{L-2}w_{L-1}$, w_Lw_{L+1}, $w_{L+1}w_{L+2}$, $w_{R-1}w_R$, $w_{R-2}w_{R-1}$, w_Rw_{R+1}, $w_{R+1}w_{R+2}$
stem bigram	w_Lw_R

Table 4.2: Performance of OWCP classifier using only lexical features on the training set of the CRAFT corpus. The columns labeled "TP", "FP", and "FN" correspond to the number of true positives, false positives, and false negatives, respectively. The columns labeled "P", "R", and "F" give the precision, recall, and F-measure, respectively. The final column labeled "σ" gives the sample standard deviation of the file-level F-measure performance.

Classifier	TP	FP	FN	P	R	F	σ
Naïve Bayes	1648	353	392	82.36	80.78	81.56	5.86
MCMaxEnt	1635	316	405	83.80	80.15	81.93	6.14
MaxEnt (Mallet)	1733	272	307	86.43	**84.95**	**85.69**	5.96
MaxEnt (OpenNLP)	1686	296	354	85.07	82.65	83.84	5.38
SVM$^{\text{light}}$ (linear)	1716	312	324	84.62	84.12	84.37	6.17
SVM$^{\text{light}}$ (polynomial)	1717	257	321	**86.99**	84.26	85.61	5.70
LIBSVM (linear)	1699	300	341	84.99	83.28	84.13	6.21
LIBSVM (polynomial)	1717	266	323	86.59	84.17	85.36	5.91

4.2.2 Orthographical Similarity Features

In some cases, the two words that make up a OWCP are quite similar. In this section the hypothesis that that word similarity can be exploited to improve classification performance on OWCPs is explored. In particular, experiments using both orthographical and semantic similarity were conducted. The idea is to capture examples of OWCPs where the two words are very similar orthographically or semantically but were missed by the classifier when trained on only the lexical features given in Table 4.1.

Error analysis of the OWCP classifier trained on only lexical features revealed false negatives where the OWCP consisted of two orthographically similar words. The following are some examples:

- ... receptors in skeletal patterning, *BMPR1A* **and** *BMPR1B*.
- ... in two different ligands in the BMP family, *Gdf5* **and** *Gdf6*,...
- ... *p50* **and** *p32* are absent in protein extracts of ...
- ... that interacts with infertility factors *DAZ* and *DAZL*

Admittedly, such examples constitute a relatively small percentage of the total number of false negatives (though this ratio was not actually calculated.) Even still, they look like examples that could be easily handled by simple orthographical similarity analysis.

Two metrics that calculate orthographical word similarity were explored. The first is based on Levenstein edit distance given in Equation (4.1) which will be referred to here as Levenstein similarity.

$$levsim(s_1, s_2) = 1 - \frac{levenstein(s_1, s_2)}{size(s_1) + size(s_2)} \tag{4.1}$$

The function "levenstein" returns the Levenstein edit distance which is described in [17]. The intuition behind this equation is the calculation of similarity as a ratio of the total number of edits to the total number of characters in both strings. This is divided by the total number of characters to give a ratio of edits to characters. This ratio will be between 0 and 1 with smaller numbers

corresponding to more similar words. Subtracting this from one will give a score of 1.0 for two identical words and 0 for words that have no similarity. For example, Equation (4.1) will return 0.875 for the words "aaaa" and "baaa" and 0.67 for the words "aaaa" and "aa".

The Levenstein similarity score for the words w_L and w_R was added as a feature to the OWCP classifier model in addition to the lexical features given in Table 4.1. Two modifications of the Levenstein similarity calculation were experimented with. The first treats all digits as being the same character (i.e. so that "Gdf5" and "Gdf6" would have a Levenstein similarity of 1.0) for the Levenstein similarity calculation. The second treats capital letters as being the same character (i.e. so that "BMPR1A" and "BMPR1B" would have a Levenstein similarity of 1.0). The results of adding the Levenstein similarity features are shown in Table 4.3. Here the results are reported for the LIBSVM classifier using a polynomial kernel with a degree of 3. While Mallet's MaxEnt performed slightly better for these features, results from LIBSVM are shown here because this classifier ultimately gives the best performance on this task as shown in Table 4.6. The first row shows the performance of the OWCP classifier using lexical features and the second row the performance when the unmodified Levenstein similarity between w_L and w_R is added as a feature. Performance improves slightly by 0.45 points between rows one and two. Treating digits as being the same in the Levenstein similarity calculation results in a negligible difference of 0.03 points. Treating capital letters as being the same gives another slight improvement of 0.32 points over using the unmodified Levenstein similarity and 0.77 points over the classifier that does not use the Levenstein similarity feature at all. Again, while the overall performance increased, the comparatively high sample standard deviations for file-level performance indicates that this feature does not reliably increase performance at the file-level.

The following examples are true positives that were added by the classifier as a result of adding Levenstein similarity features:

- Similarly, 6 to 9 month old *males* **and** *females* that are homozygous ...
- *preimplantation* **and** *postimplantation* embryos using ...

Table 4.3: Performance of OWCP classifier using lexical features and Levenstein Similarity. The results are from the LIBSVM classifier using the polynomial kernel with degree 3. The label "digits" refers to treating all digits as being the same. Similarly, the label "caps" refers to treating all capital letters as being the same.

Features	TP	FP	FN	P	R	F	σ
lexical features only	1717	266	323	86.59	84.17	85.36	5.91
+ levsim	1726	257	314	87.04	84.61	85.81	6.04
+ levsim + digits	1725	254	315	87.17	84.56	85.84	5.74
+ levsim + caps	1730	247	310	87.51	84.80	**86.13**	5.86
+ levsim + digits + caps	1728	250	312	87.36	84.71	86.01	5.79

- by which Snail *expression* **and** *proliferation* may be coupled
- (comprising axial *mesoderm* **and** *mesenchyme* cells).
- The tissues were then prepared for paraffin *embedding* **and** *sectioning*

It is interesting to note that these examples are a lot less orthographically similar to each other than the examples that motivated using orthographical similarity presented above. This may be symptomatic of a classifier that has been perturbed by a new feature such that some instances that were false negatives are now true positives and vice versa by random chance. The last three examples given above seem to reflect this kind of behavior rather than actually benefiting from the orthographically similar feature.

Because a simple orthographical similarity based feature gave a modest improvement in performance, it seems possible that using a much more sophisticated orthographical similarity metric might work even better. The second orthographical similarity metric explored was Lin's information-theoretic definition of similarity taken from [27] and is given by Equation (4.2).

$$linsim(s_1, s_2) = \frac{2 \times \sum_{t \in tri(s_1) \cap tri(s_2)} \log P(t)}{\sum_{t \in tri(s_1)} \log P(t) \sum_{t \in tri(s_2)} \log P(t)} \tag{4.2}$$

This equation provides a definition of similarity based on the character trigrams that two words share. The trigrams are weighted based on how frequently they occur in a large corpus

Table 4.4: Performance of OWCP classifier using lexical features and Lin Similarity. The results are from the LIBSVM classifier using the polynomial kernel with degree 3. The label "(wb)" refers to the use of two extra trigrams that incorporate beginning and end word boundaries. The label "digits" refers to treating all digits as being the same. Similarly, the label "caps" refers to treating all capital letters as being the same.

Features	TP	FP	FN	P	R	F	σ
lexical features only	1717	266	323	86.59	84.17	85.36	5.91
+ linsim	1717	260	323	86.85	84.17	85.49	5.99
+ linsim(wb)	1715	255	325	87.06	84.07	85.54	6.24
+ linsim(wb) + digits	1717	251	323	87.25	84.17	85.68	6.16
+ linsim(wb) + caps	1720	249	320	87.35	84.31	85.81	5.98
+ linsim(wb) + digits + caps	1720	249	320	87.35	84.31	**85.81**	5.59

of text which is captured by P(t). In this way, frequently occurring trigrams that two words share are not as important as infrequently occurring trigrams that they share. The function "tri" returns a list of character trigrams for a word. One modification that was made to this function is to add two additional trigrams corresponding to a beginning-of-word marker and the first two characters of the word and the last two characters of the word and an end-of-word marker. With this modification, for example, the token "Gdf5" would have the following trigrams: $<B>Gd$, *Gdf*, *df5*, and *f5*$<E>$. This change emphasizes the importance of the beginning- and end-of-word characters. As with Levenstein similarity, the Lin similarity score for the words $w_L$and w_R was added as a feature to the model. Similarly, treating all digits and capitalized letters as being the same was also explored. Trigram probabilities were calculated using the PMC corpus used for building the language model as described in Section §3.2.1. The number of trigram types counted using word boundary information, equated capital letters, and equated digits was 118,419 out of a total of 1.42 billion trigram instances. The performance of the resulting classifier is shown in Table 4.4. The best performing variation of the Lin similarity feature improved classification results by a modest .45 points (from 85.36 to 85.81).

Adding both the Levenstein and Lin similarities to the classifier results in a performance of 86.21 which is .08 points higher than the highest score shown in Table 4.3. Because of this very

small improvement in performance and because the Lin similarity metric requires more memory and CPU cycles this feature was abandoned.

4.2.3 Semantic Similarity Features

The two words that make up an OWCP are often semantically similar. The following examples are false negatives from the OWCP classifier that uses the Levenstein similarity as described above and the language model features as described below (see Section §4.2.4) that show two words that are semantically similar:

- ... to obtain cell counts *accurately* **and** *efficiently* ...
- ... the leptin receptor gene (Leprdb) that causes *obesity* **and** *diabetes* ...
- ... than their heterozygous *age* **and** *sex* matched littermates.
- ... The PCR products we *purified* **and** *sequenced* as described.

This observation supports the hypothesis that word-level semantic similarity could be exploited to improve performance of the OWCP classifier. As a preliminary study a list of word pairs corresponding to w_L and w_R for each conjunction was produced and given to Dr. Ted Pedersen who has done extensive work on semantic word similarity (see [42] or [41].) He ran these lists through three graph-based semantic similarity algorithms using programs created in his research lab: Lesk measure using UMLS-Similarity,[2] Lesk measure using WordNet-Similarity,[3] Wu and Palmer similarity measure using WordNet-Similarity.[4] The Lesk measure adapted for the UMLS and WordNet[1] compares the definitions of words to judge their similarity. The Wu and Palmer similarity measure[50] calculates similarity using the hierarchical depth of "lowest common subsumer" of the two words along with the depth of the words themselves. Each of these measures are graph-based, i.e. they measure distance between terms in a graph defined by some lexical or

[2] `http://search.cpan.org/dist/UMLS-Similarity`
[3] `http://www.d.umn.edu/~tpederse/similarity.html`
[4] `http://search.cpan.org/dist/UMLS-Similarity/lib/UMLS/Similarity/wup.pm`

Table 4.5: Performance of OWCP classifier using lexical and semantic similarity features. The results are from the LIBSVM classifier using the polynomial kernel with degree 3. The label "WN" refers to the use of WordNet while the label "WUP" refers to the Wu and Palmer measure.

Features	TP	FP	FN	P	R	F	σ
lexical features only	1717	266	323	86.59	84.17	85.36	5.91
Lesk + UMLS	1736	659	304	72.48	85.10	78.29	7.18
Lesk + WN	1752	469	288	78.88	**85.88**	82.23	7.08
WUP + WN	1732	262	308	**86.86**	84.90	**85.87**	5.98

terminological resource (e.g. UMLS or WordNet.) This requires that compared words actually map to entries in the resource. Unfortunately, the word pairs mapped to entries in UMLS and WordNet 33.7% and 43.8% of the time, respectively. Even still, the similarity scores that were generated for pairs of words that did map were added as features to the classifier with the hope that the mappings are more frequent for positive examples (i.e. when w_L and w_R corresponds to a OWCP.) Table 4.5 shows the effect of adding the semantic similarity features.

All three measures give a small increase in recall with the largest being that of the Lesk measure using the UMLS which gives a 1.71% absolute increase in recall. However, both features derived from the Lesk measurement have much larger decreases in precision. Only the Wu and Palmer measure provides an increase for both precision and recall resulting in a gain in F-measure by .51 points. Due to the "one-off" nature of this experiment and the modest performance gains, this feature was not subsequently used and further exploration of using semantic similarity was not done. However, the modest increase in performance is promising and suggests that further work incorporating semantic similarity might be fruitul. The low percentage of mappings suggest that a distributional semantic similarity approach might be better suited for this task.

4.2.4 Language Model Features

Features that make use of language model probabilities corresponding to competing candidate conjunct borders were described in Chapter 3. Specifically, the rank percentile of the candidate sentence corresponding to w_L and w_R were added as features. The performance of the OWCP

Table 4.6: Performance of OWCP classifier using lexical features and Language Model features. The results are from the LIBSVM classifier using the polynomial kernel with degree 3. The label "LM" refers to "Language Model" features.

Features	TP	FP	FN	P	R	F	σ
lexical features only	1717	266	323	86.59	84.17	85.36	5.91
+ LM	1764	222	276	88.82	86.47	87.63	4.55
+ LM + levsim	1779	205	261	**89.67**	**87.21**	**88.42**	**4.21**

classifier with this feature is shown in Table 4.6. Adding the language model feature improves performance of the classifier 2.27 points to an F-measure of 87.63. When the language model feature and the Levenstein similarity feature are both added to the classifier model, then performance is bumped up to 88.42, an increase of 3.06 points. It is also interesting to note that this highest performing OWCP classifier also has the lowest sample standard deviation of the file-level performance of 4.21. This is considerably lower than 5.91, the standard deviation of OWCP classifier using the LIBSVM classifier with only lexical features. The features used for this classifier resulted in the highest performing OWCP classifier on the cross-validation data set and was therefore the preferred configuration for subsequent experimentation.

The following examples are true positives that were added by the classifier as a result of adding these features:

- ... between the cornea (C) and iris root (I) was *open* **and** not *occluded*.
- ... but increased aqueous humor *production* **or** *flow* occurs during the ...
- ... is not dependent on functional *rod* **and** *cone* photoreceptors since these ...
- All mice were *bred* **and** *maintained* at The Jackson Laboratory.

Table 4.7: Best performing OWCP classifier compared with baseline classifier.

Features	TP	FP	FN	P	R	F
lexical + LM + levsim	1779	205	261	89.67	87.21	88.42
baseline classifier	1856	1287	184	59.05	90.98	71.63

4.3 Analysis of Results

4.3.1 Comparison with a Baseline Classifier

Of the 2,040 OWCPs in the training data set, $w_L$and w_R have the same part-of-speech tag (using gold-standard tags) for 1,856 (90.98%) of them. A very simple OWCP baseline classifier could always return true when $w_L$and w_R have the same part-of-speech tag. This classifier has a recall of 90.98 which is the highest recall of any of the classifiers reported above. Unfortunately, this classifier has much lower precision. Table 4.7 compares this proposed baseline system with the best performing system described above.

The OWCP classifier described above performs at 16.89 points above this simple baseline classifier, a 60% error reduction. Error analysis of the baseline classifier reveals that 928 of the true positives (exactly half) employ the part-of-speech tag "NN". Similarly, 876 of the false positives (68.1%) employ the part-of-speech tag "NN". This suggested that modifying the baseline by choosing only matching pairs of part-of-speech tags for certain tags will not be fruitful.

4.3.2 Comparison with Inter-Annotator Agreement

Positive-specific agreement was computed for the OWCPs in the BIOCC corpus. In the subset of BIOCC annotated by Elizabeth and myself, we annotated 238 and 230 OWCPs, respectively. We agreed on 214 which gives a positive-specific agreement of 94.69. In the subset of BIOCC annotated twice by myself I annotated 213 and 223 OWCPs, respectively on separate passes, and agreed with myself on 205 of them. This gives a positive-specific agreement of 95.79. These two subsets were subsequently adjudicated and were used as the holdout test set which contains 433 OWCPs. When

the best performing OWCP classifier is run on the BIOCC corpus, the performance as F-measure is 85.36. This performance improves to 86.63 when the full coordination resolution system (described in the next chapter) is evaluated because of an increase in recall. This represents an error rate that is about 2.5 times the disagreement rate for this data. This suggests that the OWCP classification task is far from a "solved" problem and would likely benefit from further research.

4.3.3 Comparison with Berkeley Parser

Section §5.1 describes a parser-based approach to coordination resolution that uses the Berkeley Parser. While this parser-based approach does not separate out OWCPs for separate classification, it nevertheless handles OWCPs as part of its parsing and thus performance on this task can be measured. Table 4.8 shows the performance of the Berkeley Parser applied to the OWCP classification task. The best performing configuration for the Berkeley Parser is, not surprisingly, when it is given POS_{gold} tags as input to the parser. However, it is not realistic to assume that gold-standard part-of-speech tags will be available for most use cases and so of greater interest is how the parser performs when it is given $\text{POS}_{\text{craft}}$ tags for input. The F-measure for the Berkeley Parser when it is trained on POS_{gold} tags and given $\text{POS}_{\text{craft}}$ tags is 83.24 which is 5.18 points lower than the best OWCP classifier.

Table 4.8: OWCP classifier compared with the Berkeley Parser. The first row repeats the best results from Table 4.6. The remaining rows show performance on the OWCP classification task using different combinations of part-of-speech tags for training and using the parser. For example, the row labeled "Berkeley (POS_{gold}, $\text{POS}_{\text{craft}}$)" indicates that the parser was trained with POS_{gold} tags and when it was run it was provided $\text{POS}_{\text{craft}}$ tags.

Features	TP	FP	FN	P	R	F	σ
lexical + LM + levsim	1779	205	261	89.67	87.21	88.42	4.21
Berkeley (POS_{gold}, POS_{gold})	1715	271	325	86.35	84.07	85.20	5.02
Berkeley (POS_{gold}, $\text{POS}_{\text{craft}}$)	1656	283	384	85.40	81.18	83.24	5.37
Berkeley ($\text{POS}_{\text{craft}}$, $\text{POS}_{\text{craft}}$)	1641	284	399	85.25	80.44	82.77	5.97

4.3.4 Effect of Separating the One-Word Conjunct Pair Classification Task

One of the motivating reasons for separating the OWCP classification task from the rest of the broader coordination resolution task was to improve overall system performance. The intuition is that by handling OWCPs separately, performance on those coordination structures would improve. Additionally, if downstream processing does not have to concern itself with these "easy" instances, then it can perform better by concentrating on only the remaining and more difficult instances. In fact, this does seem to be the case. In Table 5.4 results for a coordination resolution system that uses lexical and language model features using a linear SVM$^{\text{light}}$ model are given as 54.17% accuracy. This system was run with the best performing OWCP classifier given in Table 4.6 "upstream" of it. That is, the OWCP classifier was run and those conjunctions given a OWCP coordination structure by the OWCP classifier were not considered further by the full coordination resolution system. When this system is run without the OWCP classifier, the system must consider all conjunctions and the performance drops 2.05% to 52.12% accuracy. The drop in performance when the OWCP classifier is removed suggests that it is helpful when it is used.

4.3.5 Error Analysis

The errors that remain from the best performing OWCP classifier can be characterized in a number of ways. While no attempt was made to quantify the prevalence of the different kinds of errors being made, it is informative to present examples of some of the more frequent error types. Easily the most prevalent kind of error is caused by syntactic structural ambiguity related to e.g. noun modification or prepositional attachment. An example of the former is:

- . . . information on gene expression profiles of *whole brain* **and** *striatum*.

The classifier incorrectly classified the words "brain" and "striatum" as a OWCP. An example involving prepositional attachment ambiguity is:

- . . . highlighted a number of genes that influence *neuron proliferation* **and** *differentiation of the striatum and other neighboring forebrain structures*.

Here the classifier incorrectly classified the words “proliferation” and “differentiation” as a OWCP. Another class of errors involve errors (or at least ambiguities) in the gold-standard data. That is, there are cases where it appears the treebank data generates a poor coordination structure. The following is an example:

- . . . was chosen to minimize sampling variance by ensuring an equitable sampling of *striatal patch* and *matrix*.

Here the classifier incorrectly classified the words “patch” and “matrix” as a OWCP according to the gold-standard data. However, a OWCP may be a more appropriate coordination structure here than the one given by the gold-standard data.

4.4 Conclusions

In this chapter an OWCP classifier was presented that far out-performs a simple baseline classifier and is better than a state-of-the-art syntactic parser. Features that make use of lexical similarity and language modeling improve performance on this task. It was shown that separating this task out from the rest of the coordination resolution analysis overall system performance increases. Analysis of the remaining errors reveals that multiple kinds of syntactic structural ambiguity make this task difficult and that similar attention may need to be applied to these challenging phenomena in order to increase performance on this task.

Chapter 5

Full Coordination Resolution

In Chapter 4 a very specific subset of coordination structures were analyzed: one-word conjunct pairs. In this chapter the broader task of coordination resolution is explored for all of the coordination structures in the targeted corpora regardless of their size. A major contribution of this work is a novel yet simple algorithm for performing syntactic coordination resolution that does not employ deep syntactic parsing. This "parser-free" approach is compared and contrasted with the traditional approach of performing coordination resolution; employing a syntactic parser and extracting coordination structures from the resulting parse trees. The parser-free approach is shown to benefit from the language model features described in Chapter 3 and can perform better than the parser-based approach. Furthermore, the two approaches are shown to be complementary to each other and can be integrated to produce a higher performing system. All of the performance results reported in this chapter are from ten-fold cross-validation on the training portion of the CRAFT corpus.

5.1 Coordination Resolution using the Berkeley Parser

Syntactic parsers perform coordination resolution as an integrated part of their syntactic analysis. As such, performing this task with a state-of-the-art syntactic parser provides an important benchmark to compare other approaches against. The Berkeley Parser[43] was chosen for this task because of its competitive performance compared with other syntactic parsers, its ease of use,

and availability.[1] The Berkeley Parser is a constituent-based parser, i.e. it produces syntactic trees that consist of relationships between syntactic constituents such as noun phrases, verb phrases, and nominals, to name a few. The parser uses expectation maximization to learn probabilistic context-free grammars which can subsequently be used to assign the most likely parse tree to a sequence of words in a sentence. It excels at producing compact grammars that have very high accuracy.

Performing coordination resolution with the Berkeley Parser is fairly straightforward. The first step is to train a parsing model using the "GrammarTrainer" class provided by the Berkeley Parser library. The input to the trainer is a treebank file containing one sentence per line. The required format of treebank data is the Penn Treebank (PTB) format[2]. Because CRAFT is provided in PTB format, the Berkeley Parser can be trained directly on CRAFT data. The output from the trainer is a model which contains a learned grammar. The parser model, which is stored as a file, can be used to instantiate a "Parser" class object which can parse one sentence at a time. The input to the parser is a list of tokens and corresponding part-of-speech tags for a single sentence. The output of the parser is a full syntactic parse of the sentence. To produce coordination structures from the resulting syntactic parses, the coordination structure production algorithm described in Section §2.1.2 is used. The resulting coordination structures are then evaluated against gold-standard coordination structures.

Using the coordination structure production algorithm on the resulting parser trees may give a slight advantage to the parser-based approach over the parser-free approach described in section §5.3 because it is less vulnerable to possible mistakes in the coordination structure production algorithm. That is, if an error in the coordination structure production algorithm causes a mistake in the gold-standard data, then a perfect parse from the parser will produce matching gold-standard data because the same coordination structure production algorithm is applied. For example, during error analysis it was discovered that the conjunction in "... (Valius and Kazlauskas 1993)" was given the conjuncts "Valius" and "Kazlauskas 1993" in the gold-standard data as a result of a deficiency

[1] The Berkeley Parser is licensed under the General Public License and is available at `http://berkeleyparser.googlecode.com`

in the coordination structure production algorithm. The parser-free approach (described below) got this example wrong because it gave the conjuncts "Valius" and "Kazlauskas". The parser-based approach got this example right because it correctly parsed the parenthetical and so running the coordination structure production algorithm resulted in the same gold-standard coordination structure. After the coordination structure production algorithm was fixed to handle this case, both approaches got this example correct.

When the parser is trained with unmodified CRAFT treebank data, POS_{gold} tags are used. However, it is interesting to compare coordination resolution results when the parser is also trained using $\text{POS}_{\text{craft}}$ and $\text{POS}_{\text{genia}}$ tags (these tagger-generated tags are described in Section §3.1.1.) This is accomplished by modifying the CRAFT treebank files so that they contain those tags and then training the parser on the modified treebank file. Similarly, the parser can parse sentences using POS_{gold}, $\text{POS}_{\text{craft}}$, or $\text{POS}_{\text{genia}}$ tags. Table 5.1 provides the results of the Berkeley Parser on the coordination resolution task using ten-fold cross-validation on the CRAFT training data set. For each fold, the remaining nine folds were used to train the Berkeley Parser. Each row of the table gives a different configuration of part-of-speech tags used. The first column displays which part-of-speech tags were used to train the parser and the second column displays which part-of-speech tags were passed into the parser. For example, the first row gives the results when the parser is trained with POS_{gold} tags and POS_{gold} tags are again provided to the parser for parsing sentences. Not surprisingly, this configuration provides the best results. However, it is unrealistic to expect that gold-standard tags will be available in most usage scenarios and so of greater interest is how the parser performs when using tagger-generated tags. These results are shown in the remaining four rows. The highest performing configuration is 57.00% and is shown in the last row.

Table 5.1 also gives the failure rate of the parser as the percentage of sentences for which no parse was returned. A large failure rate will have a negative impact on accuracy because the parser will not get any coordination structures correct for any of the sentences it fails to parse. It is not surprising then, that the configuration with the highest failure rate (see the second row) has the worst performance. What is most striking about the results in Table 5.1 is the performance

Table 5.1: Performance of Berkeley parser on coordination resolution on the CRAFT corpus using different part-of-speech tags. The column labeled "Train Tags" gives the part-of-speech tags used for training the parser. The column labeled "Input Tags" gives the part-of-speech tags used at run-time that were given to the parser when parsing the sentences. The columns labeled "+" and "−" correspond to the number of coordination structures that were correct and incorrect, respectively. The column labeled "FR" gives the failure rate as the percentage of sentences for which no complete parse was returned by the parser. The column labeled "A" gives the accuracy of the parser on the coordination resolution task. The final column labeled "σ" gives the sample standard deviation of the file-level F-measure performance.

Train Tags	Input Tags	+	−	FR	A	σ
POS_{gold}	POS_{gold}	4277	2899	1.94%	59.60	6.00
POS_{gold}	POS_{genia}	2509	4667	26.35%	34.96	6.82
POS_{genia}	POS_{genia}	3613	3563	1.83%	50.35	6.46
POS_{gold}	POS_{craft}	4042	3134	3.73%	56.33	5.44
POS_{craft}	POS_{craft}	4090	3086	1.79%	**57.00**	5.51

disparity between using POS_{genia} and POS_{craft} tags as input to the parser. The large failure rate of 26.35% explains, at least in part, the poor performance (34.96% accuracy) when POS_{gold} tags are used for training the parser and POS_{genia} tags are used for parsing. This large failure rate is probably due to structural differences between texts in the GENIA and CRAFT corpora because the former consists of abstracts and the latter consists of full-text articles. Cohen et al.[11] studied structural differences between article abstracts and bodies and found, for example, that parenthetical materials are three times more frequent in article bodies than in the abstracts. Furthermore, the distribution of the types of parenthetical materials vary widely and are much more likely to contain items like citations, captions, and references to tables or figures. Such structural elements are probably not tagged correctly when POS_{genia} tags are used and this has a detrimental effect on the parser by increasing the failure rate. However, when the parser is trained using POS_{genia} tags, then the failure rate plummets to 1.83%. This drop in failure rate corresponds to a large gain in accuracy of 15.39 percentage points to 50.35%. However, this is still well below the performance of either configuration using POS_{craft} tags as input to the parser (5.98 and 6.65 percentage points, respectively.) These results highlight the sensitivity of the Berkeley Parser to part-of-speech tags that are used.

5.2 Coordination Resolution using a Dependency Parser

Another widely employed parsing paradigm, dependency parsing, requires syntactic trees whose structures consist of dependency relationships between individual words. Initial exploration with performing coordination resolution using a state-of-the-art dependency parser, the MaltParser described in [36], proved to be not very promising. The first challenge is to obtain suitable training data. Because two of the target corpora of this dissertation are provided in PTB format the data must first be converted from constituent-based syntactic trees to dependency structures. Unfortunately, the conversion from constituent-based trees to dependency-based trees is error prone. Also, converting dependency relationships to the coordination structures found in the gold-standard data proved to be much more complicated than converting the output of constituent-based parser as described in Section §2.1.2. For these reasons, full treatment to exploring a dependency parser based approach for coordination resolution was not given.

5.3 Parser-free Coordination Resolution

Every conjunction has at least two conjuncts: one on the left and one on the right of the conjunction. The token to the left of a conjunction, w_L, makes up the right border of the left conjunct and can be reliably determined. Similarly, the word to the right of a conjunction, w_R, makes up the left border of the right conjunct and is usually the word immediately adjacent to the right of the conjunction. As previously discussed in Section §4.1, w_L may not be the token immediately to the left of the conjunction if that token is a comma or semi-colon. Similarly, w_R may not be the token immediately to the right of the conjunction if that token corresponds to a word in a set of "trailing skip words" such as "then", "therefore", and "thus", among others. Since the right boundary of the left conjunct is given by w_L and the left boundary of the right conjunct is given by w_R, what remains is to find the correct left border of the left conjunct, find the correct right border of the right conjunct, and find any additional conjuncts that may exist to the left of the left conjunct. Framing the task in this way provides the intuition for the coordination resolution

algorithm described next.

5.3.1 Algorithm Description

The parser-free coordination resolution algorithm starts by first invoking a OWCP classifier as described in Chapter 4. If the classifier returns a prediction of true, then a OWCP is created as the coordination structure of the conjunction and it is done. If the classifier returns a prediction of false, then the following is performed. First a decision is made regarding whether to find the left boundary of the left conjunct or the right boundary of the right conjunct by choosing a side of the conjunction. Each token on the chosen side of the conjunction is evaluated as being a conjunct boundary or not by a binary classifier. There are two classifiers: one for classifying tokens on the left side and another for classifying tokens on the right side. Because the tokens are classified one at a time, it is possible for more than one token to be classified as a boundary token (i.e. the classifier returns a prediction of "yes" for more than one token.) ClearTK classifier wrappers may implement a score method which allows some numeric value to be associated with the classification prediction such that a higher score corresponds to a higher confidence in the classification prediction. So, the token with the highest scoring "yes" prediction will be chosen as the boundary of a conjunct. Once the boundary of a conjunct has been determined a conjunct annotation is created and added to the coordination structure for the conjunction. If none of the tokens are classified with a prediction of "yes", then the least confident "no" prediction is chosen as the conjunct boundary. The same logic is then applied to the tokens on the opposite side of the conjunction. After both the left and right conjuncts have been created, what remains to be done is determine if there are any additional conjuncts to the left of the left conjunct. This is accomplished by examining each token to the left of the left conjunct in reverse order (i.e. right to left) and classifying it as either a left boundary of an additional conjunct or not. If a token is classified as a left boundary of an additional conjunct, then a conjunct is created that spans from the classified token to the token adjacent to the left of the leftmost conjunct (excluding conjunct delimiter tokens.) In total there are three classifiers: one for finding the left boundary of the left conjunct, one for finding the right boundary of the right

conjunct, and another for finding left boundaries of additional left conjuncts.

Getting the first conjunct correct may be advantageous because it can be used for deriving features for determining the conjunct boundary on the opposite side of the conjunction. Presumably, an otherwise difficult to determine conjunct may be easier to identify if the conjunct on the other side of the conjunction is known. Therefore, the method of choosing which side to start on may have some impact on system performance. Experimentation was performed on three ways of choosing which side of the conjunction to start on. The first is to always choose the left-hand side first and the second is to always choose the right-hand side first. The third is to choose the side that has fewer tokens. The intuition for this third approach is that the side with fewer tokens will be easier to get correct than the side with more tokens because there are fewer options from which to choose the correct conjunct boundary. Performance of the parser-free approach was nearly identical between always choosing the left-hand side first and always choosing the right-hand side first. Choosing the side that has fewer tokens first gave a slight advantage (about .4 percentage points) and so this heuristic was used.

The parser-free approach described above is just one of many possible ways to tackle coordination resolution. Initial experimentation with two other approaches was also performed. One approach was to frame the problem as a chunking task using a "B-I-O" labeling scheme for identifying a token's role in a conjunct as beginning, inside, or outside a conjunct, respectively. The other approach performs pairwise classification by predicting the left boundary of the left conjunct and the right boundary of the right conjunct at the same time by looking at every possible combination of two words, one from each side of the conjunction. Neither of these approaches seemed promising from initial experimentation and so were not explored further.

5.3.2 Lexical Features for Parser-Free Approach

As before with the OWCP finder, a number of lexical features were employed for the classifiers used in the parser-free approach. The features used for the classifiers that find the left and right conjuncts are listed in Table 5.2. The classifier that finds additional conjuncts has an additional

feature corresponding to whether or not a conjunct delimiter (i.e. ',' or ';') precedes the leftmost conjunct and another feature that corresponds to whether or not a conjunct delimiter precedes the token being classified. The features involving POS tags are assumed to be POS_{craft} tags unless otherwise noted. The features involving the conjunction (features shown in Table 5.2 that use w_C) do not involve the word (w_i) which is being classified as a conjunct boundary and so are constant for each token that is considered. Using these conjunction-specific features was found to incrementally improve performance. While many different sets of lexical features were experimented with, thorough feature selection experimentation was not conducted for the parser-free approach.

Table 5.3 shows the performance of the parser-free approach on CRAFT using only lexical features for a number of different classifier libraries. The same classifiers that were used for OWCP classification as shown in Table 4.2 are used here except that LIBSVM is not compared here because its performance on the OWCP task was very similar to that of SVM^{light} and because the ClearTK wrapper for this library does not provide a score method which the algorithm requires. Instead, LIBLINEAR[14] is used because the ClearTK wrapper does provide a score method. In fact, LIBLINEAR was the classifier library used in the top performing coordination resolution system reported in this chapter and is also quite fast for both training and classification. Results for Mallet's MCMaxEnt learner are not provided because it failed to converge on the second fold. As before with the OWCP classifier, the choice of classifier library is quite important to the performance of the parser-free algorithm. The lowest performing learner, Naïve Bayes, is nearly 15 percentage points below the best performing learner, SVM^{light} which performs at 54.03% accuracy. It is interesting to note that the parser-free approach using only lexical features and the SVM^{light} learner using a polynomial kernel is just shy of three percent less than the parser-based approach which has an accuracy of 57.00.

5.3.3 Language Model Features

As before with the OWCP finder, the rank percentile produced by the language model as described in Section §3.2 is added as a feature to the OWCP classifier and each of the three

Table 5.2: Lexical features for the parser-free coordination resolution approach. The label w_i refers to the word being classified, w_C refers to the conjunction, w_{cf} and w_{cl} refer to the first and last word, respectively, of the conjunct on the opposite side of the conjunction (if available), and w_{ao} refers to the word that is adjacent to the conjunction on the opposite side of the conjunction from w_i - i.e. w_{ao} will either be w_L or w_R depending on which side of the conjunction w_i is on.

feature name	feature taken from
word	w_i, w_{i-1}, w_{i-2}, w_{i-3}, w_{i+1}, w_{i+2}, w_{i+3} w_C, w_{C-1}, w_{C-2}, w_{C-3}, w_{C+1}, w_{C+2}, w_{C+3}
POS tag	w_i, w_{i-1}, w_{i-2}, w_{i-3}, w_{i+1}, w_{i+2}, w_{i+3} w_C, w_{C-1}, w_{C-2}, w_{C-3}, w_{C+1}, w_{C+2}, w_{C+3}
distance	w_i, w_C
3 character suffix	w_i
3 character prefix	w_i
capital type	w_i
contains hyphen	w_i
numeric type	w_i
lower case word	w_i
word bigram	$w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$, w_iw_{cf}, w_iw_{cl}, w_iw_{ao} $w_{C-2}w_{C-1}$, $w_{C-1}w_C$, w_Cw_{C+1}, $w_{C+1}w_{C+2}$
POS bigram	$w_{i-2}w_{i-1}$, $w_{i-1}w_i$, w_iw_{i+1}, $w_{i+1}w_{i+2}$, w_iw_{cf}, w_iw_{cl}, w_iw_{ao} $w_{C-2}w_{C-1}$, $w_{C-1}w_C$, w_Cw_{C+1}, $w_{C+1}w_{C+2}$
stem bigram	w_iw_{cf}, w_iw_{cl}, w_iw_{ao}

Table 5.3: Performance of the parser-free coordination resolution system using only lexical features. The final row shows the best performing configuration using the Berkeley Parser from Table 5.1 that uses part-of-speech tags from a tagger.

Classifier	+	−	A	σ
Naïve Bayes (Mallet)	2818	4358	39.27	5.43
MaxEnt (OpenNLP)	3143	4033	43.80	5.94
SVM$^{\text{light}}$ (linear)	3716	3460	51.78	5.79
MaxEnt (Mallet)	3749	3427	52.24	6.04
LIBLINEAR	3857	3319	53.75	6.22
SVM$^{\text{light}}$ (polynomial)	3877	3299	**54.03**	5.70
Parser-Based	4090	3086	57.00	5.51

conjunct boundary classifiers described above. In addition to adding the language model feature to these classifiers the language model is also employed to directly find conjuncts with high precision. As described in Section §3.2 a probability is assigned by the language model to each candidate sentence corresponding to every candidate conjunct border for a given side of the conjunction. These probabilities tend to be vanishingly small and are only useful when compared relative to each other which is why the rank percentile is used as a feature instead of the probability itself. However, another way to compare the probabilities generated from the language model is to normalize them such that they add to one. These normalized probabilities were also added as features to the classifiers but were not found to be as effective as the rank percentiles and so they are not used for this purpose. However, they do provide a mechanism to find conjuncts with very high precision. For example, if the normalized probability of the highest ranking candidate sentence is 95% or higher, then the corresponding conjunct is correct 88.55% of the time with 3.24% recall. This high precision language model conjunct finder is run directly after the OWCP classifier. The best performing configuration creates conjuncts when the the highest ranking candidate conjunct boundary corresponds to a single word conjunct and has normalized probability greater than 45%. This language model conjunct finder has a precision of 90.10% and a recall of 30.84% for one-word conjuncts. If a conjunct exists on either side of the conjunction, then the parser-free approach simply uses that conjunct and proceeds to the opposite side of the conjunction if necessary.

Table 5.4 gives the performance of the parser-free approach using the language modeling features and the high-precision language model conjunct finder. In each case, the performance increases from 2 to 3 percentage points which is shown in the column labeled "Δ". The highest performing classifier library shown in Table 5.4 is again the SVM^{light} classifier with an accuracy of 56.59 which is just .41 percentage points less than the parser-based approach which has an accuracy of 57.00.

Table 5.4: Performance of the parser-free coordination resolution system using lexical and language model features. The final column gives the difference as percentage points of the previous column with the corresponding results given in Table 5.3.

Classifier	+	−	A	Δ	σ
SVM^{light} (linear)	3887	3289	54.17	2.39	5.27
MaxEnt (Mallet)	3957	3219	55.14	2.90	4.59
LIBLINEAR	4000	3176	55.74	1.99	4.93
SVM^{light} (polynomial)	4061	3155	**56.59**	2.56	5.05

Table 5.5 shows the performance of the parser-free approach by coordination type using the SVM^{light} classifier library. Coordination types are discussed in Section §2.1.3. The performance on every node type excepting OTHER improves when the language model features are applied. However, stratifying the system's performance by coordination type reveals large performance disparities. For example, the highest performing coordination type, ADJP, performs at an accuracy of 77.69 which is 54.02 percentage points higher than the worst performing coordination type, S, whose accuracy is 23.67. This suggests that the parser-free approach simply doesn't work very well for certain types of coordination structures. One possible explanation for the performance disparity is that the parser-free approach is doing a better job of learning coordination types with short conjuncts. The last column of Table 5.5 provides the average conjunct length in words for each type. The average length correlates quite well with coordination resolution performance. In fact, if the types UCP and OTHER are excluded, then the average conjunct length almost perfectly predicts performance rank. For example, the type with the shortest conjuncts, ADJP, performs the best while the type with the longest conjuncts, S, performs the worst. One possible explanation

for this phenomena is the misuse of the distance feature in the classifier models. The distance feature's value is the distance in words of the word being classified (i.e. w_i) and the conjunction. The intuition is that by learning all coordination types with the same set of classifiers the distance feature will generally favor words that are closer to the conjunction than words that are further away because, on average across all types, conjuncts are short. This will adversely effect the correct selection of words that are conjunct boundaries for coordination structures of type S and VP because these conjuncts are, on average, much longer. As expected, when this feature is removed system performance on shorter coordination types (e.g. ADJP, NML, and NP) worsens while performance on longer coordination types (e.g. S and VP) improves. However, large disparities in performance between coordination types remain, e.g. even though performance on ADJP worsens and S improves when the distance feature is removed, the accuracy for the former is still much higher than that of the latter. This suggests a broader problem with the parser-free approach; it can not handle some coordination types very well.

Table 5.5: Performance of the parser-free coordination resolution approach stratified by type when run with SVM^{light} using a polynomial kernel. The column labeled "Type" refers to the type of the coordination structure derived from the CRAFT tree from which it was obtained. The column labeled "Count" gives the total number of coordination structures of that type are found in the CRAFT training set. The column labeled "Lexical" shows the performance as accuracy for a given type when the parser-free system is trained on only the lexical features given in Table 5.2. Similarly, the column labeled "LM" shows the performance as accuracy for a given type when the parser-free system is trained on both the lexical features and the language model features. The column labeled "Δ" gives the positive difference between the preceeding two columns. The final column labeled "Len." gives the average length (as the number of words) of conjuncts for the given type.

Type	Count	Lexical	LM	Δ	Len.
S	528	21.40	23.67	2.27	14.14
UCP	203	39.90	47.29	7.39	2.57
PP	206	48.00	49.03	1.03	8.53
VP	1124	47.69	50.00	2.31	8.61
NP	2976	56.00	57.80	1.80	4.17
NML	1388	60.23	64.70	4.47	1.96
OTHER	258	68.68	68.22	-0.46	4.36
ADJP	493	73.63	77.69	4.06	1.77
Total	7176	54.03	56.59	2.56	5.04

5.3.4 Type-Specific Conjunct Boundary Classifiers

A broader explanation for the performance disparities between different types could be that trying to learn conjunct boundaries for all coordination types with a single set of classifiers (one for each of the left conjunct, the right conjunct, and additional conjuncts) is simply too confusing for the classifiers. One way to test this hypothesis is to train the parser-free coordination resolution system on specific coordination types and observe the performance for the given type. This exercise points towards an oracle-based system in which an oracle gives the system the correct coordination type before attempting to find conjunct boundaries. This allows the system to triage to a set of conjunct boundary classifiers that performs the best on the provided coordination type. For example, if the oracle says that the type of a coordination structure for a given conjunction is ADJP, then the system can use the classifiers trained on the coordination structures of type ADJP. For this oracle-based system classifiers were trained for the following types; ADJP, NML, NP, PP, S, SBAR (one of the types grouped under OTHER), and VP. As expected, the highest performing system for each of these types was the corresponding set of classifiers trained on that type. For the remaining types, UCP and the the remaining types grouped under OTHER, a set of classifiers was chosen for each type that performed the best on that type out of all the systems built including the systems given in Table 5.4 and Table 5.8. Table 5.6 gives the results of this oracle-based system when all sets of classifiers are built with LIBLINEAR. These results are compared with the performance of the parser-free system trained with LIBLINEAR given in Table 5.4. LIBLINEAR is used here because the oracle system using LIBLINEAR classifiers has higher results (by one percentage point) than for SVM$^{\text{light}}$.

Table 5.6 demonstrates that knowing the coordination structure type in advance gives the system a large advantage. It also demonstrates that the parser-free approach is not necessarily ill-suited for certain kinds of coordination types like S and VP. In fact, the oracle-based system performs at 60.61 for the S type which is higher than the accuracy of 59.95 for the NP type. Of course, having the luxury of an oracle that provides the coordination type is nice but, unfortunately,

Table 5.6: Performance of the parser-free approach when an oracle provides the coordination type allowing the system to triage to a set of conjunct boundary classifiers for which the type is best suited. The column labeled "LM" gives the type-specific performance for the parser-free approach trained with lexical and language model features using the LIBLINEAR model. The column labeled "Oracle" gives the performance of a system aided by an oracle that provides the correct type of the coordination structure. The column labeled "Δ" gives the positive difference between the previous two columns.

Type	Count	LM	Oracle	Δ	Len.
S	528	7.95	60.61	52.66	14.14
VP	1124	45.28	56.14	10.86	8.61
PP	206	47.09	47.09	0.00	8.53
UCP	203	51.23	56.65	5.42	2.57
NP	2976	57.96	59.95	1.99	4.17
OTHER	258	67.44	81.01	13.57	4.36
NML	1388	68.73	74.21	5.48	1.96
ADJP	493	80.12	82.35	2.23	1.77
Total	7176	55.74	63.99	8.25	5.04

unrealistic to expect. This raises the question of how well can the coordination type be predicted? To answer this question, a classifier was built that predicts the coordination type given a conjunction and features derived from the surrounding context. The features used are very similar to those used for the OWCP classifier and so will not be repeated here. The best performing classifier was a LIBSVM classifier using a polynomial kernel which has an overall performance 74.95% as F-measure. Table 5.7 shows the F-measure of the type classifier for each type. The types grouped together as OTHER were treated as separate types and the results are aggregated together for compactness. One promising result from this table is the high performance on the VP type. This suggests that using the classifier for providing the coordination type may compare favorably with using the oracle. Unfortunately, the type classifier's performance on the type S, which would likely benefit the system the most to know, is relatively poorly. Generally speaking, types that occur less frequently (e.g. S, SBAR, and ADVP) perform with higher precision than recall. Correspondingly, the type that occurs the most often, NP, has much higher recall (86.69) than precision (71.93). As such, many instances are confused as NP's. For example, 526 (or 37.90%) of NMLs are confused as NPs.

Table 5.7: Performance of the coordination type classifier by type.

Type	Count	P	R	F
ADJP	493	86.98	81.34	84.07
NML	1388	66.69	58.72	62.45
NP	2976	71.93	86.69	78.62
PP	206	70.16	87.86	78.02
S	528	75.17	41.86	53.77
UCP	203	57.61	26.11	35.93
VP	1124	89.92	87.92	88.91
OTHER	258	84.48	54.65	66.37
Total	7176	74.95	74.95	74.95

The parser-free algorithm was expanded to leverage the type classifier introduced above. For each conjunction, the type classifier determines the coordination type of the corresponding coordination structure. This result is used to triage coordination resolution to a set of conjunct boundary classifiers trained on the type predicted by the type classifier. Two ways of training this augmented system were attempted. The first uses the gold-standard coordination types to determine which set of type-specific conjunct boundary classifiers a training instance should be sent to. The second uses the results of the type classifier to determine which set of type-specific conjunct boundary classifiers a training instance should be sent to. Table 5.8 gives the results for the type-classifier-augmented parser-free system using both training strategies with the SVM$^{\text{light}}$ and LIBLINEAR classifiers. The top performing system that uses LIBLINEAR and the type classifier generated types for training has an accuracy of 58.33% which is 1.33 percentage points higher than the parser-based approach which performs at 57.00% accuracy. For this top performing system, performance on the type VP improved substantially to 52.27% accuracy from 45.28% which is 3.87 points shy of the 56.14% performance given by the oracle-based system whose results are reported in Table 5.6. Unfortunately, this system still performed very poorly on the type S with an accuracy of 27.08% for the best performing system given in Table 5.8 which is far below the oracle enhanced system reported in Table 5.6 of 60.61%. These two results together suggest that high type classification accuracy is a key to improved performance using the type classifier and that

further work on the type classifier may prove fruitful.

Table 5.8: Performance of parser-free approach using type-specific classifiers. The second column labeled "Types" provides the types that were used to determine which training examples were sent to type-specific classifiers for learning. The value "gold" means that a coordination structure's gold-standard coordination type was used. The value "system" means that the type assigned by the type classifier was used.

Classifier	Types	+	−	A	σ
SVM$^{\text{light}}$	gold	4144	3032	57.75	5.03
SVM$^{\text{light}}$	system	4181	2995	58.26	5.27
LIBLINEAR	gold	4169	3007	58.10	5.22
LIBLINEAR	system	4186	2990	**58.33**	5.28
LIBLINEAR $\text{POS}_{\text{genia}}$	system	4110	3066	57.27	5.70

The final row of Table 5.8 shows how the system works when trained and tested using $\text{POS}_{\text{genia}}$ tags instead of $\text{POS}_{\text{craft}}$ tags. While the accuracy goes down to 57.27% this is considerably higher than the accuracy of the Berkeley Parser based approach when it uses $\text{POS}_{\text{genia}}$ tags which is 50.35%. This result indicates that the parser-free approach is much less sensitive to the part-of-speech tags that are used compared to the parser-based approach.

5.4 Integrated Coordination Resolution

Given the very similar overall performances of the Berkeley Parser based approach described in Section §5.1 and the parser-free approach described in Section §5.3, one might think that the two approaches are learning to correctly resolve the same coordination structure instances. This would be a disappointing result because it would suggest that while the two approaches are conceptually quite different they are ultimately doing essentially the same thing at about the same performance. Additionally, this would mean that there would be no way to combine the two approaches to create an integrated system that performs better. Fortunately, this is not the case as there are many true positives given by both systems which are not found by the other. Table 5.9 shows the breakdown of conjunctions that are correctly resolved by these two systems. The parser-based system used here is the best performing system shown in Table 5.1 using $\text{POS}_{\text{craft}}$ tags. The parser-free system

used here is the best performing system shown in Table 5.8 which uses LIBLINEAR. There are 5,183 coordination structures that were correctly resolved by at least one of the systems of the 7,176 conjunctions in the training data. Therefore, an integrated system could achieve an accuracy of 72.73% if an oracle was available to choose which system should provide coordination structure for a given conjunction. This is a dramatic performance gain over either system by itself and suggests that it may be possible to automatically integrate these two systems. In contrast, a comparison of the results generated by the two best parser-free systems from Table 5.8 (one using $\text{SVM}^{\text{light}}$ and the other LIBLINEAR) shows that only 4,559 (or 63.53%) of the coordination structures were correctly resolved by at least one of these two systems.

Table 5.9: The distribution of correctly resolved coordination structures between the parser-based approach and the parser-free approach. Each coordination structure in the training set is counted once in one of the first four rows. If both systems correctly resolved the coordination structure then it is counted as "Both correct". If only the parser-based approach got it right, then it is counted as "Parser-based only". If only the parser-free approach got it right, then it is counted as "Parser-free only". Otherwise, it is counted as "Neither correct" because neither system got it right.

Both correct	3,109	43.32%
Parser-based only	981	13.67%
Parser-free only	1,077	15.01%
Neither correct	2009	28.00%
total	7,176	100%

The results in Table 5.1 are reported using accuracy because each conjunction can be counted as correct or incorrect as discussed in Section §2.4. This makes perfect sense for the parser-free approach introduced in Section §5.3 because a coordination structure is proposed for each conjunction it processes. However, for the parser-based approach, using accuracy is somewhat problematic because for the sentences that fail to parse, no coordination structures are produced. These instances can be considered as false negatives but not false positives which means that precision and recall will not be equivalent, i.e. the former is higher than the latter. This means that for the Berkeley Parser, F-measure will be higher than accuracy (which is calculated the same as recall here.) For example, when POS_{gold} tags are used for training the parser and $\text{POS}_{\text{craft}}$ tags are

used for parsing, the precision is 59.69, the recall is 56.33, and the F-measure is 57.96. The higher precision of the parser-based approach will be exploited when the two approaches are integrated in this section.

One simple way to integrate these two systems is to accept coordination structures that are produced by the Berkeley Parser for types that it performs better on and have the parser-free approach handle the remaining coordination structures. Because coordination structures produced by the Berkeley Parser are derived from a full syntactic parse, coordination types (i.e. the type of node that is the parent of the conjunction) are also predicted by the parser. Coordination structures that are not assigned a type by the parser that the parser performs better on are dismissed and the conjunction is passed on to the parser-free approach. Again, because the parser-based approach always has higher precision than recall, the precision of the parser-based approach is compared with the accuracy of the parser-free approach when choosing which types the Berkeley Parser should handle. Table 5.10 shows for each type the precision of the parser-based approach compared with the accuracy of the parser-free approach. By adding up the true positives predicted by the higher performing system for each type a simplistic prediction of the accuracy of the proposed integrated system gives 60.81%.

The integrated system implemented here works as follows. First, the OWCP classifier classifies each conjunction. If it returns a preciction of "true", then a OWCP coordination structure is created and that coordination structure is complete and no further processing is necessary. Next, the Berkeley Parser performs coordination resolution. If the coordination type given by the parser is one of the following types, then the coordination structure is kept otherwise it is discarded: ADVP, CIT, LST, PP, PRN, S, SBAR, SQ, UCP, VP, X. Next, the high-precision language model conjunct finder is run. Next, the parser-free approach is run on any remaining conjunctions for which there are no coordination structures. Table 5.11 shows the results of the top performing integrated systems. The performance of 60.91% accuracy is the highest accuracy reported in this chapter on the CRAFT training data set and is 3.91 and 2.58 percentage points higher than the best performing parser-based and parser-free systems, respectively. The integrated system is 11.62

Table 5.10: Performance of the parser-based and parser-free coordination resolution approaches by type. The column labeled "PB (P)" gives type-specific performance as precision for the parser-based system trained with POS_{gold} tags and run with $\text{POS}_{\text{craft}}$ tags. The column labeled "PF (A)" gives the type-specific performance as accuracy for the parser-free system that gave the highest results in Table 5.8. The column labeled "I (A)" gives the performance of the best-performing integrated system in Table 5.11.

Type	Count	PB (P)	PF (A)	I (A)
PP	206	**48.40**	42.72	49.51
UCP	203	**56.08**	48.28	54.19
VP	1124	**55.00**	52.27	54.45
NP	2976	56.11	**57.44**	56.12
S	528	**64.39**	27.08	61.93
NML	1388	61.80	**70.05**	69.33
ADJP	493	76.86	**79.11**	77.28
OTHER	258	**80.05**	77.28	80.23
Total	7176	57.00	58.33	60.91

percentage points less than the oracle-based system in which the oracle chooses for each conjunction the system that gets the corresponding coordination structure correct if one exists. This result suggests that there is room for further improvement using an integrated approach for coordination resolution.

Table 5.11: Performance of the integration of the parser-based and parser-free approaches compared with an oracle-based system. Each system integrates the Berkeley Parser trained on POS_{gold} tags with parsing performed with $\text{POS}_{\text{craft}}$ tags. The parser-free systems are those reported in the first four rows of Table 5.8. For example the system labeled "Integration 4" uses LIBLINEAR, is augmented with a type classifier, and the type-specific classifiers were given coordination structures based on the types that were predicted by the type classifier.

System	+	−	A	σ
Integration 1	4314	2862	60.12%	5.40
Integration 2	4338	2838	60.45%	5.55
Integration 3	4339	2837	60.47%	5.30
Integration 4	4371	2805	**60.91%**	5.34

Table 5.10 shows the performance of the best performing integrated system for each coordination type in the final column. Because there is no way to know which system will actually end up handling a particular conjunction the respective performances of the two systems cannot reliably

predict the performance of the integrated system. For the ADJP type, the accuracy of 77.28% is lower than the performance given by the parser-free system given in Table 5.10 of 79.11% but slightly higher than the parser-based approach. This trend is continued for the NML type except that the performance is much closer to the parser-free system rather than the parser-based system. The types NP and PP performed lower than either system in isolation. The performance of type S of 61.93 for the integrated system was much closer to the parser-based system (64.39) than the parser-free system (27.08). This fortunate result largely explains the improvement of the integrated system over the parser-free approach which performs much lower for this type.

5.5 Conclusions

In this chapter parser-based, parser-free, and integrated coordination resolution systems were presented and they were compared on the CRAFT training data set using ten-fold cross-validation. The parser-free approach benefits from the same language model features used for the OWCP classifiers described in Chapter 4. Additionally, a coordination type classifier can be employed to further improve performance by facilitating the training and selection of sets of classifiers that are trained for specific coordination types. With these two improvements, the parser-free approach out performs the parser-based approach on the CRAFT training data set. Furthermore, when $\text{POS}_{\text{genia}}$ tags are used the performance gap between the two approaches gets much larger, i.e. the parser-free approach works much better than the parser-based approach. In the next chapter, these two approaches are compared on the CRAFT holdout evaluation, the BIOCC corpus, and the GENIA corpus.

Chapter 6

Results

In Chapter 5 the parser-based and parser-free approaches were described and compared using ten-fold cross-validation using the training portion of the CRAFT corpus. In this chapter, both approaches are evaluated against the CRAFT holdout evaluation test set, the BIOCC corpus, and the GENIA corpus. The holdout evaluation test set represents data from the CRAFT corpus that is "unseen". That is, there was no error analysis performed using this portion of CRAFT while developing the two approaches and so performance results may better reflect the true performance of the two approaches. The BIOCC corpus, described in Section §2.2, represents sentences found "in the wild" where no gold-standard tokens or sentences are given and there is no treebank data for which to train a parser. Additionally, the sentences in BIOCC were chosen at random from a very large corpus of biomedical texts and as such should better represent "average" sentences in the biomedical domain. The coordination resolution results on BIOCC, therefore, may present the most realistic performance that can be expected when these two approaches are applied to any sentences found in the biomedical literature. The parser-free approach is shown to outperform the parser-based approach on this corpus. Evaluation of both approaches is performed on the Hara subset of the GENIA corpus (see Section §2.3) which provides a direct comparison with the previous state-of-the-art in biomedical coordination resolution. These results show that both the parser-based approach using the Berkeley Parser and, to a greater extent, the parser-free approach outperform the work by Hara et al.[18] which is described in Section §1.1.4.

6.1 Results on CRAFT Test Set

As described in Section §2.1, the forty-six articles in the CRAFT corpus were split into a training set consisting of thirty-six files and a test set consisting of the remaining ten files. This test set was not used in any way to develop the two approaches described in Chapter 5. For example, all error analysis that informed improvements to the two approaches was conducted exclusively on results obtained from ten-fold cross-validation on the thirty-six files in the training set. Similarly, the configurations of the two approaches that were evaluated and reported here were chosen based solely on the performance results obtained during ten-fold cross-validation. Therefore, the results reported in this section may give a more realistic measure of how well these two approaches could be expected to perform on other full-text articles that meet the selection criteria for inclusion in the CRAFT corpus.

Table 6.2 shows the performance of a number of coordination resolution system configurations that are described in Chapter 5. The reported systems are listed in Table 6.1. In general, the results on the CRAFT holdout evaluation test set are consistent with the results reported in Chapter 5 using cross-validation on the training portion of the CRAFT corpus. The final column in Table 6.2 shows the performance difference between the holdout evaluation and cross-validation. Nearly every row (excepting one) shows a drop in performance on the holdout evaluation data. This result suggests that the coordination resolution systems are overfitting to the training data set. However, this result may instead be a reflection of the fact that the coordination structures in the CRAFT corpus are not i.i.d. as discussed in Section §2.1.1 and, therefore, differences in performance (in this case, negative) should be expected because, possibly, the ten files in the holdout data set are more difficult.

One mildly disappointing finding is that the positive impact of using the language model features was less for the holdout evaluation than it was for cross-validation. For example, the performance gain between LIBLINEAR-1 and LIBLINEAR-2 is 1.23% for the holdout evaluation where for cross-validation it was 1.99% (as shown in Table 5.4.) Similarly, the performance gain

Table 6.1: Listing of coordination resolution systems whose results are reported in this chapter. The untitled second column will have one of three values: 'PB' for parser-based approach, 'PF' for parser-free approach, and 'IN' for an integration of the parser-based and parser-free approaches. The third column titled "CV Results" provides a reference to the table and row from Chapter 5 that displays performance of the system using cross-validation on the CRAFT training data set.

System Name		CV Results	Description
Berkeley-1	PB	Table 5.1, row 4	Trained with POS_{gold} tags and tested with POS_{craft} tags.
Berkeley-2	PB	Table 5.1, row 5	Trained with POS_{craft} tags and tested with POS_{craft} tags.
SVM-1	PF	Table 5.3, row 6	SVM^{light} (polynomial) + lexical features
SVM-2	PF	Table 5.4, row 4	SVM^{light} (polynomial) + lexical and language model features
SVM-3	PF	Table 5.8, row 1	SVM-2 using type-specific models trained with gold coordination types.
SVM-4	PF	Table 5.8, row 2	SVM-2 using type-specific models trained with classifier-generated types.
LIBLINEAR-1	PF	Table 5.3, row 5	LIBLINEAR + lexical features
LIBLINEAR-2	PF	Table 5.4, row 3	LIBLINEAR + lexical and language model features
LIBLINEAR-3	PF	Table 5.8, row 3	LIBLINEAR-2 using type-specific models trained with gold coordination types.
LIBLINEAR-4	PF	Table 5.8, row 4	LIBLINEAR-2 using type-specific models trained with classifier-generated types.
Integration-1	IN	Table 5.11, row 1	SVM-3 & Berkeley-1 integrated
Integration-2	IN	Table 5.11, row 2	SVM-4 & Berkeley-1 integrated
Integration-3	IN	Table 5.11, row 3	LIBLINEAR-3 & Berkeley-1 integrated
Integration-4	IN	Table 5.11, row 4	LIBLINEAR-4 & Berkeley-1 integrated

Table 6.2: Performance of the parser-based and parser-free approaches on the CRAFT holdout evaluation set. The column labeled 'Δ' gives the performance difference in absolute percentage points between the performance reported in the column to the left labeled "A" with the performance reported in Chapter 5 using cross-validation.

System	+	−	A	Δ	σ
Berkeley-1	1065	804	56.98	0.65	4.38
Berkeley-2	1052	817	56.29	-0.71	8.58
SVM-1	992	877	53.08	-0.95	6.84
SVM-2	1025	844	54.84	-1.75	6.37
SVM-3	1054	815	56.39	-1.36	7.49
SVM-4	1044	825	55.86	-2.42	6.25
LIBLINEAR-1	984	885	52.65	-1.10	6.67
LIBLINEAR-2	1007	862	53.88	-1.86	7.42
LIBLINEAR-3	1062	807	56.82	-1.28	7.14
LIBLINEAR-4	1068	801	57.14	-1.19	6.56
Integration-1	1118	751	59.82	-0.30	6.33
Integration-2	1106	763	59.18	-1.27	5.01
Integration-3	1119	750	**59.87**	-0.60	6.63
Integration-4	1117	752	59.76	-1.15	6.35

between SVM-1 and 2 is 1.76% for the holdout evaluation compared with 2.56% for cross-validation. Another mildly disappointing finding is that performing coordination resolution with type-specific classifiers as discussed in Section §5.3.4 gives an additional performance gain that is less than or matches the performance of the parser-based approach rather than surpassing it as it did for cross-validation. For example, SVM-4 outperformed Berkeley-1 by 1.95% in the cross-validation but was 1.12% worse for the holdout evaluation. Similarly, LIBLINEAR-4 outperformed Berkeley-1 by 2.00% in the cross-validation but was only 0.16% better for the holdout evaluation. This result is somewhat surprising because the accuracy of the coordination type classifier was no worse on the test data than it was on the training data. In fact, it actually improved incrementally from 74.95 (as shown in Table 5.7) to 75.29. These results together suggest that the parser-based approach using type-specific classifiers and the parser-free approach perform at roughly the same level overall on the CRAFT corpus.

Integration of the two approaches provides additional gains when evaluated on the test set that are similar to the results on cross-validation. The accuracy of the best-performing integrated system, 59.87%, on the holdout evaluation data set is about one percent less that reported for cross-validation of 60.91%. The performance gain over either approach individually was about 3.0% in both cases (2.89% and 2.73% for the parser-based and parser-free approaches, respectively.)

6.1.1 Error Analysis

Table 6.3 shows coordination resolution performance results by the number of conjuncts in a coordination structure. Each system performs better on coordination structures that have only two conjuncts. This is not a surprising result given the fact that 86.8% of the training examples have only two conjuncts[1] because there is a lot more training data for left and right conjuncts than there is for additional conjuncts to the left of the left conjunct. However, it is disappointing that performance declines with the number of conjuncts because inter-annotator

[1] The proportion of two-conjunct coordination structures happens to be nearly identical (86.82 vs. 86.78) for both the training and testing data.

agreement actually increases for the BIOCC corpus when there are four or more conjuncts. The higher IAA is presumably due to the fact that the structure of coordination structures containing four or more conjuncts are easier to recognize because there exists a readily recognizable pattern of items in a list that make up these coordination structures. Automatic coordination resolution performance on these structures is likely worse because there are far fewer training examples. However, examination of a few errors by the "Integration-4" system reveal how easy it is to get very close to the correct coordination structure but still get counted as wrong because of the exacting evaluation criteria which requires all conjuncts to be correct (see Section §2.4 for more details on evaluation.) For example, consider the following three examples shown respectively with the gold-standard coordination structure followed by the erroneous coordination structure produced by "Integration-4":

- ... the [*extraembryonic mesoderm*], [*allantois*] **and** [*chorion*] ...
- ... [*the extraembryonic mesoderm*], [*allantois*] **and** [*chorion*] ...
- ... [*XRCC1*], [*ERCC5*], [*GSTP1*], ..., [*CAT*] **and** [*ERCC2*] ...
- ... XRCC1, [*ERCC5*], [*GSTP1*], ..., [*CAT*] **and** [*ERCC2*] ...
- ... [*serum glucose*], [*fatty acid*], [*organic acid*], **and** [*carnitine*] experiments ...
- ... [*serum glucose*], [*fatty acid*], [*organic acid*], **and** [*carnitine experiments*] ...

In the first example, the word "the" is erroneously included in the first conjunct. In the second example, the conjunct "XRCC1" is ommitted. In fact, there are a large number of examples in which a coordination structure consisting of three or more conjuncts in the gold-standard data is missing the first conjunct in the corresponding coordination structure generated by a system but is otherwise correct. In the third example, the word "experiments" is erroneously included in the final conjunct. Each of these examples represent coordination structures that were evaluated as wrong, and therefore no credit was given, even though in each case it is clear that they are nearly correct.

This observation suggests that it might be informative to consider evaluating coordination resolution at the conjunct level which would allow for partial credit for examples like these. Conjunct-level evaluation is addressed in Section §6.1.1.1. Examples of coordination structures containing three or more conjuncts for which the "Integration-4" system was not close to the gold-standard, i.e. it did not get two or more conjuncts correct, are quite rare.

Table 6.3: Performance of coordination resolution by number of conjuncts. The column titled "Conjuncts" gives the number of conjuncts in the evaluated coordination structures for that row. The column titled "Count" gives the number of coordination structures with the given number of conjuncts. The column labeled "PB" gives the performance of the parser-based system labeled "Berkeley-1" given in Table 6.1, "PF" the parser-free system labeled "LIBLINEAR-4", and "IN" the integrated system labeled "Integration-4."

Conjuncts	Count	PB	PF	IN
2	1,622	57.71	58.69	61.34
3	167	53.29	47.31	53.89
4	38	50.00	44.74	52.63
5	19	52.63	47.37	52.63
6+	23	47.83	47.83	47.83
Total	1,869	56.98	57.14	60.25

Table 6.4 shows coordination resolution results for the different types of coordination structures on the holdout test set for the CRAFT corpus. Overall, these results are quite similar to what was seen on the training set using cross-validation as shown in Table 5.10. However, the differences between the precision of the parser-based and the accuracy of the parser-free approach for any given type have changed. For example, in Table 5.10 the precision of the parser-based approach was 61.80 for the type "NML" while the accuracy of the parser-free approach was 70.05 giving a difference of 8.25. Here the difference is only 2.05. For the type "PP" the higher performing system actually changed from the parser-based to the parser-free approach. This means that a suboptimal set of types to be handled by the parser-based approach when running the integrated system was chosen. When "Integration-3" and "Integration-4" are run such that coordination structures of type "PP" from the Berkeley Parser are not used the performance sees a small positive bump to 60.25 for both systems from 59.87 and 59.76, respectively.

Table 6.4: Performance of the parser-based, parser-free, and integrated coordination resolution approaches by type. The columns labeled "PB (A)" and "PB (P)" gives type-specific performance as accuracy and precision, respectively, for the parser-based system labeled "Berkeley-1". The column labeled "PF (A)" gives the type-specific performance as accuracy for the parser-free system labeled "LIBLINEAR-4". The column labeled "I (A)" gives the performance of the integrated system labeled "Integration-3".

Type	Count	PB (A)	PB (P)	PF (A)	I (A)
ADJP	105	76.19	77.67	78.10	78.10
NML	372	61.29	62.47	64.52	64.78
NP	770	56.88	59.67	59.74	59.87
PP	60	41.67	43.10	55.00	45.00
S	129	54.26	58.82	27.91	55.04
UCP	55	49.09	50.94	50.91	58.18
VP	324	47.22	50.16	46.30	49.07
OTHER	54	81.48	83.02	72.22	85.19
Total	1,869	56.98	59.50	57.14	59.87

Table 6.5 shows example coordination structures produced by the parser-free system labeled "LIBLINEAR-4". For each type a correct and an incorrect coordination structure is shown. While it is not possible to choose examples that comprehensively represent each type, these examples were chosen because they are typical of their respective categories. It is interesting to observe that for many of the incorrect coordination structures the error is caused by confusion about how the conjoined members can be modified. For example, the incorrect "PP" coordination structure shows the phrase "located laterally" as not being included in the second conjunct. This implies that the coordination resolution is happy to modify both "distal streak region" and "narrow swath of cells" with "located laterally" when, according to the gold-standard data, it should only modify the latter. These data suggest that getting a better handle on how e.g. noun phrases can be modified could be a direction for future work on coordination resolution.

6.1.1.1 Conjunct-level Evaluation

Because the conjunction-level evaluation is very exacting, coordination structures that are nearly correct are given no partial credit. It is informative to look at conjunct-level evaluation to

Table 6.5: Examples of correct and incorrect coordination structures by type produced by the parser-free system labeled "LIBLINEAR-4". For each type (excepting those grouped together under "OTHER") an example of a correct (+) and incorrect (−) coordination structure produced by "LIBLINEAR-4" is shown. Directly following each incorrect coordination structure is the corresponding gold-standard (G) coordination structure. For most of the examples, the complete sentence is not shown and ellipses ("...") can be assumed to precede and follow an example.

Type		Example
ADJP	+	[structural] and [irreversible] developmental abnormalities.
	−	in the [pre-] or [perinatal period] include Bochdalek hernia
	G	in the [pre-] or [perinatal] period include Bochdalek hernia
NML	+	[Pax3] and [MyoD] expression is detected
	−	[muscle precursor cell migration] and [differentiation]
	G	muscle precursor cell [migration] and [differentiation]
NP	+	such as [complete amuscularization] or [phrenic nerve disruption].
	−	the [right middle lobe] and [accessory] buds
	G	[the right middle lobe] and [accessory buds]
PP	+	T is also ectopically expressed in mutant extraembryonic mesoderm [at the anterior embryonic junction] and [along the chorion].
	−	the expression of Dll1 [in the distal streak region] and [in only a narrow swath of cells] located laterally
	G	the expression of Dll1 [in the distal streak region] and [in only a narrow swath of cells located laterally]
S	+	[We used BLAST against the nr and EST databases] and [we found perfectly matching clones covering an ORF of 546 bp].
	−	Mig12 is present in both [the nucleus and the cytoplasm] and [the relative abundance] in the two compartments is variable.
	G	[Mig12 is present in both the nucleus and the cytoplasm] and [the relative abundance in the two compartments is variable].
UCP	+	Fourteen-week-old male [transgenic] and [wild-type] littermates
	−	distinguished from [type II] or [mixed muscle]
	G	distinguished from [type II] or [mixed] muscle
VP	+	embryos [were smaller in size] and [lacked the development of an accessory lobe]
	−	DNA was [isolated] and [separated on 1% agarose gel].
	G	DNA was [isolated] and [separated] on 1% agarose gel.

get an alternative analysis of how well the two approaches perform at getting conjuncts correct. For the parser-free approach, in particular, conjunct-level performance gives a more direct evaluation of the classifiers that determine conjunct boundaries. Conjunct-level performance is given as F-measure because the number of conjuncts proposed by a coordination resolution system may

differ from the number of conjuncts provided by the gold-standard data. The "Berkeley-1" parser-based system and "LIBLINEAR-4" parser-free system have nearly identical precision of 76.1 and 76.27, respectively. However, "Berkeley-1" has worse recall than "LIBLINEAR-4" by 2.68 points giving it an overall F-measure that is 1.47 points less than "LIBLINEAR-4". Note, that these two systems have nearly identical accuracy at the conjunction level of 56.98 and 57.14 for "Berkeley-1" and "LIBLINEAR-4", respectively. These results together show that while they get about the same number of coordination structures correct and incorrect, the incorrect coordination structures produced by "LIBLINEAR-4" have more correct conjuncts than "Berkeley-1".

Table 6.6: Conjunct-level performance.

System	P	R	F
Berkeley-1	76.1	71.69	73.83
LIBLINEAR-4	76.27	74.37	75.3
Integration-4	77.82	76.76	77.28

Figure 6.1 shows the conjunct-level performance for different lengths of conjuncts. Not surprisingly, the coordination resolution systems have higher performance on shorter conjuncts and that performance gets worse as the length of the conjuncts increases. However, it is interesting to note that performance for all three systems remains relatively constant for conjuncts of length eight through thirteen. For conjuncts of length fourteen or greater, the parser-based system performs better. However, because the distribution of conjunct lengths is highly skewed towards shorter conjuncts the higher performance of the parser-based system does not translate to higher over all performance.

6.1.2 Computational Complexity

The parser-free coordination approach described in Chapter 5 runs in $O(n)$ for a given sentence where n is the number of words in a sentence. This follows straightforwardly from the following. For every conjunction in a sentence all of the words to the left of the conjunction are visited once by the left conjunct border classifier. Similarly, all the words to the right of the con-

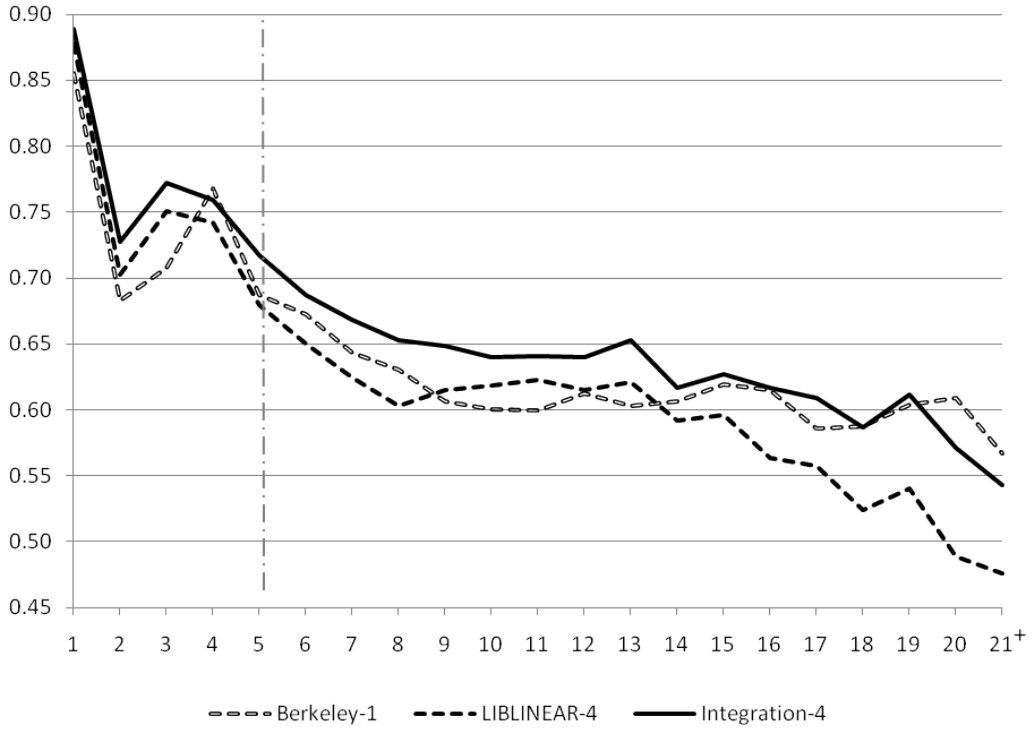

Figure 6.1: Conjunct-level performance by length. The F-measure (y-axis) is shown for different lengths in words of conjuncts (x-axis.) For example, "Integration-4" has an F-measure of 88.87 for conjuncts of length one. For conjunct lengths of one through four the exact F-measure is shown. For conjunct lengths of five and greater, the shown F-measures are smoothed by averaging the F-measure for a given length with the F-measures for the preceeding two lengths and the following two lengths. For example, the F-measure for "Integration-4" for conjuncts of length five is 70.85. When averaged with the F-measures for conjuncts of length three, four, six, and seven, the smoothed F-measure is 71.76. The graph is smoothed because as the length of the conjuncts increases, the number of conjuncts for that length decreases and correspondingly the exact F-measures fluctuate making it difficult to visualize the overall trends in the data.

junction are visited once by the right conjunct border classifier. Also, all of the words to the left of the left border of the left conjunct are visited by the extra conjunct classifier. Finally, because all of the words will be visited again for each conjunction in a sentence another term should be added to the computational complexity. This term is technically bound by n for pathological sentences that have coordination patterns like "A and B and C and D and E". However, in practice the number of conjunctions is less than the number of sentences so it is safe to leave this term out. When the language model features are used, the probability of n-1 candidate sentences are computed. Depending on the computational complexity of this procedure (presumably $O(n^2)$), the extraction of these features may dominate the overall computational complexity. However, in practice the coordination resolution system runs only incrementally slower when the language model features are used. In contrast, Hara et al.[18] report that their system has a computational complexity of $O(n^4)$.

The Berkeley Parser has been shown by Cer et al.[6] to be the fastest among the most widely used constituent parsers which include the Stanford, Charniak, Charniak-Johnson, and Bikel parsers. Admittedly, constituent parsers do a lot more work than the parser-free approach since they produce a deep syntactic parse which is then used here to produce coordination structures. Even still, it is informative to compare the parser-based and parser-free approaches. Both approaches were run against the CRAFT holdout test set ten times and timed. Time spent loading classifier and grammar models was subtracted from the run times. The fastest and slowest times from both systems were removed leaving eight data points for each. The parser-free approach ("LIBLINEAR-4") ran in average of 109.6 seconds with a standard deviation of 4.2 seconds. The parser-based approach ("Berkeley-2") ran in average of 810.2 seconds with a standard deviation of 4.1 seconds. Given that there are 1,869 conjunctions in the test set, the parser-free approach resolves 17.1 conjunctions per second while the parser-based approach resolves 2.3 conjunctions per second. Therefore, the parser-free approach was 7.4 times faster than the parser-based approach for this task.

6.2 Results on BIOCC Corpus

Table 6.7 shows the performance of several systems on the BIOCC corpus which is described in detail in Section §2.2. Results are reported for both the training set and the test set. Here the Berkeley Parser is trained on CRAFT data because BIOCC does not provide syntactic trees and so the same grammar models are used for coordination resolution on both the training and test data. The parser-free approach results on the training data come from ten-fold cross-validation while the results on the test data come from models trained on the training set. The tags used for training the Berkeley Parser on CRAFT can be either POS_{gold}, $\text{POS}_{\text{craft}}$, or $\text{POS}_{\text{genia}}$. However, because BIOCC does not provide its own gold-standard part-of-speech tags there is no corresponding POS_{gold} tags to provide as input to the CRAFT-trained parser. The best performing parser-based system is "Berkeley-2" which has an accuracy of 49.83% with a sentence failure rate of 2.5%. When the parser is trained with POS_{gold} or $\text{POS}_{\text{genia}}$ tags and then run with $\text{POS}_{\text{genia}}$ tags as input to the parser, the accuracy drops to 43.51% and 45.50%, respectively. The performance of 49.83% for "Berkeley-2"is 6.46 percentage points less than the accuracy of 56.29% it achieved on the CRAFT test set.

One way that the BIOCC corpus differs from the CRAFT corpus is that the coordination structures in the training and test sets are randomly selected. In contrast, the CRAFT corpus was stratified by the full-text articles the coordination structures were in. As discussed in Section §2.1.1, this characteristic of CRAFT has undesirable consequences on the ability to calculate statistical significance because the data can not be considered i.i.d. However, it is much safer to consider the folds in the training set and the test set to be i.i.d. and so calculating statistical signficance is feasible. For the results in Table 6.7 the McNemar test statistic was used to test statistical significance. In Table 6.7, each result was found to be significantly different at a 0.95 confidence from the result in the cell directly above it except for the difference between "Berkeley-2" and "Berkeley-1" for the test set and "SVM-2" and "LIBLINEAR-2" for the test set (the latter difference is significant at the 0.90 confidence level.)

Table 6.7: Performance of the parser-based and parser-free approaches on coordination resolution on the BIOCC corpus. The performance is reported as accuracy for both the training data (using ten-fold cross-validation) and the test data. The systems labeled "Berkeley-1", "Berkeley-2", and "LIBLINEAR-4_{craft}" were each trained on the training set of the CRAFT corpus and are therefore directly comparable. The remaining systems were trained using BIOCC data.

System	Train	Test
Berkeley-1	48.14	49.28
Berkeley-2	50.09	49.83
LIBLINEAR-4_{craft}	**55.05**	**54.50**
LIBLINEAR-1	51.80	50.38
SVM-1	52.83	52.58
LIBLINEAR-2	54.58	54.98
SVM-2	**55.98**	**56.77**

Unlike the parser-based approach, the parser-free approach can be trained on the BIOCC corpus. As such, it has a distinct advantage over the parser-based approach for this data. Not surprisingly the parser-free systems "LIBLINEAR-2" and "SVM-2" do not see a large drop in accuracy but remain approximately the same. For example, the accuracy of the "LIBLINEAR-2" system is 55.74% on the CRAFT training data and 54.58% on the BIOCC training data, a drop of 1.16%. However, its accuracy on the CRAFT test data is 53.88% and on the BIOCC test data 54.98%, an increase of 1.10%. Similarly, performance of the "SVM-2" system drops 1.16% on the training data sets and increases by 1.93% on the test sets. Another interesting result from Table 6.7 is that the language model features have a larger positive impact on the BIOCC data than they did on the CRAFT test set as measured by the difference in accuracy between "LIBLINEAR-1" and "LIBLINEAR-2" and similarly between "SVM-1" and "SVM-2". Here the improvement on the test data was 4.60% and 4.19% for "LIBLINEAR-2" and "SVM-2", respectively. In contrast the respective improvements on the CRAFT test set were only 1.23% and 1.76%. Results for the parser-free systems that make use of type-specific token boundary classifiers (namely, "SVM-3", "SVM-4", "LIBLINEAR-3", and "LIBLINEAR-4") are not reported here because BIOCC does not provide type information in the gold-standard data. However, it is possible to leverage the types provided in the CRAFT corpus as discussed next.

While it is interesting to observe that the parser-free approach has comparable performance on both the CRAFT and BIOCC corpora, the comparison between the parser-based and parser-free approaches on the BIOCC data is not fair for those parser-free systems trained on the BIOCC data. However, the system labeled "LIBLINEAR-4_{craft}" was trained on the CRAFT training data and then evaluated on the BIOCC training and test sets and provides a fair comparison with the parser-based approach. "LIBLINEAR-4_{craft}" outperforms "Berkeley-2" by 4.96% and 4.67% on the training data and test data, respectively. This result suggests that the parser-free approach is a more robust approach.

6.3 Comparison with Hara et al.

In this section the parser-based and parser-free approaches presented in this dissertation are compared with the previous state of the art on the Hara corpus which is described in Section §2.3. The Hara corpus consists of five folds and the evaluation results are from five-fold cross-validation. When the Berkeley Parser is evaluated on this corpus five different grammar models are created, one for each fold that is tested. The training data for each model consists of the other four folds plus sentences from the tested fold that do not contain coordination structures. These training sentences were selected to be consistent with how Hara et al. trained the parser models that they evaluated. Similarly, only POS_{gold} tags are used for training the parser to be consistent with Hara et al. For the Hara corpus, $\text{POS}_{\text{genia}}$ tags come from a part-of-speech tagger that is trained on all of the GENIA corpus that is not found in the Hara subset.

A slightly relaxed evaluation criteria is used here to be consistent with the results reported by Hara et al.[18]. Instead of requiring all of the conjuncts to be correct for the coordination structure only the left-hand boundary of the leftmost conjunct and the right boundary of the right conjunct must be correct. In practice this makes very little difference for performance and so it is not compared here with the stricter criteria used elsewhere throughout this dissertation. An additional difference between this corpus and the other two examined in this chapter is that coordination structures for the disjunction "but" are also included in the data. Performance on the disjunctive coordination structures is very similar to that of the ones for "and" and "or" but tends to be a few percentage points less. This small drop in performance has very little effect on overall performance because the disjunctions account for only 6.8% of the coordination structures. Incidentally, the conjunction "or" accounts for 9.9% of the coordination structures and "and" accounts for the remaining 83.3%.

Table 6.8 shows the results of several systems on the Hara corpus. Consistent with the results on the other corpora, adding the language model features (i.e. the difference between "SVM-1" and "SVM-2" and that of "LIBLINEAR-1" and "LIBLINEAR-2") and using type specific classifiers (i.e.

the difference between "SVM-2" and "SVM-3" and that of "LIBLINEAR-2" and "LIBLINEAR-3") both improve performance. The best performing system, "SVM-3", is 3.17% higher than the Berkeley parser. The systems "SVM-4" and "LIBLINEAR-4" were not run on this corpus because they are more involved to set up and based on the results from the other two corpora they are not expected to perform much differently than "SVM-3" and "LIBLINEAR-3", respectively.

Table 6.8: Performance of the parser-based and parser-free approaches on coordination resolution on the Hara corpus. The system labeled "Berkeley" is the parser-based approach using the Berkeley Parser trained on the Hara corpus using POS_{gold} tags with $\text{POS}_{\text{genia}}$ tags provided as input to the parser.

System	+	−	A
Berkeley	2,178	1,420	60.53
SVM-1	2,149	1,449	59.73
SVM-2	2,247	1,351	62.45
SVM-3	2,292	1,306	63.70
LIBLINEAR-1	2,103	1,495	58.45
LIBLINEAR-2	2,183	1,415	60.67
LIBLINEAR-3	2,282	1,316	63.42

Table 6.9 and Table 6.10 show results from Table 4 of Hara et al.[18] and results for the parser-based and parser-free approaches developed in this dissertation. Table 6.9 gives results when gold-standard part-of-speech tags are provided as input and Table 6.10 uses tagger tags. One of the encouraging results from these data is that the parser-free approach, "SVM-3", has the best overall performance for both tables. The column labeled "Hara" gives the results for their system which does not have the best performance for any coordination structure type excepting "PP" when POS_{gold} tags are provided and "OTHER" which only has three instances. When POS_{gold} tags are provided, the parser-free system "SVM-3" outperforms the Hara approach by 3.3% and the parser-based approach "Berkeley" outperforms the Hara approach by 2.4%. When $\text{POS}_{\text{genia}}$ tags are provided, the parser-free system "SVM-3" outperforms the Hara approach by 6.2% and the parser-based approach "Berkeley" outperforms the Hara approach by 3.2%. It should be noted that the part-of-speech tags used by the Hara approach for these results were those produced by

the Charniak-Johnson parser and so differences in the quality of the produced tags may account for some of the difference in performance.

One of the surprising results from Table 6.9 and Table 6.10 is the disparity in performance between the Berkeley Parser and the two parsers examined by Hara et al.: the Bikel-Collins Parser and the Charniak-Johnson Parser. Table 6.9 shows the Berkeley Parser outperforming the Charniak-Johnson Parser by 11.8 percentage points and, similarly, Table 6.10 shows the Berkeley Parser outperforming the Bikel-Collins Parser by 7.6 percentage points. These results are surprising because a number of studies (c.f. [6]) have shown that these three parsers perform at very similar levels when evaluated on parsing accuracy. A probable explanation for this discrepancy when they are evaluated on the coordination resolution task is that Hara et al. made a poor assumption about how best to apply the parsers to the task of coordination resolution. In the GENIA corpus, parent nodes of coordination structures are marked with the label "COOD". Hara et al. used these markers to update the syntactic categories of those nodes with the suffix "-COOD". The parsers were then trained with these modified node types and expected to learn these modified syntactic categories directly. In contrast, the parser-based approach taken with the Berkeley Parser (as described in Section §5.1) does not use the "COOD" labels but rather trains the parsers using the unmodified treebank data. The parses produced by the Berkeley Parser were then used to create coordination structures using the coordination structure production algorithm. Initial experiments with running the Berkeley Parser using the training method used by Hara et al. suggest that performance drops considerably using that approach for that parser. Therefore, an area of further research would be to apply these other parsers to the coordination resolution task using the coordination structure production algorithm.

6.4 Conclusions

In this chapter the parser-free approach was shown to perform at the same level as the parser-based approach on the CRAFT test data and the integration of the two approaches provided a further increase in performance. In contrast, the parser-free approach performed better on the

Table 6.9: Performance of the Berkeley Parser and the parser-free system compared with Hara et al. results using POS_{gold} tags. For each system, training was performed using POS_{gold} tags and POS_{gold} tags were provided as input to the respective coordination resolution approaches. The columns labeled "Hara" and "BC" give the respective performance of the Hara system and the Bikel-Collins parser. These results are taken directly from Table 4 of Hara et al[18].

Type	Count	Hara	BC	Berkeley	SVM-3
NP	2317	64.2	45.5	62.1	**67.2**
VP	465	54.2	**67.7**	62.2	62.4
ADJP	321	80.4	66.4	**81.3**	81.0
S	188	22.9	**67.0**	**67.0**	18.1
PP	167	**59.9**	53.3	55.7	**59.9**
UCP	60	36.7	18.3	36.7	**46.7**
SBAR	56	51.8	85.7	**87.5**	75.0
ADVP	21	85.7	90.5	90.5	**95.2**
Others	3	**66.7**	33.3	0.0	33.3
Total	3598	61.5	52.1	63.9	**64.8**

BIOCC and Hara corpora than the parser-based approach. Furthermore, the parser-free approach outperformed the Hara approach which was the previous state-of-the-art for coordination resolution in the biomedical domain. Also, it was shown that the parser-free approach was more than seven times faster than the parser-based approach.

Table 6.10: Performance of the Berkeley Parser and the parser-free system compared with Hara et al. results using POS_{genia} tags. For each system, training was performed using POS_{gold} tags and POS_{genia} tags were provided as input to the respective coordination resolution approaches. The columns labeled "Hara" and "CJ" give the respective performance of the Hara system and the Charniak-Johnson parser. These results are taken directly from Table 4 of Hara et al[18].

Type	Count	Hara	CJ	Berkeley	SVM-3
NP	2317	62.5	50.1	59.6	**66.6**
VP	465	42.6	**61.9**	56.1	57.9
ADJP	321	76.3	48.6	74.8	**81.9**
S	188	15.4	63.3	**66.5**	17.6
PP	167	53.9	58.1	52.7	**58.7**
UCP	60	38.3	26.7	31.7	**40.0**
SBAR	56	33.9	**83.9**	80.4	73.2
ADVP	21	85.7	90.5	85.7	**100.0**
Others	3	**33.3**	0.0	0.0	**33.3**
Total	3598	57.5	52.9	60.5	**63.7**

Chapter 7

Discussion

In Chapter 2 the three corpora used for training and evaluation throughout this dissertation were described. One salient characteristic of all three corpora is the wide diversity of coordination structures they contain. It was shown that coordination structures differ with respect to syntactic type, number of conjuncts, and length of conjuncts. This diversity speaks to the difficulty of performing coordination resolution accurately. While the specific parser-based and parser-free approaches developed in this research have pushed forward the state-of-the-art on this task, there remains a fairly large gulf between the reported performance results of automated coordination resolution and how consistently humans can perform this task. For the BIOCC corpus, the highest performing system acheived an accuracy of 56.77%. In contrast, inter-annotator agreement was 83.61%. Agreement between individual annotators and the adjudicated coordinations structures was as high as 94.26%. These results indicate that it may be possible for machines to improve significantly over the results presented in this dissertation. This chapter discusses the challenges that remain for computational approaches to match the performance of humans and gives recommendations for future work on this task. Finally, the significance of this work and conclusions are presented.

7.1 Multiple Sources of Structural Ambiguity

The structural ambiguity introduced by coordination phenomena competes, in a sense, with several other kinds of structural ambiguity such as prepositional phrase attachment and modifier

scope. Consider, for example, the following two sentences:

- The targeting construct was *linearized with NotI* **and** *electroporated into CJ7 ES cells.*
- . . . and thus difficult to *digitize* **and** *compile* into a standardized and integratable format.

Both examples end with a prepositional phrase beginning with the word "into". In the first example, the prepositional phrase belongs to the second conjunct but in the second example, the prepositional phrase is not part of a conjunct because it modifies both "digitize" and "compile". Similarly, in Section §6.1.1 it was discussed that a number of observed errors related to modifier scope. The following example was given to illustrate this problem:

- . . . *muscle precursor cell migration* **and** *differentiation* . . .
- . . . muscle precursor cell *migration* **and** *differentiation* . . .

Here, the incorrect coordination structure produced by the parser-free approach is followed by the gold-standard coordination structure. There are plenty of examples in the gold-standard data that look syntactically similar to the incorrect coordination structure given above. Here is one example:

- . . . *primordial germ cells* **and** *spermatogonia* . . .

The above examples suggest that focused attention on these sources of structural ambiguity, i.e. prepositional phrase attachment and noun modification, could benefit the coordination resolution task and may point to ways in which they may be resolved together. It should be possible, for example, to isolate prepositional phrase attachment into a separate subtask such that only prepositional phrases and the constituents that they modify are identified. This would facilitate a comparison between a parser-based approach and any number of possible parser-free approaches which would likely lead to insights on how to tackle this task analogous to those discovered in this dissertation for coordination. The new insights on how to resolve prepositional phrase attachment could then, hopefully, be used to by a parser to better handle examples like the "into" example above.

7.2 Integrated Approaches

The three paradigms for coordination resolution that were discussed in this dissertation were the parser-based approach which uses the Berkeley Parser which learns probabilistic context-free grammars from treebank data (see Section §5.1), the parser-free approach which uses conjunct boundary classifiers (see Section §5.3), and the Hara approach which uses sequence alignment on a weighted edit graph (see Section §1.1.4). All three of these approaches make use of machine learning techniques which produce some notion of a probability or confidence score. More sophisticated ways of integrating these three approaches could be explored that leverage the produced probabilities or scores. For example, one could train a classifier that uses these scores as features for classification of conjunct candidates produced by these three different approaches. Alternatively, it may be possible to improve the integrated approach found in Section §5.4 by, for example, first running the parser-based approach followed by the Hara approach, followed by the parser-free approach and experimenting with different thresholds of confidence for the produced coordination structures to determine which approach's coordination structure should be used. Another possible approach is to extend a syntactic parser such that it can accept and work with coordination structures produced by the parser-free or Hara approach. Again, by using the confidence scores produced by these two approaches only high quality coordination structures could be handed over to the parser. This might not only improve the coordination resolution of the parser but improve overall parsing performance.

7.3 Feature Selection

A better exploration of existing and new features may prove to be an effective way to improve performance of the parser-free approach. For starters, a more rigorous exploration of the existing lexical features may point towards a better performing combination of the features than were given in Table 5.2. Similarly, a feature that uses the output of the type classifier could be evaluated. This may prove more effective than using the classified types to triage instances to different sets

of type-specific classifiers as was done in Section §5.3.4. Similarly, there are a number of different ways to experiment with language model features that were not extensively explored. For example, all of the language models were built with 4-grams. Experiments with bigram- and trigram-based models might also prove fruitful. In Section §3.2.2 it was mentioned that a number of alternative approaches for leveraging language model probabilities were examined. For example, normalizing for candidate sentence length by dividing by the number of words was found to perform worse than adding in the probability of the candidate conjunction. However, there are a number of alternative normalization schemes that could also be tried. One common approach is to look at the changes in perplexity between words in a sequence and penalize for large fluctuations. Another approach for leveraging language model probabilities that was examined involved the comparison of n-gram phrases derived from the candidate conjuncts rather than entire candidate sentences. Another strategy for improving the language model derived features is to experiment with additional word-level features that are used to build the language model. In this dissertation only word stems, part-of-speech tags, and sentence boundary markers were explored. However, the SRILM toolkit allows for the use of arbitrary features to be added to the model. One potentially useful feature that could be added to the language models are word dependency relationships derived from a dependency parser. Another possible research direction for language model derived features is to look at "micro" models derived from the text of the document the target conjunction is found in. It may be possible to build, for example, a bigram model using only the text of the document that it is found in or documents that have been clustered together with the current document. All of these possible ways to experiment with ways to improve the language model suggest that such research would likely uncover an approach that performs better on the target metric used here, overall average rank percentile. However, it might also prove fruitful to add many language model derived features to the conjunct boundary classification models rather than only using one.

As discussed in Section §4.2.3, the graph-based semantic similarity features were shown to give a very small improvement for F-measure on the one-word conjunct pair classification task. One likely reason this technique may not have been more effective may be due to the poor coverage of

the underlying resources (i.e. WordNet and the UMLS) for the words being classified. Therefore, using semantic similarity metrics that are based on distributional statistics of words may prove to be more effective because, presumably, the coverage would be better. However, widening the notion of "similarity" to that of "relatedness" may also prove to be an effective strategy for discovering better features for coordination resolution. For example, an OWCP such as "age and gender" shows two words that are clearly related but not necessarily similar. The resolution of one-word conjunct pairs as discussed in Chapter 4 provides a good task for evaluating different semantic similarity and semantic relatedness metrics that are either graph-based and/or distributional. One can imagine that extensive experimentation along these lines would likely produce effective features for coordination resolution and may also yield give additional insights into coordination as well.

7.4 Significance

This work has narrowly focused on the task of automated coordination resolution as an interesting and challenging problem in and of itself. However, performant coordination resolution is of little interest as a stand-alone application and is really of practical interest only as it relates to "downstream" processing such as information extraction tasks. There are two means by which information extraction may benefit from better coordination resolution. The first is that better coordination resolution could be leveraged to improve syntactic parsing performance. Information extraction systems that use a syntactic parser may then benefit simply due to improved results from the parser. However, given a high-quality and fast coordination resolution solution, information extraction systems may be refactored to make direct use of produced coordination structures. For example, Cohen et al.[12] describe a high-precision information extraction system that leverages coordination structures directly. They demonstrate that using a parser-based coordination resolution approach to provide coordination structures to their protein-protein interaction recognition system resulted in a significant performance boost from an overall F-measure of 24.7 to 27.6.

7.4.1 Porting Coordination Resolution to a New Domain

It is interesting to contrast the amount of effort required to annotate the BIOCC corpus versus the CRAFT corpus. The BIOCC corpus was annotated with a very simple annotation scheme and takes a few hours to train a domain expert (e.g. a biologist) with little or no linguistics background how to perform the task consistently. The CRAFT corpus, in contrast, is annotated with a very rich annotation scheme that facilitates representation of deep syntactic analysis of a sentence. Furthermore, the work must be performed by a highly trained annotator with an extensive background in linguistics in addition to exposure to the subject domain of the texts. It is a reasonable hypothesis that the annotation speed of directly annotating coordination structures (as was done for BIOCC) is much faster than annotating syntactic trees (as was done for CRAFT.) These observations suggest that porting the parser-free approach to a different domain such as clinical notes or legal documents should be much easier than for the parser-based approach because creating new training data is much easier and less expensive. Furthermore, if a syntactic parser were developed or extended to make use of coordination structures as input, then an aternative path for porting a syntactic parser to a new domain presents itself. That is, a syntactic parser could be trained on data that is already available (e.g. CRAFT) and ported to another domain (e.g. the medical domain) by using coordination structures produced by a parser-free coordination resolution system ported to that domain by means of directly annotating coordination structures in texts of that domain and training on those structures. In a similar vein, the annotation of coordination structures by domain experts may be a means for improving syntactic annotation in a new domain when resources allow for both. That is, syntactic annotation quality might improve if the Linguists who are performing the syntactic annotation are given coordination structures annotated by domain experts.

7.5 Conclusions

As discussed in Section §1.1 the task of coordination resolution has not received the focused attention that it deserves. This research has shown that by focusing directly on this task, performance can be improved and insights on the phenomena can be gained. A simple, fast, and accurate alorithm for coordination resolution was presented that advances the state-of-the-art for coordination resolution. This chapter has proposed future research directions for this task.

- talk about DDNs - talk about distributional approaches for semantic similarity - talk about cotraining / semi-supervised approaches - google n-gram approach

Bibliography

[1] Satanjeev Banerjee and Ted Pedersen. An adapted lesk algorithm for word sense disambiguation using wordnet. In Alexander Gelbukh, editor, Computational Linguistics and Intelligent Text Processing, volume 2276 of Lecture Notes in Computer Science, pages 117–171. Springer Berlin / Heidelberg, 2010.

[2] Ann Bies, Mark Ferguson, Karen Katz, Robert Macintyre, Major Contributors, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. Bracketing guidelines for treebank ii style penn treebank project, 1995.

[3] Ann Bies, Seth Kulick, and Mark Mandel. Parallel entity and treebank annotation. In CorpusAnno '05: Proceedings of the Workshop on Frontiers in Corpus Annotations II, pages 21–28, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[4] Daniel M. Bikel. A distributional analysis of a lexicalized statistical parsing model. In Dekang Lin and Dekai Wu, editors, Proceedings of EMNLP 2004, pages 182–189, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[5] E. Buyko, K. Tomanek, and U. Hahn. Resolution of coordination ellipses in biological named entities using conditional random fields. In PACLING 2007-Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, pages 163–171, 2007.

[6] D. Cer, M.C. de Marneffe, D. Jurafsky, and C.D. Manning. Parsing to Stanford Dependencies: Trade-offs between speed and accuracy. may 2010.

[7] 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`.

[8] F. Chantree, A. Kilgarriff, A. De Roeck, and A. Willis. Disambiguating coordinations using word distribution information. Proceedings of RANLP2005, 2005.

[9] Eugene Charniak and Mark Johnson. Coarse-to-fine n-best parsing and maxent discriminative reranking. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), pages 173–180, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

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

[11] K. Bretonnel Cohen, Helen Johnson, Karin Verspoor, Christophe Roeder, and Lawrence Hunter. The structural and content aspects of abstracts versus bodies of full text journal articles are different. BMC Bioinformatics, 11(1):492, 2010.

[12] Kevin B. Cohen, Karin Verspoor, Helen L. Johnson, Chris Roeder, Philip V. Ogren, William A. Baumgartner Jr, Elizabeth White, Hannah Tipney, and Lawrence Hunter. High-precision biological event extraction with a concept recognizer. In Proceedings of the Workshop on BioNLP: Shared Task, pages 50–58. Association for Computational Linguistics, 2009.

[13] Michael Collins. Head-driven statistical models for natural language parsing. Computational linguistics, 29(4):589–637, 2003.

[14] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research, 9:1871–1874, 2008.

[15] David Ferrucci and Adam Lally. UIMA: an architectural approach to unstructured information processing in the corporate research environment. Natural Language Engineering, 10(3-4):327–348, 2004.

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

[17] Dan Gusfield. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, 1997.

[18] Kazuo Hara, Masashi Shimbo, Hideharu Okuma, and Yuji Matsumoto. Coordinate structure analysis with global structural constraints and alignment-based local features. In ACL-IJCNLP '09: 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: Volume 2, pages 967–975, Morristown, NJ, USA, 2009. Association for Computational Linguistics.

[19] William Hersh and Ellen Voorhees. Trec genomics special issue overview. Inf. Retr., 12(1):1–15, 2009.

[20] 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, 2005.

[21] Deirdre Hogan. Coordinate noun phrase disambiguation in a generative parsing model. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 680–687, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[22] G. Hripcsak and A.S. Rothschild. Agreement, the f-measure, and reliability in information retrieval. Journal of the American Medical Informatics Association, 12(3):296–298, 2005.

[23] Thorsten Joachims. Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, 1999.

[24] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii. GENIA corpus: a semantically annotated corpus for bio-textmining. Bioinformatics, 19(suppl 1):i180–i182, 2003.

[25] Martin Krallinger, Alexander Morgan, Larry Smith, Florian Leitner, Lorraine Tanabe, John Wilbur, Lynette Hirschman, and Alfonso Valencia. Evaluation of text-mining systems for biology: overview of the second biocreative community challenge. Genome Biology, 9(Suppl 2):S1, 2008.

[26] Matthew Lease and Eugene Charniak. Parsing biomedical literature. In In Proceedings of the Second International Joint Conference on Natural Language Processing (IJCNLP-05), Jeju Island, Korea, pages 58–69, 2005.

[27] D. Lin. An information-theoretic definition of similarity. In Proceedings of the 15th International Conference on Machine Learning, volume 1, pages 296–304. Citeseer, 1998.

[28] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: the penn treebank. Comput. Linguist., 19(2):313–330, 1993.

[29] M. Marneffe, B. Maccartney, and C. Manning. Generating typed dependency parses from phrase structure parses. In Proceedings of LREC-06, pages 449–454, 2006.

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

[31] David McClosky, Eugene Charniak, and Mark Johnson. Effective self-training for parsing. In Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, pages 152–159, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[32] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. Non-projective dependency parsing using spanning tree algorithms. In HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, pages 523–530, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[33] Preslav Nakov and Marti Hearst. Using the web as an implicit training set: Application to structural ambiguity resolution. In Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pages 835–842, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics.

[34] Jens Nilsson, Joakim Nivre, and Johan Hall. Graph transformations in data-driven dependency parsing. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 257–264, Sydney, Australia, July 2006. Association for Computational Linguistics.

[35] J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. MaltParser: A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(02):95–135, 2007.

[36] Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. Maltparser: A language-independent system for data-driven dependency parsing. Natural Language Engineering, 13(02):95–135, 2007.

[37] Joakim Nivre and Ryan McDonald. Integrating graph-based and transition-based dependency parsers. In Proceedings of ACL-08: HLT, pages 950–958, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[38] Philip V. Ogren. Knowtator: A Protégé plug-in for annotated corpus construction. In Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume: demonstrations, page 275. Association for Computational Linguistics, 2006.

[39] Philip V. Ogren, Philipp G. Wetzler, and Steven Bethard. ClearTK: a UIMA toolkit for statistical natural language processing. In UIMA for NLP workshop at Language Resources and Evaluation Conference (LREC), 2008.

[40] Philip V. Ogren, Philipp G. Wetzler, and Steven Bethard. ClearTK: a framework for statistical natural language processing. In Proceedings of the Biennial GSCL Conference 2009. Gunter Narr Verlag, 2009.

[41] Ted Pedersen, Serguei V.S. Pakhomov, Siddharth Patwardhan, and Christopher G. Chute. Measures of semantic similarity and relatedness in the biomedical domain. Journal of Biomedical Informatics, 40(3):288 – 299, 2007.

[42] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. Wordnet::similarity: measuring the relatedness of concepts. In HLT-NAACL '04: Demonstration Papers at HLT-NAACL 2004 on XX, pages 38–41, Morristown, NJ, USA, 2004. Association for Computational Linguistics.

[43] Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. Learning accurate, compact, and interpretable tree annotation. In Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, pages 433–440, Sydney, Australia, July 2006. Association for Computational Linguistics.

[44] Philip Resnik. Semantic similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. Journal of Artificial Intelligence, 11(11):95–130, 1999.

[45] Masashi Shimbo and Kazuo Hara. A discriminative learning model for coordinate conjunctions. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 610–619, 2007.

[46] A. Stolcke. SRILM - an extensible language modeling toolkit. In Seventh International Conference on Spoken Language Processing, volume 3. Citeseer, 2002.

[47] Y. Tateisi, A. Yakushiji, T. Ohta, and J. Tsujii. Syntax Annotation for the GENIA corpus. In Proceedings of the IJCNLP, pages 222–227, 2005.

[48] Y. Tsuruoka, Y. Tateishi, J.D. Kim, T. Ohta, J. McNaught, S. Ananiadou, and J. Tsujii. Developing a robust part-of-speech tagger for biomedical text. Advances in Informatics, pages 382–392, 2005.

[49] David Vadas and James Curran. Adding noun phrase structure to the penn treebank. In Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 240–247, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[50] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics, pages 133–138, Morristown, NJ, USA, 1994. Association for Computational Linguistics.

[51] Kenji Sagae Yuka Tateisi, Yusuke Miyao and Junichi Tsujii. Genia-gr: a grammatical relation corpus for parser evaluation in the biomedical domain. In Proceedings of the Sixth International Language Resources and Evaluation (LREC'08), Marrakech, Morocco, may 2008. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.