Journal of Biomedical Informatics 42 (2009) 852–865

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Porting a lexicalized-grammar parser to the biomedical domain ☆

Laura Rimell *, Stephen Clark

Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford OX1 3QD, UK

ARTICLE INFO

Article history:
Received 27 May 2008
Available online 25 December 2008

Keywords:
Statistical parsing
Tagging
CCG
Porting
Annotation
Evaluation

ABSTRACT

This paper introduces a state-of-the-art, linguistically motivated statistical parser to the biomedical text mining community, and proposes a method of adapting it to the biomedical domain requiring only limited resources for data annotation. The parser was originally developed using the Penn Treebank and is therefore tuned to newspaper text. Our approach takes advantage of a lexicalized grammar formalism, Combinatory Categorial Grammar (CCG), to train the parser at a lower level of representation than full syntactic derivations. The CCG parser uses three levels of representation: a first level consisting of part-of-speech (POS) tags; a second level consisting of more fine-grained CCG lexical categories; and a third, hierarchical level consisting of CCG derivations. We find that simply retraining the POS tagger on biomedical data leads to a large improvement in parsing performance, and that using annotated data at the intermediate lexical category level of representation improves parsing accuracy further. We describe the procedure involved in evaluating the parser, and obtain accuracies for biomedical data in the same range as those reported for newspaper text, and higher than those previously reported for the biomedical resource on which we evaluate. Our conclusion is that porting newspaper parsers to the biomedical domain, at least for parsers which use lexicalized grammars, may not be as difficult as first thought.

1. Introduction

Parsing technology has improved dramatically over the past decade, to the point where robust parsers are now available which can deliver fairly accurate linguistic representations for sentences of unrestricted, naturally occurring text. This improvement is due primarily to two factors: one, the use of wide-coverage grammars, typically extracted from treebanks consisting of sentences manually annotated with parse trees; and two, the use of statistical parsing models to deal with the large search space resulting from the use of a wide-coverage grammar. Crucially, both advances rely on the existence of a large treebank, which is needed to obtain a grammar with reasonable coverage and to provide enough training data for accurate estimation of a statistical parsing model.

The largest treebank currently available is the Penn Treebank [1], which dominates contemporary parsing research in Natural Language Processing (NLP). However, the sections of the Penn Treebank that have been used for parser development consist of sentences from the Wall Street Journal, making the resulting parsers highly tuned to newspaper text. Hence one of the key questions currently facing researchers in statistical parsing is how to take state-of-the-art parsers of newspaper text and adapt them to other domains.

From the perspective of biomedical text mining, this community is in need of high quality language processing tools and resources, including syntactic parsers. In this paper, we consider the question of how to take an existing parser of newspaper text, one that we argue is ideally suited to biomedical text mining applications, and port it to the biomedical domain. The key innovation is to consider the *level of representation* at which the porting needs to take place, and we argue that this consideration leads naturally to a parser using a lexicalized grammar formalism, in our case Combinatory Categorial Grammar (CCG) [2].

CCG is lexicalized in the sense that it associates with each word in a sentence an elementary syntactic structure, in CCG's case a lexical category encoding subcategorisation information. Since lexical categories can be treated as labels, standard statistical tagging techniques can be used to assign lexical categories to words in a sentence; this technique is known as *supertagging* [3,4]. One hypothesis we have investigated in this paper is that, since a sequence of lexical categories is a flat, rather than hierarchical, representation, the manual annotation of lexical categories can be performed relatively quickly; in addition, since CCG lexical categories contain so much information, porting at this level of representation alone is sufficient for successful domain adaptation [5].

The CCG parser we use exploits three levels of representation [6]: a first level in which part-of-speech (POS) tags are assigned to each

☆ We would like to thank Sampo Pyysalo for sharing information about the BioInfer corpus and Andrew Clegg for discussion of his parser evaluation methods. We would also like to thank the two anonymous reviewers for their suggestions. This work was supported by EPSRC Grant EP/E035698/1: Accurate and Efficient Parsing of Biomedical Text.

* Corresponding author. Fax: +44 1865 273839.
E-mail addresses: laura.rimell@comlab.ox.ac.uk (L. Rimell), stephen.clark@com lab.ox.ac.uk (S. Clark).

1532-0464/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2008.12.004

word, where the POS tags are fairly coarse-grained grammatical labels; a second level in which CCG lexical categories are assigned to each word, where the lexical categories are fine-grained grammatical labels encoding subcategorisation information; and a third, hierarchical level consisting of CCG derivations. Thus a natural question to ask in this framework is whether annotated data in the new domain is needed at all three levels for successful porting, or whether annotated data at the lower levels suffices to produce an accurate parser.

The experimental results confirm our hypothesis: we find that simply retraining the POS tagger on biomedical data leads to a large improvement in performance, compared to the newspaper parser applied to biomedical text, and that using annotated data at the more complex, lexical category level of representation (but not the derivation level) improves parsing accuracy further. The parsing accuracies we obtain for the biomedical data, using a test suite of gold-standard grammatical relations, are in the same range as those reported for newspaper text (albeit using a slightly different evaluation scheme). Our conclusion is that porting newspaper parsers to the biomedical domain, at least for parsers which use lexicalized grammars, may not be as difficult as first thought.

Section 2 describes the problem of parser domain adaptation and discusses some previous approaches. Sections 3 and 4 introduce the CCG grammar formalism and parser used in the experiments. These are relatively detailed introductions for those members of the biomedical text mining community not familiar with linguistically motivated statistical parsing. Section 5 describes our approach to porting the CCG parser, exploiting the multiple levels of representation used by the parser. This section also describes the annotation procedure that was used to create a new resource of annotated biomedical sentences, and the improved accuracy of the CCG supertagger when trained on this resource. Finally, Section 6 describes the procedure required for evaluating the parser using a resource not based on CCG, and gives final parsing results.

2. Domain adaptation

Wide-coverage statistical parsers derive their robustness and accuracy from the large corpora on which they are trained. Performance declines, however, when a parser is used to analyse text with vocabulary or usage conventions which vary from those in the training data, since the probabilities in the parsing model do not reflect the characteristics of the new domain. Moreover, there may be new syntactic constructions which do not appear at all in the original training data.

Most modern parsers, including the CCG parser used in this paper, are trained on Wall Street Journal (WSJ) articles from the Penn Treebank (PTB) [1]. Such parsers may be over-tuned to newspaper text. Performance of WSJ-trained parsers has been observed to decline on other corpora, such as the Brown corpus section of the PTB, a mixed-genre corpus including both non-fiction and fiction [7]. It is generally supposed that adaptation is even more difficult for technical domains such as biomedical literature, with its highly specialised vocabulary and conventions. Clegg and Shepherd [8] found that the Bikel [9], Charniak [10], and Collins [11] parsers scored significantly lower on data from the GENIA treebank [12] than on a WSJ test set, down from 86.4% to 75.6% *F*-score on average, using a constituent-based evaluation method (see Section 6.1.1). Lease and Charniak [13] showed that the unadapted Charniak parser scored lower on Brown data (83.4% *F*-score) than on WSJ data (89.5%), and lower still on GENIA [14] (76.3%), also using a constituent-based evaluation.

Manually annotating large amounts of data with parse trees is costly, particularly in technical domains (see [8] for examples of constructions which only a biomedical domain expert would be able to annotate). The Penn Treebank, containing one million words of text, took roughly three years to develop, and thus it is not practical to annotate the amount of training data required for fully supervised training for each new domain. Several techniques have therefore been proposed to adapt existing parsers, tuning them to a new domain while leveraging their existing wide coverage.

One standard approach relies on having a small amount of annotated data in the target domain (in-domain data) and a large amount of annotated data from another domain, such as newspaper text. A new model is developed which includes information from both domains. This can be done by simply merging the data with a weighting factor (Gildea [7] uses this method for adaptation to the Brown corpus, as do Hara et al. [15] and the present paper for the biomedical domain), or with more complex methods. Bacchiani et al. [16,17] discuss *maximum a posteriori* methods for parsing and language modelling in general. With this type of approach the in-domain data is used to tune the parser to the target domain, while the out-of-domain data provides smoothing.

When no annotated in-domain data is available, so-called "unsupervised" approaches can be used. One approach which has proved effective is self-training, where the "best guesses" of a parser on previously unseen data are used to create new training data. McClosky et al. [18] showed that self-training is useful within a single domain, when the parser's *n*-best parses are reranked with a discriminative reranker, and the highest probability reranked parse is added to the training data. Self-training has been used for domain adaptation from WSJ text to the Brown corpus and the British National Corpus [17,19,20]. In recent work by McClosky and Charniak [21], self-training has also been used for adaptation to the biomedical domain. Other unsupervised approaches include co-training [22], where two independent parsers are trained on each other's highest probability parses (or parses scoring highly according to some utility function [23]), and structural correspondence learning, where a comparison of features in the original and target domains is used to create a common feature representation which is meaningful across both domains. This latter approach has been used by Blitzer et al. [24] to adapt a POS tagger.

An important question in the adaptation process is the level of representation on which to focus. Since many parsers produce (or rely on) multiple levels of representation, including, at a minimum, POS tags and syntactic bracketing, there are multiple possible levels at which to perform domain adaptation. It is an open question whether a particular level is more useful for porting, and whether the choice of level should vary by domain. Lease and Charniak [13] hypothesize that unfamiliar vocabulary is a key problem in the biomedical domain. They used a retrained POS tagger as input to the bracketing step, achieving an increase in parsing accuracy without training a new parsing model. They also experimented with other lexical resources including a domain-specific dictionary and named entities, but found that these did not have a significant effect over and above the retrained POS tagger.

Our approach is similar in spirit to that of Lease and Charniak [13], in the sense that we focus adaptation efforts on lower, more local levels of representation than full parse trees. For a lexicalized-grammar parser such as the CCG parser, the CCG lexical category is an additional, intermediate level of representation which can be exploited. Clark et al. [5] performed a pilot study in which they retrained the CCG supertagger with question data suitable for a question–answering system, resulting in improved supertagging accuracy on questions. This should in turn lead to improved parsing accuracy [25]. Our current approach tests whether the method of training at lower levels of representation, focusing on the POS and lexical category levels, can be used to adapt a parser to biomedical data.

In the domain of lexicalized-grammar parsing, Hara et al. [26] follows a similar approach to Clark et al. [5] and Clark and Curran [25]. They used the parser of Ninomiya et al. [27], a version of the Enju parser [28]. Enju is based on Head Driven Phrase Structure Grammar (HPSG) [29], a grammar formalism which also has an intermediate level of representation for lexical entries. Hara et al. trained a new log-linear model of lexical entry assignments based on the GENIA treebank and incorporated this with the existing newspaper-trained model, with a resulting increase in parser *F*-score from 86.4% to 89.0% on HPSG predicate-argument dependencies automatically extracted from the GENIA treebank. Unlike our work, lexical entries for the new model were taken from a gold standard including full parse trees, whereas our data set was annotated only at the lexical category level, and so it provides a more practical test of local annotation.

3. Combinatory Categorial Grammar

Combinatory Categorial Grammar (CCG) [2] is a linguistic grammar formalism which originated in the syntactic theory literature, but one which has been heavily influenced by computational considerations. In this regard it is similar to other grammar formalisms in NLP, such as Tree Adjoining Grammar [30], Head Driven Phrase Structure Grammar [29] and Lexical Functional Grammar [31].

There is a large body of recent work on using CCG for robust, wide-coverage parsing [6,32–35]. Clark and Curran [6] makes the case for using CCG for wide-coverage parsing, and demonstrates that accurate and efficient parsing is possible with CCG. There are two key advantages to CCG which make it well-suited to practical parsing, and ideally suited to large-scale applications involving biomedical text processing. First, CCG allows direct access to the predicate-argument structure of a sentence (roughly who did what to whom), including long-range dependencies between words.[1] Such predicate-argument relations are crucial for discovering interesting relationships between biomedical entities such as genes and proteins. Second, by exploiting the lexicalized nature of CCG, the parser of Clark and Curran [6] is surprisingly efficient, parsing tens of sentences per second on standard hardware. This is an order of magnitude faster than the commonly used Collins parser [11], for example. High parser speeds are crucial for large-scale text processing involving large numbers of biomedical research papers.

3.1. The theory of CCG

CCG [2] is a type-driven lexicalized theory of grammar based on Categorial Grammar [36]. What this means is that CCG associates with each word in a sentence a syntactic category, or type. If the category takes arguments, then the type and directionality of the arguments are part of the category specification. The main difference between CCG and a context-free grammar (CFG), for example, is that CCG takes much of the information that is represented in a CFG by rewrite rules, and pushes it down to the lexical level.

CCG categories can be either basic or complex. Examples of basic categories are *S* (sentence), *N* (noun), *NP* (noun phrase) and *PP* (prepositional phrase). Complex categories are built recursively from basic categories, and indicate the type and directionality of arguments (using slashes), and the type of the result. For example, the following category for the transitive verb *activates* specifies its first argument as a noun phrase to its right (since the outermost slash is pointing to the right), its second argument as a noun phrase to its left (since the innermost slash is pointing to the left), and its result as a sentence:

$$\text{activates} := (S \setminus NP)/NP \tag{1}$$

Categories are combined in a derivation using *combinatory rules*. In the original Categorial Grammar [37], which is context-free, there are two rules of *functional application*:

$$X/Y \quad Y \Rightarrow X \ (>) \tag{2}$$

$$Y \quad X \setminus Y \Rightarrow X \ (<) \tag{3}$$

where *X* and *Y* denote categories (either basic or complex). The first rule is *forward application* (>) and the second rule is *backward application* (<). Fig. 1 gives an example of a derivation using these rules.[2]

CCG extends the original Categorial Grammar by introducing additional combinatory rules. The first is *forward composition*, which Steedman denotes by $>\mathbf{B}$ (since **B** is the symbol used by Curry to denote function composition in combinatory logic [38]):

$$X/Y \quad Y/Z \ \Rightarrow_{\mathbf{B}} \ X/Z \ (>\mathbf{B}) \tag{4}$$

Forward composition is often used in conjunction with *type-raising* (**T**), as in Fig. 2. In this case type-raising takes a subject noun phrase (*the site*) and turns it into a functor looking to the right for a verb phrase; *the site* is then able to combine with *regulates* using forward composition, giving *the site regulates* the category *S*[*dcl*]/*NP* (a declarative sentence missing a noun phrase to the right). It is exactly this type of constituent which the object relative pronoun category is looking for to its right: *(NP\NP)/(S*[*dcl*]/*NP*). CCG was designed to capture precisely this sort of long-range dependency: in this example of object extraction from a relative clause, *interaction* and *regulates* can be arbitrarily far apart (for example *the interaction which the evidence shows the site regulates*), but the dependency between the verb and direct object can still be recovered through the category for the relative pronoun.

Note that *the site regulates* is a perfectly reasonable constituent in CCG, having the type *S*[*dcl*]/*NP*. This is in contrast with most other grammar formalisms, and leads to a flexible notion of constituency which also allows elegant analyses for sentences such as *early indications suggested but the experimental results did not support a correlation*, even though this construction is often described as "non-constituent coordination".[3] In this example, *early indications suggested* and *the experimental results did not support* have the same type, allowing them to be coordinated, resulting in *early indications suggested but the experimental results did not support* having the type *S*[*dcl*]/*NP*.

Note also that it is this flexible notion of constituency which leads to so-called "spurious ambiguity" in CCG, since even the simple sentence *early indications suggested a correlation* will have more than one derivation, with each derivation leading to the same set of predicate-argument dependencies. Steedman [2,39] motivates the existence of this additional ambiguity, giving a number of examples across a range of linguistic levels, including the phonological, as well as syntactic, level of representation.

Steedman [2] describes a small number of additional combinatory rules which are part of the theory, and Clark and Curran [6] describes the combinatory rules used in the CCG parser. Clark and Curran [6] also describes the approach taken to resolving the additional, "spurious" ambiguity arising in CCG, through the use of *normal-form* derivations. Informally, a normal-form derivation is one

[1] Long-range dependencies occur in a variety of syntactic constructions, and are problematic for shallow parsing approaches; examples arising from extraction from a relative clause and coordination are discussed below.

[2] Some of the basic categories carry features indicating their subtype; *S[dcl]*, for example, is a declarative sentence.

[3] Examples like these are often called "non-constituent coordination" because *early indications suggested* and *the experimental results did not support* are not constituents in most theories, since they contain a subject and only part of a verb phrase. However, a standard linguistic test for whether a unit is a constituent is whether the unit can be coordinated, suggesting that these examples should be treated as constituents.

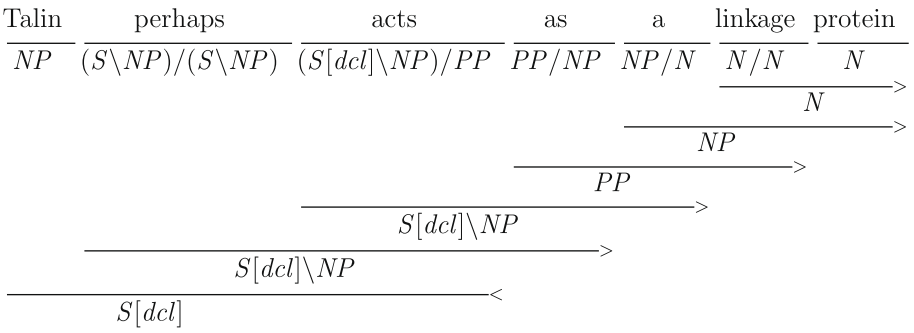

Fig. 1. Example derivation using forward and backward application.

where type-raising and composition are only used where necessary (for example in the object extraction case in Fig. 2 and the non-constituent coordination case described above). The treebank used to develop the parser, CCGbank [40], contains normal-form CCG derivations for the sentences in the Penn Treebank.

3.2. CCGbank

CCGbank [40] is the grammatical resource used to develop the CCG parser. It provides the grammar used by the parser (or at least part of the grammar for the biomedical version), and it provides training data for the statistical models. CCGbank was created by converting the phrase-structure trees in the Penn Treebank into CCG normal-form derivations. This was a non-trivial process, involving some pre-processing of the phrase-structure trees in order to allow correct CCG analyses for some constructions, such as coordination, and required a significant amount of development time. Hockenmaier and Steedman [40] gives a detailed description of the procedure used to create CCGbank.

The grammar consists primarily of two components: a lexicon, and a small set of manually defined combinatory rules (which were described in Section 3.1). Each word in the lexicon is associated with a list of lexical categories which can be assigned to the word; these lists are obtained from the CCGbank derivations. Since the lexicon is extracted from a limited sample (roughly 40,000 sentences), it is missing entries for some word-category pairs that it will require in practice, especially for rare, and of course unseen, words. For example, if the word *exhibit* does not appear in the training data, then the lexicon will not contain a list of lexical categories which can be assigned to the word. Similarly if the word appears only a few times, the lexicon may not contain a complete list. In the parser implementation, the list for a word is considered complete if the word appears more than a fixed number of times in the training sections of CCGbank (in this work 20 times). For words appearing less frequently, the supertagger, the component which assigns lexical categories to words, can assign any lexical category to the word which is consistent with the word's POS tag.

For rare and unseen words the supertagger is still able to make an informed decision regarding the likely lexical categories, since it has access to a large number of contextual clues, including the POS tag of the target word, with which to make a decision. The supertagger is described in Section 4.1. The lexical category set extracted from CCGbank and used by the supertagger and parser contains 425 categories. Clark and Curran [6] gives the complete list of rules used by the parser, an extract from the lexicon, and a more detailed description of how CCGbank was used to develop the CCG parser.

4. The CCG parser

The stages in the CCG parsing pipeline are as follows. First, a POS tagger assigns a single POS tag to each word in a sentence. POS tags are fairly coarse-grained grammatical labels indicating part-of-speech (e.g. NNS, plural noun); the Penn Treebank set, used here, contains 48 labels. Second, a CCG supertagger assigns lexical categories to the words in the sentence. Supertagging was originally developed for lexicalized Tree Adjoining Grammar [3], but has been particularly successful for wide-coverage CCG parsing [4,6]. The advantage in using a supertagger to assign the lexical categories, rather than allowing the parsing stage to make the decision, is that this greatly reduces the search space; moreover, since supertaggers run in linear time, the whole process is significantly more efficient (in time and space). Finally, the parsing stage combines the categories, using CCG's combinatory rules, and builds a packed chart representation containing all the derivations which can be built from the lexical categories. The Viterbi algorithm efficiently finds the highest scoring derivation from the packed chart.

4.1. The POS tagger and CCG supertagger

The taggers are based on Maximum Entropy tagging methods [41], using log-linear probability distributions to model local decisions at each point in the tagging process. The conditional probability of a tag sequence, given a sentence, is the product of the conditional probabilities of each tag, given the tag's context. The

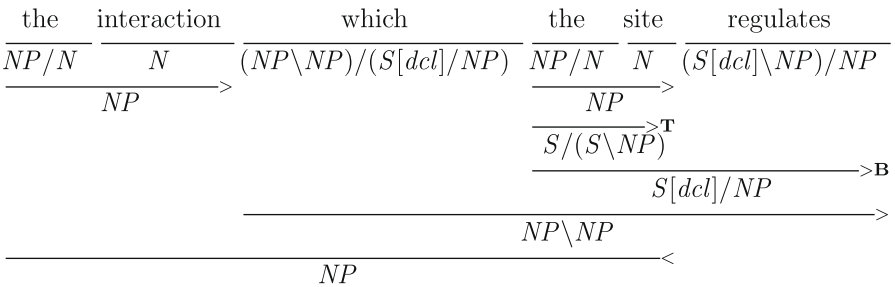

Fig. 2. Example derivation using type-raising and forward composition.

context is taken to be two words either side of the target word, including the previous two tags. The Viterbi algorithm for sequences is used to find the highest scoring tag sequence.

The advantage of Maximum Entropy taggers is that there is great flexibility in which features can be included in the model. For example, the POS tagger uses features based on the internal properties of the target word, such as prefix and suffix information, and whether the word contains special categories such as hyphens, in order to make tagging decisions for rare and unknown words. If an unknown word ends in *ing*, for example, its POS tag is likely to be VBG (verbal present participle). The features can be overlapping and non-independent, and the training algorithm will account for that. Here we use the standard generalised iterative scaling (GIS) algorithm to train the taggers, using the implementation described in Curran and Clark [42].

The taggers are highly efficient, tagging hundreds of thousands of words per second, making them ideally suited for text mining applications requiring the processing of large amounts of data. The POS tagger has state-of-the-art accuracy: over 97% per-word accuracy on newspaper text, using Section 23 of the Penn Treebank as the test set [42].

The features used by the supertagger are the words and POS tags in the five-word window surrounding the target word (two words either side), and the two previously assigned lexical categories to the left. Information useful for assigning categories to rare and unknown words is provided by the POS tags, and so the morphological and word-internal features used by the POS tagger are not required by the supertagger.

The supertagging problem is much harder than POS tagging, since the CCG lexical category set has 425 labels compared to only 48 POS tags. In fact, the per-word accuracy for the supertagger is only around 92% on unseen newspaper text, which corresponds to a few mistakes per sentence, on average, and is not high enough for robust and accurate parsing [43]. The solution is to define a *multi-tagger*, in which more than one lexical category can be assigned to each word, when the context is insufficient to make a confident tagging decision. The multi-tagger uses the *forward–backward* algorithm to define a distribution over the complete lexical category set, for each word, and then that distribution is used to assign the tags; Clark and Curran [6] gives the details and describes how the supertagger interacts with the parser.

Bangalore and Joshi [3] describe supertagging as *almost parsing*, with the implication that successful supertagging leaves the parser with very little left to do. However, even with the correct lexical category sequence, there can still be a large amount of ambiguity left for the parser to resolve, due to the highly productive nature of the grammar. Furthermore, the need for a multi-tagger suggests that the task of assigning lexical categories is partly a parsing problem as well as a tagging problem, since non-local information may be needed to assign a lexical category correctly. On the other hand, Clark and Curran [35] presents an oracle experiment on newspaper text in which parsing accuracy is 98% when the parser is presented with the correct lexical categories, showing that the parsing model is able to effectively resolve whatever ambiguity remains after correct lexical category assignment (although perfect supertagging is not possible in practice).

4.2. The parser

The parser uses the standard CKY chart-parsing algorithm [44,45] described in Steedman [2]. This is a dynamic programming algorithm which fills in a parse chart bottom-up, first filling in the cells with the smallest spans (single words) and increasing the span size until the whole sentence is covered. The number of possible derivations increases rapidly with increasing sentence length, growing exponentially. This is especially true with the wide-coverage grammars used here. The standard approach is to use a *packed chart* representation, in which equivalent nodes are grouped together into equivalence classes in the chart. Nodes are equivalent if they contain the same category, the same head word associated with that category, and the same span indicating which words in the sentence are covered by the category. Such nodes are equivalent because they produce the same structures in any subsequent parsing. Clark and Curran [6] describes in detail how packed charts are used to efficiently represent a large number of CCG derivations. The packed chart representation allows efficient decoding, with the Viterbi algorithm for trees finding the highest scoring derivation. The combination of supertagging and Viterbi decoding allows the parser to process around 20 newspaper sentences per second [6].

The probability model used by the parser is a log-linear probability model, also called a Maximum Entropy model [46] or conditional random field [47] in the NLP literature. The model form is the same as that used by the taggers, except that the taggers use *local* log-linear models to make each tagging decision, whereas the parsing model is a *global* model defined over complete derivations. Clark and Curran [6] contains the details of the models, and also describes how cluster resources and a parallelised implementation of the training algorithm can be used to deal with the large parse space resulting from the wide-coverage grammar.

The parser has been evaluated on the predicate-argument dependencies in CCGbank, obtaining a labelled *F*-score of 85.5% on Section 23. We have also evaluated the parser on DepBank [48], using the Grammatical Relations scheme of Briscoe et al. [49]. The parser scores 81.1% *F*-score overall. Clark and Curran [50] gives *F*-scores broken down by relation type. Section 6 describes how the parser can be evaluated in a similar way on a biomedical resource.

5. Porting the CCG parser

Our methodology for porting the CCG parser was developed with two goals in mind, both of which reflect the fact that annotation of technical material by domain experts is a bottleneck for domain adaptation. The first goal was to leverage non-domain expertise wherever possible, in this case by using annotators familiar with CCG but not with the biomedical literature. The second goal was to obtain as much improvement in parser performance as possible from the levels of representation which are easiest to annotate, which meant training new models for the POS tagger and the supertagger but not for the parser.

In this section, we describe the process of retraining the POS tagger and the supertagger, and give results showing improved performance at these levels. Section 6 describes how these improvements affect the parser itself.

5.1. Retraining the POS tagger

POS tags are known to be important features for statistical parsing. In particular, they have been shown to be important in a CCG supertagging model: in Clark [43] the inclusion of POS features resulted in an increase from 88.1% to 90.4% in the accuracy of CCG lexical category assignment on newspaper text.

The biomedical domain poses a challenge for WSJ-trained POS taggers because of its highly specialised technical vocabulary. Lease and Charniak [13] found that a WSJ-trained POS tagger achieved only 84.6% accuracy on biomedical data, which is unacceptably low as a first stage in parsing, but that retraining the POS tagger with in-domain data increased accuracy to 95.9%. Using the output of the retrained POS tagger as input to the Charniak parser [10] also improved parsing accuracy from 78.3% to 80.8%, on a constituent-based evaluation measure, without any other changes in the parsing pipeline.

A plausible first step in porting the CCG parser was therefore to train the POS tagger on in-domain vocabulary. This task was easy to accomplish due to the ready availability of gold standard POS-tagged biomedical data.

5.1.1. Methodology

As in Lease and Charniak [13], we used the GENIA corpus [14] to train a new POS tagger model. GENIA consists of 2000 MEDLINE abstracts, or about 18,500 sentences and 440,000 words, selected from the results of a PUBMED search for the MeSH terms *human*, *blood cell*, and *transcription factor*. GENIA is one of the largest and most widely used biomedical corpora for NLP. It is manually annotated with gold standard POS tags using the PTB tag set, as well as biological named entities.

For our parsing experiments, which used a different evaluation corpus, the POS tagger model was trained on the full GENIA corpus. For the POS tagging and supertagging experiments, we held out the first 1000 sentences to use as a test set, and trained the POS tagger model on the remainder. In order to keep the data more consistent, we used only sentences within abstracts and discarded article titles.

Before training the POS tagger we converted multiple POS tags to single tags. Consistent with PTB practice, some words in the GENIA POS-tagged files are assigned two tags when the annotator considers the part-of-speech ambiguous. For example, in the noun phrase *chloramphenicol acetyl transferase*, the word *acetyl* is assigned "JJ—NN" to indicate that it could be an adjective (JJ) or a noun (NN). In such cases we arbitrarily chose one POS tag to replace the multiple tag (every instance of JJ—NN, the most common multiple tag, was converted to NN). This is not likely to have affected our results since the GENIA data is already noisy in this regard, e.g. sometimes JJ or NN alone is assigned to *acetyl* in other instances of the same noun phrase.

It should be noted that, in addition to gold standard POS tags, we used sentence boundaries and tokenisation (word boundaries) as given in GENIA. Grover et al. [51,52] observe that the level of tokenisation needed for identifying biological entities may not be the ideal level of tokenisation for a parser, and that downstream improvements can be obtained by "packaging up" scientific expressions into single tokens, but we did not investigate that here. We also accepted certain idiosyncracies in the GENIA tokenisation. A suffix that is treated as part of a compound word in one location may be treated as a separate token in another location if a named-entity boundary intervenes. For example, *cell-specific*, *tissue-specific*, and *erythroid-specific* are all treated as compound words and therefore single tokens, but *CD30-specific* and *TSHR-specific* are divided into two tokens so that the prefix can be tagged. We did delete apparently erroneous token-internal white space from a handful of tokens.

Our experiments tested two POS models, one trained on WSJ Sections 02–21 from the PTB, and the other trained on the GENIA corpus with the first 1000 sentences held out.[4] When evaluating the WSJ-trained POS tagger, we collapsed the distinction between the NNP (proper noun) and NN (common noun) tags. GENIA does not use NNP for the names of genes, proteins, or other biological entities, but only for people and locations. Since biological entities often have characteristics that lead the WSJ-trained tagger to assign the NNP tag, and the decision to use NN for biological entities is essentially a matter of convention for which we did not wish to penalize the WSJ-trained model, we converted all instances of NNP to NN in the gold standard and the POS tagged data before evaluating.

[4] We also considered a hybrid POS model trained on a combination of WSJ and GENIA data. However, preliminary experiments showed that this model performed no better than the GENIA-only POS model, either for POS tagging alone or when using the POS tagger as the first stage in the parsing pipeline.

5.1.2. Results and discussion

The results of POS tagging the first 1000 sentences from GENIA with the two models are shown in Table 1. Accuracy improved from 93.4% with the WSJ model to 98.7% with the GENIA model. In comparison, the WSJ model achieves 97.3% on WSJ Section 23 [42], so the results on biomedical data can be considered in line with state-of-the-art results for newspaper text. The results for the GENIA model are also in line with previous literature on POS tagging biomedical data, for example Tsuruoka et al. [53], in which a GENIA-trained POS tagger achieved 98.6% on a held-out section of GENIA.

5.2. Retraining the CCG supertagger

The general strategy of training a new supertagger model as part of a domain adaptation process for the CCG parser has been investigated previously in a pilot study on a different domain. Clark et al. [5] showed that the supertagger could be adapted to *What*-questions, such as *What President became Chief Justice after his presidency?*, for use in a question–answering system. Many parsers perform particularly poorly on question sentences, which are not well represented in the PTB. It was found that annotating a set of approximately 1200 questions with CCG lexical categories, and adding 1000 of these, with a weighting factor of ten, to the original training data for the supertagger, improved supertagging accuracy from 72.0% to 93.6%.

Clark et al. [5] were unable to perform a parser evaluation on the question data, since gold standard syntactic derivations were not available (only gold standard lexical category sequences). In our current experiment, we were able to evaluate parser performance to determine the effect of the new POS and supertagger models; these results are given in Section 6.

5.2.1. Methodology

In order to train a new supertagger model, the first 1000 sentences from GENIA (again excluding article titles), or approximately 118 abstracts and 27,000 words, were manually annotated with CCG lexical categories. For these experiments, the two authors served as annotators. This meant that annotation decisions which required a domain expert could not be made definitively. We approached this work as a proof-of-concept, accepting that some annotation errors would be made from the perspective of biological accuracy. However, we found that much of the annotation could be completed without domain expertise.

The GENIA sentences with gold standard POS tags were first parsed with the WSJ-trained pipeline of CCG supertagger (as a multi-tagger) and parser. The lexical categories produced by the parser were then used as a starting point for correction by the annotators. The use of the multi-tagger and the parser resulted in more accurate lexical category assignments than would have been obtained using the supertagger as a single tagger.

Both annotators independently annotated the first 100 sentences, after which disagreements were discussed and a set of annotation guidelines produced. Agreement between the annotators on the first set of 100 sentences was already 95.0%, as measured by the percentage of words on which they agreed. Disagreements were mostly related to the internal structure of noun phrases and the attachment of prepositional phrases.

Each annotator then independently annotated an additional 400 or 575 sentences, which included an overlap set of 75 sentences.

Table 1
POS tagger accuracy on GENIA data.

Model	Accuracy (%)
WSJ	93.4
GENIA	98.7

The overlap set was used to re-check agreement. This set was found to have 94.0% agreement. A few areas were identified in which the two annotators' practices had diverged; these were normalised by a combination of automatic and manual corrections. Throughout the process, corrections to previously-annotated data were made on an ongoing basis as necessary to correct mistakes or incorporate later decisions.

Clark et al. [5] reported that the question data set was annotated with lexical categories by a single annotator in less than a week. The annotation of the GENIA data, which has a significantly longer average sentence length than the questions, required approximately 3 min per sentence on average (though the speed per sentence varied considerably, with most being quite straightforward and about one in every ten requiring extra attention). Annotation speed did increase as the task became routine. The data set of 1000 sentences was processed in approximately 50 annotator-hours, suggesting that a biomedical corpus annotated with lexical categories can be produced in a reasonable time-frame, especially given gold standard POS tags.

A number of strategies were adopted for handling annotation decisions that were difficult for non-domain experts. Compound nouns and other long noun phrases were particularly difficult, especially those containing technical vocabulary. The goal in annotating a long noun phrase is to recognise which of the words modify, or describe, the head noun (normally the last word in the phrase) and which words modify other words in the phrase. In (5), all the words preceding the noun *sites* are modifiers of *sites*. Hence these are annotated with the lexical category *N*/*N*, that is, a word that takes a noun to its right and produces another noun.

two|*N*/*N* purine-rich|*N*/*N* binding|*N*/*N* sites|*N* (5)

In (6), on the other hand, the adjective *distinct* modifies the noun *epitopes*, but the adverb *functionally* modifies *distinct*, and so is annotated with the category (*N*/*N*)/(*N*/*N*) instead of *N*/*N*:

functionally|(*N*/*N*)/(*N*/*N*) distinct|*N*/*N* epitopes|*N* (6)

These particular examples were straightforward to annotate even for non-domain experts. However, the internal structure of other noun phrases was impossible to determine without specialist knowledge. For those phrases we introduced a pseudo-lexical category, *NM* (''noun modifier''), which we used in the early stages of annotation. In (7), for example, we were unable to judge whether *mobility* modifies *shift* or *assay*, and which word *electrophoretic* modifies. In (8), we were unable to decide whether the coordinated terms are *HLA-DRA* and *DQB reporter constructs*, or whether *HLA-DRA* is a kind of reporter construct, in which case the coordinated items are *HLA-DRA* and *DQB*.

electrophoretic|*NM* mobility|*NM* shift|*NM* assay|*N* (7)

HLA-DRA|*NM* and|*NM* DQB|*NM* reporter|*NM* constructs|*N* (8)

Essentially, the *NM* notation created a totally flat noun phrase with no internal structure, which represented the annotator's belief that there was not enough information available to annotate the structure.

After all sentences had been annotated, noun phrases marked with the *NM* tag were converted to proper CCG categories. Most *NM* tags were converted to *N*/*N*, with conjunctions and a few other more complex constructions being treated appropriately. The conversion thus created a simple right-branching structure for most noun phrases to which it applied. Although this may not have been correct in all instances, the result is consistent with the parser evaluation corpus, BioInfer [54,55], which treats most noun phrases as simple right-branching structures. This practice is consistent with many corpora, including the PTB, and while it represents an area in which the state-of-the-art in NLP has room for improvement, it means that our conversion strategy for *NM*s is unlikely to have had a negative effect on our parsing results. The information about which noun phrases had originally been tagged as *NM* was retained for future consultation with domain experts. All the following discussion refers to the converted version of the data.

Another type of decision that is difficult for non-domain experts is identifying the attachment point of a prepositional phrase (PP). For example, in (9), we did not know whether the phrase *via NF-kappa B activation* referred to the verb *induce* (i.e. the induction occurred via NF-kappa B activation) or to the noun *expression* (i.e. the IL-2 expression occurred via NF-kappa B activation).

Our data suggest that lipoxygenase metabolites activate ROI *formation which then induce IL-2 expression via NF-kappa B activation*. (9)

CCG lexical categories distinguish between PPs that modify verb phrases (VPs) and those that modify noun phrases (NPs), but do not indicate which verb phrase or noun phrase is modified if there are multiple possibilities; these attachment decisions are left to the parser. The strategy we adopted for cases where the status of the PP as a VP or NP modifier was unclear was to use the annotator's best guess based on a layperson's interpretation of the terms involved, and to default to coding the preposition as an NP modifier in the most difficult cases.

We also developed tagging conventions for the internal structure of common phrases such as *kappa B*, *T cell*, and *in vitro*; distance expressions such as *1.3 kilobases upstream of*; and a variety of other constructions. For this annotation project our goal was consistency in the annotation, even when this may have led to errors from a biological perspective; we leave the consultation of domain experts on uncertain cases to future work.

When assigning CCG lexical categories we attempted to follow CCGbank conventions wherever possible. This entailed avoiding the introduction of new complex CCG categories not appearing in CCGbank unless absolutely necessary, and annotating common verbs with the same subcategorisation frames found in newspaper text unless it was clear that their use in biomedical text was different. These practices ensured consistency with CCGbank and gave the WSJ-trained supertagger, which provides the baseline performance for our experiments, the best possible chance on the biomedical data.

It is worth noting that the final version of the annotated data set had 93.9% agreement (by word) with the lexical categories originally assigned by the parser. This suggests that the CCG supertagger and parser are already well-suited to the types of sentences observed in biomedical literature (at least when gold standard POS tags are used).

Our experiments tested three supertagger models. The first was the standard model trained on WSJ Sections 02–21 of CCGbank. The second model, GENIA, was trained on the 1000 annotated GENIA sentences, with no newspaper text. The third model was a hybrid model trained on a combination of WSJ Sections 02–21 of CCGbank and the annotated sentences from GENIA.

Adding the biomedical data to the newspaper data serves two purposes for the new supertagger model. First, it enhances the lexicon by adding new words and word-category pairs that appear in the biomedical data but not the newspaper data. Second, it provides additional, in-domain data for training the model. For this latter purpose we used a weighting factor of 10 for the biomedical data, i.e. 10 copies of the GENIA sentences were added to a single copy of the WSJ data. The weighting factor is necessary to keep the 40,000 newspaper sentences from dominating the 1000 GENIA sentences when training the model. We chose the factor of 10 based on preliminary experiments.

5.2.2. Results and discussion

We evaluated the supertagger models using 10-fold cross-validation. The 1000 annotated sentences from GENIA were divided into 10 sequential sets of 100 sentences each, and a supertagger model was trained with each test set held out. The accuracies reported here are averages of the scores on the 10 folds, evaluated against the gold standard lexical categories. The accuracy reported for the WSJ model is also an average over the same 10 folds.

The first step in the supertagger experiments was to POS tag the test data. For the WSJ supertagger, we used both the WSJ and the GENIA POS taggers, in order to investigate the effect of improved POS tagging accuracy alone on the supertagger. For the GENIA and hybrid supertaggers, we used only the GENIA POS tagger. This reflects real-world conditions, since it would be unusual to have gold standard lexical categories for biomedical data with which to train a supertagger model, without also having gold standard POS tags available to train a POS tagger.

The results of supertagging the 1000 GENIA sentences with the three supertagger models (and two POS taggers) are shown in Table 2. The full WSJ pipeline achieved 89.0% accuracy. Simply using the GENIA POS tagger as the first stage in the pipeline gave an absolute improvement of 2.2 percentage points, to 91.2%. Based on this improvement, we expect that simply using the GENIA POS tagger will also yield an improvement in parsing, since parser performance is dependent on supertagger performance, and this turns out to be the case, as discussed in Section 6.

The GENIA supertagger model performed essentially no better than the WSJ supertagger model with the GENIA POS tagger. We hypothesize that this is due to a lack of coverage on common grammatical constructions, resulting from the small size of the training set. The best result was obtained with the hybrid supertagger model, with 93.0% accuracy. This result is comparable to that reported for a supertagger trained and tested on newspaper text [6].

In practice, the accuracy of the lexical category assignments made by the multi-tagger in combination with the parser would be higher than the figures given here for the single tagger. However, evaluating the single tagger is an easy way to measure improvements in the supertagger independent of the parser. Improvements in single tagging also imply improvements in multi-tagging, since the underlying supertagger model has improved for biomedical text.

We next investigated whether the improvements in supertagger accuracy were due to improvements on any particular lexical categories. Table 3 shows the precision (P), recall (R), and F-score by lexical category for the full WSJ and full biomedical (GENIA POS tagger and hybrid supertagger) pipelines, for non-punctuation categories appearing at least 100 times in the gold standard data. These figures were calculated by concatenating the 10 tagged folds and comparing them with the gold standard data. F-score was calculated as $2 \times P \times R/(P + R)$.

An improvement can be seen across the board, with no particular subset of categories being responsible for the overall increase in accuracy, though a few did show dramatic improvements. There was a sizeable improvement for the category $(S[dcl]\backslash NP)/NP$, representing a basic transitive verb. The categories $(NP\backslash NP)/NP$ and $((S\backslash NP)\backslash(S\backslash NP))/NP$, representing PP modifiers of NPs and VPs, respectively, also showed an improvement, mostly in precision. This means that the supertagger got better at distinguishing potential PP attachment sites.

The categories $(S[pss]\backslash NP)/PP$, $(S[dcl]\backslash NP)/PP$, and PP/NP all improved, which means that the supertagger got better at recognising prepositional phrase arguments of verbs (the first two categories are subcategorisation frames for verbs that take PPs as arguments, and the third is the category of the PP argument itself). The category $S[adj]\backslash NP$, representing an adjectival predicate, is another with a notable improvement. Our hypothesis is that adjectival predicates were being better distinguished from passives, since the category $S[pss]\backslash NP$, which is often confused with $S[adj]\backslash NP$ by the supertagger, also improved.

Finally, there was a large improvement (36.1 percentage points) for the category $(N/N)/(N/N)$, despite the fact that we did not always annotate the internal structure of noun phrases. This shows that when the training data includes some information about noun phrase structure, the supertagger is able to make use of it.

Table 4 shows the five categories accounting for the largest number of errors in the WSJ and biomedical pipelines, and the two categories that each one was most often confused with. The number of errors is the number of times the category appeared in the gold standard but was incorrectly recognised by the supertagger (i.e. recall errors). The categories appearing in the rightmost column are the ones output by the supertagger instead. The category N/N accounted for the most errors both before and after porting, but the porting process brought that number down by almost 50%, from 543 to 294. Confusion of N/N with $(N/N)/(N/N)$ increased from 5 instances to 33, though, showing that the retrained supertagger may have incorrectly assigned internal structure to some noun phrases. However, errors in identifying genuine instances of $(N/N)/(N/N)$ also showed a notable decrease, from 286 to 149.

The distinction between prepositions that modify VPs – $((S\backslash NP)\backslash(S\backslash NP))/NP$ – and NPs – $(NP\backslash NP)/NP$ – remained among the most difficult even for the biomedical pipeline. The total number of errors on these two categories combined did show a decrease of about 16%, however. It is also noteworthy that both before and after porting, in addition to the two preposition categories, the three categories with the most errors were ones related to noun phrase structure.

Table 2
Supertagger accuracy on 1000 GENIA sentences with gold standard CCG lexical categories (average of 10-fold cross-validation).

Model	Accuracy (%)
WSJ (WSJ POS)	89.0
WSJ (GENIA POS)	91.2
GENIA (GENIA POS)	91.3
Hybrid (GENIA POS)	93.0

6. Parser evaluation

So far we have demonstrated that new POS tagger and supertagger models trained on biomedical data both contribute to improved supertagging accuracy on a subset of GENIA. We next evaluated the results of these changes on the parser itself, using the BioInfer corpus [55]. The standard newspaper-trained parser model was used for this evaluation, so the improvement in parsing accuracy is due only to the new POS tagger and supertagger models.

BioInfer is annotated with grammatical relations in the Stanford format [56]. In this section, we describe grammatical relation schemes and the BioInfer corpus. We then discuss how the CCG parser output is mapped to the Stanford format, and the results of the evaluation.

6.1. Methodology

6.1.1. Grammatical relation schemes

State-of-the-art parsers are based on a wide range of grammar formalisms, including CCG, HPSG [28], and the phrase-structure grammar of the PTB [10,11]. Most parsers produce formalism-specific output, and the majority of parser evaluations to date have used test sets drawn from the same corpus used to develop the parser. These practices have made meaningful comparison of parsers difficult.

Table 3
Supertagger accuracy by lexical category.

Category	Freq	WSJ pipeline			Biomedical pipeline			Diff
		P	*R*	*F*	*P*	*R*	*F*	
N	6382	91.3	95.4	93.3	95.1	97.3	96.2	2.9
N/N	5855	91.1	90.7	90.9	94.0	95.0	94.5	3.6
NP[nb]/N	2160	97.4	99.7	98.5	99.4	99.9	99.6	1.1
(NP\ NP)/NP	2132	82.5	92.5	87.2	87.5	94.7	90.9	3.7
conj	899	99.6	98.4	99.0	98.7	99.3	99.0	0.0
((S\ NP)\ (S\ NP))/NP	689	69.6	65.2	67.3	76.3	67.8	71.8	4.5
(S[dcl]\ NP)/NP	522	81.6	82.4	82.0	91.5	92.3	91.9	9.9
PP/NP	482	72.0	68.9	70.4	83.3	84.0	83.7	13.3
S[pss]\ NP	405	82.8	85.4	84.1	88.1	91.4	89.7	5.6
(N/N)/(N/N)	360	75.5	20.6	32.3	82.1	58.6	68.4	36.1
(S[dcl]\ NP)/(S[pss]\ NP)	339	94.3	97.3	95.8	97.1	97.1	97.1	1.3
S[em]/S[dcl]	247	95.9	95.5	95.7	98.0	97.2	97.6	1.9
NP	238	77.0	95.8	85.4	97.0	96.2	96.6	11.2
(S\ NP)\ (S\ NP)	201	76.6	89.6	82.6	80.8	90.0	85.2	2.6
(S[b]\ NP)/NP	182	79.8	95.6	87.0	95.6	94.5	95.0	8.0
(S[dcl]\NP)/S[em]	173	97.5	91.3	94.3	98.8	98.3	98.6	4.3
(S[dcl]\NP)/(S[adj]\NP)	167	90.6	86.8	88.7	93.3	92.2	92.8	4.1
(S[ng]\NP)/NP	161	79.7	90.1	84.5	97.3	88.2	92.5	8.0
(S[to]\NP)/(S[b]\NP)	160	87.3	98.8	92.7	100.0	98.1	99.1	6.4
S[adj]\NP	154	65.5	61.7	63.5	78.9	82.5	80.6	17.1
(S[pss]\NP)/PP	147	67.6	63.9	65.7	76.7	78.2	77.4	11.7
(S\NP)/(S\NP)	139	85.6	85.6	85.6	91.5	85.6	88.5	2.9
(S[dcl]\NP)/(S[b]\NP)	139	97.2	100.0	98.6	99.3	99.3	99.3	0.7
(NP\NP)/(S[dcl]\NP)	138	99.2	95.7	97.4	99.3	98.6	98.9	1.5
(S[dcl]\NP)/PP	123	75.0	51.2	60.9	85.4	90.2	87.7	26.8
S/S	112	99.0	87.5	92.9	95.3	90.2	92.7	−0.2
(S/S)/NP	102	85.6	93.1	89.2	89.1	96.1	92.5	3.3

The parsing community has not yet converged on a single standard for comparative evaluation of parsers. However, a category of evaluation schemes based on grammatical relations [57,58] holds promise for several reasons. First, such schemes are relatively independent of any particular parser or linguistic theory. Second, they transparently encode syntactic/semantic relations, which makes it possible to evaluate a parser's performance on a particular linguistic construction, or its suitability for a particular application [57,58]. Clegg and Shepherd [59] discuss the benefits of grammatical relation schemes in the biomedical domain, noting that they can help identify parsers that are good at producing the types of relations used in information extraction tasks.

Following Carroll et al. [58], we use the term *grammatical relation* (GR) to refer to a labelled syntactic dependency between a head and a dependent. Fig. 3 shows the Stanford dependencies for the phrase *absence of alpha-syntrophin leads to structurally aberrant neuromuscular synapses*. The label names a syntactic relation, e.g. 'prepositional modifier', and the words in the relation are the head and dependent. The word *to* is a dependent of the verb *leads*, for example, but also a head with *synapses* as its dependent. The word *synapses* itself is the head of the noun phrase *structurally aberrant neuromuscular synapses*.

Table 4
Lexical categories accounting for the most supertagger errors, and categories they were frequently confused with.

Category	Errs	Freq confused with
WSJ pipeline		
N/N	543	*N, NP*
N	292	*N/N, N\N*
(N/N)/(N/N)	286	*N/N, (S[dcl]\NP)/NP*
((S\NP)\(S\NP))/NP	240	*(NP\NP)/NP, PP/NP*
(NP\NP)/NP	159	*((S\NP)\(S\NP))/NP, PP/NP*
Biomedical pipeline		
N/N	294	*N, (N/N)/(N/N)*
((S\NP)\(S\NP))/NP	222	*(NP\NP)/NP, PP/NP*
N	171	*N/N, S[adj]\NP*
(N/N)/(N/N)	149	*N/N, N*
(NP\NP)/NP	114	*((S\NP)\(S\NP))/NP, PP/NP*

Parser evaluation using a GR scheme is performed by calculating precision, recall, and *F*-score measures against gold standard dependencies. We use the strictest form of evaluation, in which the label, head, and dependent must all be correct for the dependency to be marked as correct. However, it is also possible to evaluate based on unlabelled dependencies or to relax the evaluation in other ways [58].

In addition to the advantages already mentioned, a GR scheme was a natural choice for evaluating the CCG parser since it already produces GR output in the format of Briscoe and Carroll [60] (hereafter BC), so that no major changes to the architecture of the parser were required. However, considerable work was still required in implementing the mapping from the CCG representation to the Stanford GRs, because of the differences between the Stanford format and BC; the mapping is described in Section 6.1.3.

GR schemes contrast with constituent-based evaluation schemes, of which the best known is PARSEVAL [61]. Constituent-based evaluation has long been the standard for PTB parsers. Such methods count the number of labelled, bracketed nodes in the parsed data that correspond to nodes in the gold standard trees. For a node to be marked as correct, the left and right edges of the phrase it dominates must be identified correctly, as well as

nsubj(leads, absence)	[nominal subject]
prep(absence, of)	[prepositional modifier]
pobj(of, alpha-syntrophin)	[object of preposition]
prep(leads, to)	[prepositional modifier]
pobj(to, synapses)	[object of preposition]
amod(synapses, neuromuscular)	[adjectival modifier]
amod(synapses, aberrant)	[adjectival modifier]
advmod(aberrant, structurally)	[adverbial modifier]

Fig. 3. Grammatical relations in Stanford format for the phrase *absence of alpha-syntrophin leads to structurally aberrant neuromuscular synapses*.

the node label (e.g. the left and right edges of a noun phrase must be identified, as well as the NP label of the node dominating the phrase).

Parser evaluation scores obtained using constituent-based schemes and GR schemes are not directly comparable, because the evaluation methods are so different. For this reason, it is not possible for the results in this paper to be compared with the recent porting work of McClosky and Charniak [21], for example, who evaluate the Charniak parser using a constituent-based evaluation.

It has been argued that constituent-based metrics are more forgiving than GR schemes [58]. State-of-the-art parsers often produce accuracies above 90% on the PTB using PARSEVAL metrics, while the state-of-the-art for GR based evaluation on the PTB is only slightly above 80%. In the biomedical domain, Clegg and Shepherd [59] performed an evaluation of several (non-adapted) parsers using both constituency-based and GR based evaluation; the average *F*-score for the former was 76.8% and for the latter 71.3%. When comparing evaluation scores, it must also be taken into account whether the evaluation is in the native format of a parser or whether conversion to the evaluation format has been necessary, in which case some loss should be attributed to the conversion [50].

6.1.2. Evaluation corpus

A number of biomedical corpora have been developed in recent years, including the GENIA treebank [12] and the PennBioIE corpus [62], both of which are manually annotated with labelled syntactic bracketing in PTB style. BioInfer [55] is the first available manually corrected GR corpus in the biomedical domain. The evaluation in Clegg and Shepherd [59] used a corpus in the Stanford format derived from the GENIA treebank, but while the treebank itself was manually annotated, the conversion to Stanford format was not manually corrected.

BioInfer contains 1100 sentences, or about 34,000 words, from a set of abstracts retrieved from PUBMED focusing on the topic of protein–protein interaction. The corpus was originally developed in the format of Link Grammar [63] by Pyysalo et al. [54], and converted to the Stanford GR format of de Marneffe et al. [56] by an automatic process followed by manual correction [55]. The Stanford format is based on the GR schemes of Briscoe and Carroll [60] and King et al. [48] but differs from them in several ways; in particular, the relation types in Stanford are more fine-grained than those in BC.

BioInfer uses the "uncollapsed" variant of the Stanford scheme, which includes a slightly more verbose representation of coordinations and prepositional phrases than the collapsed variant (see [56] for details). BioInfer also makes a small number of modifications to the scheme, of which the most significant is that prenominal modifiers are not distinguished as belonging to the relation types *amod* (adjectival modifier) and *nn* (nominal modifier, e.g. in a compound noun), but rather are all labelled with a new relation type, *nmod* (nominal modifier).

Pyysalo et al. [55] have evaluated two parsers, Charniak–Lease [13] and BioLG [64], against BioInfer, and we were able to use these results for a comparative evaluation with the CCG parser. We used the same division of BioInfer into 600 development sentences and 500 test sentences as in Pyysalo et al. [55].

It is worth noting that although GENIA and BioInfer both consist of MEDLINE abstracts, they focus on different subdomains in the biomedical literature. Differences in vocabulary might therefore be expected, and indeed the BioInfer test set has an 11.7% unknown word rate respect to the GENIA corpus, significantly higher than a held-out section of GENIA itself. This difference may have had implications for the performance of the POS tagger and supertagger on BioInfer (as, in fact, might differences in tokenisation or other coding conventions between the two corpora). We were unable to test this since gold standard POS tags and lexical categories are not available for BioInfer, but, as our experiments show, using the GENIA-trained POS and supertaggers did yield a significant improvement in parser performance on BioInfer despite potential effects of the different subdomains.

6.1.3. Mapping the parser output to Stanford dependency format

Although no architectural changes to the CCG parser were required to produce output in Stanford format, the mapping required a significant investment of time. Clark and Curran [50] provide a full account of the process of mapping from CCG dependencies to GRs in the BC format. A number of complexities were involved, including the fact that the mapping is many-to-many, and that the valency of the CCG and BC dependencies sometimes differ. Since there is no direct mapping from one GR format to another – recall that Stanford is more fine-grained than BC – many of the decisions involved had to be made anew for Stanford.

CCG dependencies are transformed into GRs in two stages. The first stage involves a mapping between the CCG dependencies and the GRs, defined in a file known as the "markedup" file in the parser implementation. In the second stage, a post-processing script is used to process the GRs, correcting some remaining differences.

The native output format of the CCG parser is CCG dependency structures, which are defined in terms of the argument slots in CCG lexical categories. For example, in the phrase *studies demonstrate that...*, the dependency in (10) encodes the fact that *studies* is the subject of *demonstrate*.

$$\langle demonstrate, (S[dcl]\backslash NP_1)/S[em]_2, 1, studies\rangle \quad (10)$$

The category for *demonstrate*, $(S[dcl] \setminus NP)/S[em]$, has two argument slots: one for a clause ($S[em]$) and the other for a subject noun phrase (NP). The result is a declarative sentence ($S[dcl]$). The subject noun phrase slot, subscripted 1, is filled by *studies*. This information must be converted into the Stanford dependency in (11), with the label nsubj (nominal subject). The CCG dependency structure containing the other argument of *demonstrate* is converted to a second Stanford dependency.

$$(\text{nsubj demonstrate studies}) \quad (11)$$

In the post-processing step, a set of general rules is applied to bring the output of the conversion defined in the markedup file further in line with the Stanford format. For example, numerical modifiers of nouns (e.g. *three proteins*) are identified by their POS tag. It turned out that the post-processing script was even more important for the Stanford output than it had been for the BC output, making a difference of more than 10 percentage points in the *F*-score. We hypothesize that this is due to the prevalence of complicated coordination and apposition structures in biomedical text, since dependencies related to both of these structures are modified in the script.

It is clear that the mapping from the native parser format to the evaluation format has a significant effect on the final results, and the accuracy of the mapping itself imposes an upper bound on the results which can be expected from the parser. This is discussed further in Section 6.2.

6.2. Results and discussion

The results of parsing the BioInfer test set with three different pipelines are shown in Table 5. The first row gives the results for the original, newspaper-trained pipeline; the second row gives the results for the pipeline with the GENIA POS tagger and the newspaper-trained supertagger; and the third row gives the results for the final biomedical pipeline, with the GENIA POS tagger and the hybrid supertagger. In addition to precision, recall, and *F*-score on grammatical relations, we report coverage, that is, the percentage

Table 5
CCG parser accuracy on BioInfer.

Model	Cov (%)	P	R	F
WSJ POS, WSJ super	97.2	76.4	75.6	76.0
GENIA POS, WSJ super	99.0	80.7	80.1	80.4
GENIA POS, hybrid super	99.8	81.8	81.3	81.5

of sentences that received a full analysis by the parser. Following the practice of S. Pyysalo for BioInfer evaluation (pers. comm.), we ignored a small set of difficult tokens, for example bibliographic references, which are not syntactically connected to the rest of the sentence and do not participate in any relations in the BioInfer corpus.

Table 5 shows that retraining the POS tagger resulted in an absolute increase of 4.4 percentage points in *F*-score, and retraining the supertagger resulted in an additional increase of 1.1 points. In addition, there was an improvement in coverage. The original pipeline had 14 parse failures (where the parser could not generate a syntactic structure for the entire sentence) out of 500 sentences, whereas the final pipeline had only one. We attribute the greater number of parse failures with the original pipeline to the fact that the grammar lacked coverage of some of the syntactic structures used in biomedical data. As in the supertagger experiments, we did not consider a pipeline including the GENIA supertagger with the newspaper-trained POS tagger, since it would be unexpected in a real-world situation to have gold standard lexical categories available for biomedical text without having gold standard POS tags. We also did not consider the supertagger model trained only on GENIA, since it was outperformed by the hybrid model in the supertagging experiments.

In Table 5, precision, recall, and *F*-score are measured over sentences which were parsed successfully. Table 6 shows the results with the recall score (and *F*-score) modified to reflect the parse failures. Here, we see an absolute increase in *F*-score of 5.3 percentage points from retraining the POS tagger and an additional increase of 1.4 points from retraining the supertagger.

To interpret these results correctly, it must be remembered that there is an upper bound on the *F*-score which the parser can achieve on GRs, due to the loss inherent in the conversion from the CCG parser's native format to the GR format. Clark and Curran [50] calculated the upper bound of the parser on newspaper text, using the BC scheme, to be 84.8%. It is not possible to obtain an upper bound for the biomedical GR corpus, because this would require gold standard CCG representations for the biomedical data, which we do not have, but we hypothesise that the upper bound would be roughly in the same range as that for the newspaper data. With these figures in mind, we can see that the *F*-score of 81.4% for the biomedical pipeline on BioInfer is in the same range as the *F*-score of the original pipeline on newspaper text, which was 81.8% [50], although the results are not directly comparable since they are based on different GR schemes for evaluation.

Table 7 compares the results of the final CCG parser pipeline with the results for the Charniak–Lease parser and the BioLG parser reported in Pyysalo et al. [55]. Both of these parsers are adapted to the biomedical domain to at least some degree. The Charniak–Lease parser is the Charniak constituent parser with a POS tagger trained on

Table 6
CCG parser accuracy on BioInfer, with recall reflecting parse failures.

Model	P	R	F
WSJ POS, WSJ super	76.4	73.2	74.7
GENIA POS, WSJ super	80.7	79.4	80.0
GENIA POS, hybrid super	81.8	81.1	81.4

Table 7
Parser accuracy on BioInfer.

Parser	P	R	F
CCG parser	81.8	81.1	81.4
Charniak–Lease	78.4	79.9	79.4
BioLG	79.6	76.1	77.8

GENIA [13]. Tools provided with the Stanford parser were used to convert the output of the Charniak–Lease parser to the Stanford dependency format. The BioLG parser [64] is an adaptation of the LG parser, a rule-based, non-probabilistic dependency parser based on Link Grammar [63], to the biomedical domain, incorporating a POS tagger trained on GENIA as well as other modifications.

The CCG parser achieves an *F*-score on BioInfer two points above that of the next best performer, Charniak–Lease. Since the methods used to convert native parser output to the Stanford format differ widely between these two parsers, it is not possible to draw firm conclusions about their relative performance based on these figures. However, these results provide strong evidence that our porting method was successful.

Table 8 shows the results for the original and final pipelines on the 600 BioInfer development sentences, broken down by dependency type. The frequency column gives the number of times each dependency type occurs in the gold standard. This breakdown allows us to see the source of the performance improvements, and which dependency types are the most difficult for the parser. For this analysis we evaluated only sentences where both pipelines had coverage in order to have a fair comparison. Rare dependency types are omitted from the table, as is *dep*, a generic dependency type used when the correct type cannot be determined. The dependency types are arranged in rough equivalence classes to the BC dependency types, in order to facilitate comparison with the parser's performance on newspaper text. It should be noted, though, that the BC and Stanford schemes are sufficiently different that such a comparison is approximate at best, since the relationship between the dependency types in the two schemes is many-to-many.

As with supertagging performance (Table 3), the improvement in parser performance was due not to a large improvement on any particular dependency type, but rather to a general improvement across the board. A clear increase in accuracy can be seen in dependencies representing the core argument structure of a sentence, such as *nsubj* (nominal subject) and *dobj* (direct object), which are crucial for information extraction applications.

The link between improved POS and supertagging and increased accuracy in recovery of these core dependencies is quite direct. For example, in the sentence *Conversely, inhibition of LIMK's activity by expressing a dominant negative construct, LIMK1-, or expression of the constitutively active S3A cofilin mutant induces loss of actin filaments at the phagocytic cup and also inhibits phagocytosis.*, the WSJ POS tagger assigned the NNS (plural noun) tag to the word *induces*. As a result, the supertagger treated it as a nominal modifier of *loss*, and the parser did not recover the direct object relation between *induces* and *loss*. The GENIA POS tagger assigned the correct POS tag (VBZ, present tense verb) and the biomedical pipeline recovered the dependency correctly.

Increased accuracy can also be seen on dependencies such as *aux* (auxiliary verb), *xsubj* (controlling subject of an infinitival clause), *num* (numeric modifier), *advmod* (adverbial modifier), and *xcomp* (finite clausal complement). In the sentence *However, the recombinant fusion protein containing wild-type beta-catenin precipitated alpha-catenin from these cells.*, the WSJ pipeline incorrectly generated an *advmod* relation between *containing* and *However*, rather than between *precipitated* and *However*, because the POS tagger failed to recognize *precipitated* as a verb. The biomedical pipeline recovered the dependency correctly.

Table 8
CCG parser accuracy on BioInfer development data, by dependency type. The leftmost column shows the nearest equivalent BC relation.

BC	Rel	Freq	WSJ pipeline			Bio pipeline			Diff
			P	*R*	*F*	*P*	*R*	*F*	
conj	cc	749	66.34	63.68	64.99	69.09	67.16	68.11	3.12
	conj	1045	52.97	46.03	49.26	55.98	52.82	54.36	5.10
ta	appos	279	14.90	11.11	12.73	19.43	12.19	14.98	2.25
aux	aux	210	85.22	93.33	89.09	97.50	92.86	95.12	6.03
	auxpass	248	97.97	97.18	97.57	98.73	93.95	96.28	−1.29
det	det	1255	89.93	91.08	90.50	92.45	92.67	92.56	2.06
dobj	dobj	492	77.55	84.96	81.09	87.52	91.26	89.35	8.26
	pobj	2116	88.73	89.32	89.02	90.59	92.34	91.46	2.44
ncsubj	nsubj	612	73.29	82.52	77.63	81.26	90.69	85.71	8.08
	nsubjpass	229	78.42	82.53	80.43	76.57	79.91	78.21	−2.22
	xsubj	54	44.62	53.70	48.74	61.22	55.56	58.25	9.51
ncmod	advmod	311	70.55	73.95	72.21	72.81	77.49	75.08	2.87
	neg	36	83.33	83.33	83.33	83.33	83.33	83.33	0.00
	nmod	3177	85.32	83.44	84.37	88.73	88.20	88.46	4.09
	num	276	70.27	65.94	68.04	84.46	76.81	80.46	12.42
	poss	60	93.33	70.00	80.00	97.78	73.33	83.81	3.81
	preconj	26	72.73	30.77	43.24	59.26	61.54	60.38	17.14
	prep	2077	78.08	78.05	78.06	77.82	78.72	78.27	0.21
cmod	advcl	79	77.55	48.10	59.38	78.00	49.37	60.47	1.09
	rcmod	105	71.96	73.33	72.64	68.22	69.52	68.87	−3.77
	ref	105	75.76	71.43	73.53	71.00	67.62	69.27	−4.26
	rel	105	90.43	80.95	85.43	88.66	81.90	85.15	−0.28
ccomp	ccomp	114	73.53	65.79	69.44	77.55	66.67	71.70	2.26
	complm	111	81.90	77.48	79.63	84.91	81.08	82.95	3.32
	mark	66	78.72	56.06	65.49	78.43	60.61	68.38	2.89
xcomp	cop	131	88.55	88.55	88.55	83.10	90.08	86.45	−2.10
	xcomp	74	71.79	75.68	73.68	91.53	72.97	81.20	7.52
xmod	amod	17	14.63	35.29	20.69	38.10	47.06	42.11	21.42
	partmod	201	53.31	64.18	58.24	57.98	68.66	62.87	4.63
	Total	14,525	77.81	77.00	77.40	81.06	80.32	80.69	3.29

The biomedical pipeline does show a decrease in accuracy on two sets of dependency types. There is a small decrease on *auxpass* (passive auxiliary) and *nsubjpass* (subject of passive), meaning that the new pipeline performed slightly worse than the original pipeline on passive constructions in general. We believe that many of these cases can be attributed to confusion between adjectival and verbal passives. It was mentioned in Section 5.2.2 that the retrained supertagger showed improved recognition of both types of passives. However, adjectival passives were sometimes overgenerated. In the sentence *Interestingly, the regions that show the most structural diversity are located at or near the actin-binding site of profilin.*, for example, the GENIA POS tagger assigned the tag JJ (adjective) to the word *located*, and the supertagger and parser treated it as an adjective. The original WSJ pipeline correctly assigned the POS tag VBN (verbal past participle).

A decrease in accuracy can also be seen on the dependencies *rcmod*, *ref*, and *rel*, all of which are related to relative clause constructions. One possible explanation for the superior performance of the WSJ pipeline on relative clauses is that this pipeline makes use of the NNP (proper noun) POS tag, which, as discussed in Section 5.2.2, is not used in the GENIA corpus for biological entities. In a phrase such as *The domains in CBP that are involved in CREB binding and transcriptional activation*, the WSJ POS tagger assigned the NNP tag to *CBP*, and the parser may have used this information to determine that the relative clause (*that are involved...*) modifies *domains*, not *CBP*; the biomedical pipeline incorrectly associated it with *CBP*.

In Table 8 we showed the performance of the WSJ pipeline and the biomedical pipeline, both on biomedical data. We also considered whether it was possible to make a comparison between the performance of the new pipeline on biomedical data, and that of the original pipeline on newspaper data, i.e. comparing each pipeline on its target domain. This is difficult because of the differences between the Stanford and BC formats. One difference between the two pipelines does stand out, which is that the *F*-score of the original pipeline on the *conj* (conjunction) relation in newspaper text was 77.5 [6], while the *F*-scores of the new pipeline on *conj* and *cc*, which are both related to coordination constructions in Stanford, were much lower for the biomedical data. A possible interpretation is that conjunctions are more difficult to parse in biomedical than newspaper data. We attribute this difference to the prevalence of potentially ambiguous coordinations involving long noun phrases in biomedical text. For example, in the phrase *Electrical stimulation of cardiocyte contraction did not enhance alpha-cardiac actin or myosin heavy chain (alpha + beta) mRNA transcript levels*, the parser mistakenly coordinates *actin* and *myosin* rather than *actin* and *chain*. Improving performance on such constructions may turn out to require new parsing models as well as new models at the lower levels of representation.

7. Conclusion

We have presented a method for adapting a lexicalized-grammar parser to the biomedical domain which leverages the parser's existing wide-coverage grammar based on newspaper text, requiring limited additional resources for annotation of biomedical data. Our first result is that a large improvement in accuracy can be obtained simply by using a POS-tagger trained on biomedical data, which is in line with previous results. We further showed that retraining a supertagger, which operates at an intermediate level of linguistic

representation, improves parsing accuracy further, and that manually annotated data for the supertagger can be obtained relatively cheaply. The non-trivial process of evaluating a parser outside of its native grammatical formalism was described, and our results on the BioInfer corpus are the highest presented in the literature on this corpus. Our conclusion is that porting newspaper parsers to the biomedical domain, at least for parsers which use lexicalized grammars, may not be as difficult as first thought. We propose that the main difference between newspaper and biomedical text lies in the lexical items, not the syntax, and that accurate identification of parts of speech and lexical categories for biomedical vocabulary goes a long way towards enabling accurate parsing in this domain, given an accurate model for newspaper text. The CCG parser is available at http://svn.ask.it.usyd.edu.au/trac/candc/wiki.

References

[1] Marcus Mitchell, Santorini Beatrice, Marcinkiewicz Mary. Building a large annotated corpus of English: the Penn Treebank. Comput Linguist 1993;19(2):313–30.

[2] Steedman Mark. The syntactic process. Cambridge, MA: The MIT Press; 2000.

[3] Bangalore Srinivas, Joshi Aravind. Supertagging: an approach to almost parsing. Comput Linguist 1999;25(2):237–65.

[4] Clark Stephen, Curran James R. The importance of supertagging for wide-coverage CCG parsing. In: Proceedings of COLING-04, Geneva, Switzerland, 2004. p. 282–8.

[5] Clark Stephen, Steedman Mark, Curran James R. Object-extraction and question-parsing using CCG. In: Proceedings of the EMNLP Conference, Barcelona, Spain, 2004. p. 111–8.

[6] Clark Stephen, Curran James R. Wide-coverage efficient statistical parsing with CCG and log-linear models. Comput Linguist 2007;33(4):493–552.

[7] Gildea Daniel. Corpus variation and parser performance. In: 2001 conference on empirical methods in natural language processing (EMNLP), Pittsburgh, PA, 2001.

[8] Clegg Andrew B, Shepherd Adrian J. Evaluating and integrating treebank parsers on a biomedical corpus. In: Proceedings of the association for computational linguistics 43rd annual meeting workshop on software, Ann Arbor, US, 2005.

[9] Bikel Daniel M. Intricacies of collins parsing model. Comput Linguist 2004;30(4):479–511.

[10] Charniak Eugene. A maximum-entropy-inspired parser. In: Proceedings of the 1st meeting of the NAACL, Seattle, WA, 2000. p. 132–9.

[11] Collins Michael. Head-driven statistical models for natural language parsing. Comput Linguist 2003;29(4):589–637.

[12] Tateisi Yuka, Yakushiji Akane, Ohta Tomoko, Tsujii Jun'ichi. Syntax annotation for the GENIA corpus. In: Proceedings of the companion volume of the second international joint conference on natural language processing (IJCNLP-05), Jeju Island, Korea, 2005. p. 222–7.

[13] Lease Matthew, Charniak Eugene. Parsing biomedical literature. In: Proceedings of the second international joint conference on natural language processing (IJCNLP-05), Jeju Island, Korea, 2005.

[14] Kim Jin-Dong, Ohta Tomoko, Teteisi Yuka, Tsujii Jun'ichi. GENIA corpus – a semantically annotated corpus for bio-textmining. Bioinformatics 2003;19:i180–2.

[15] Hara Tadayoshi, Miyao Yusuke, Tsujii Jun'ichi. Adapting a probabilistic disambiguation model of an HPSG parser to a new domain. In: International joint conference on natural language processing (IJCNLP), Jeju Island, Korea, 2005. p. 199–210.

[16] Bacchiani Michiel, Roark Brian, Saraclar Murat. Language model adaptation with MAP estimation and the perception algorithm. In: Proceedings of the human language technology conference and meeting of the North American chapter of the association for computational linguistics (HLT-NAACL), 2004. p. 21–4.

[17] Bacchiani Michiel, Riley Michael, Roark Brian, Sproat Richard. MAP adaptation of stochastic grammars. Comput Speech Lang 2006;20:41–68.

[18] McClosky David, Charniak Eugene, Johnson Mark. Reranking and self-training for parser adaptation. In: Proceedings of the association for computational linguistics (COLING-ACL), Sydney, Australia, 2006.

[19] McClosky David, Charniak Eugene, Johnson Mark. Effective self-training for parsing. In: Proceedings of the conference on human language technology and North American chapter of the association for computational linguistics (HLT-NAACL), Brooklyn, New York, 2006.

[20] Foster Jennifer, Wagner Joachim, Seddah Djamé, van Genabith Josef. Adapting WSJ-trained parsers to the British National Corpus using in-domain self-training. In: Proceedings of the 10th international conference on parsing technologies (IWPT), Prague, 2007.

[21] McClosky David, Charniak Eugene. Self-training for biomedical parsing. In: Proceedings of the association for computational linguistics (ACL-08, short papers), Columbus, Ohio, 2008, p. 101–104.

[22] Steedman Mark, Osborne Miles, Sarkar Anoop, Clark Stephen, Hwa Rebecca, Hockenmaier Julia, et al. Bootstrapping statistical parsers from small datasets. In: Proceedings of the 11th conference of the European association for computational linguistics, Budapest, Hungary, 2003. p. 331–8.

[23] Steedman Mark, Hwa Rebecca, Clark Stephen, Osborne Miles, Sarkar Anoop, Hockenmaier Julia, et al. Example selection for bootstrapping statistical parsers. In: Proceedings of the annual meeting of the North American association for computational linguistics (NAACL-HLT-03), Edmonton, Canada, 2003. p. 157–64.

[24] Blitzer John, McDonald Ryan, Pereira Fernando. Domain adaptation with structural correspondence learning. In: Empirical methods in natural language processing conference (EMNLP), Sydney, Australia, 2006. p. 120–8.

[25] Clark Stephen, Curran James R. Partial training for a lexicalized-grammar parser. In: Proceedings of the human language technology conference and the annual meeting of the North American chapter of the association for computational linguistics (HLT-NAACL'06), New York, 2006. p. 144–51.

[26] Hara Tadayoshi, Miyao Yusuke, Tsujii Jun'ichi. Evaluating impact of re-training a lexical disambiguation model on domain adaptation of an HPSG parser. In: Proceedings of IWPT, Prague, Czech Republic, 2007. p. 11–22.

[27] Ninomiya Takashi, Matsuzaki Takuya, Tsuruoka Yoshimasa, Miyao Yusuke, Tsujii Jun'ichi. Extremely lexicalized models for accurate and fast HPSG parsing. In: Proceedings of the EMNLP conference, 2006.

[28] Miyao Yusuke, Tsujii Jun'ichi. Probabilistic disambiguation models for wide-coverage HPSG parsing. In: Proceedings of the 43rd meeting of the ACL, University of Michigan, Ann Arbor, 2005. p. 83–90.

[29] Pollard Carl, Sag Ivan. Head-driven phrase structure grammar. Chicago: The University of Chicago Press; 1994.

[30] Schabes Yves, Abeillé Anne, Joshi Aravind. Parsing strategies with 'lexicalised' grammars: application to tree adjoining grammar. In: Proceedings of the 12th COLING conference, Budapest, Hungary, 1988.

[31] Kaplan Ronald M, Bresnan Joan. Lexical–functional grammar: a formal system for grammatical representation. In: Bresnan Joan, editor. The mental representation of grammatical relations. Cambridge, MA: The MIT Press; 1982. p. 173–281. Reprinted in Dalrymple, Mary, Kaplan, Ronald M, Maxwell, John, Zaenen, Annie, editors. Formal issues in lexical–functional grammar. Stanford: Center for the Study of Language and Information; 1995. p. 29–130.

[32] Clark Stephen, Hockenmaier Julia, Steedman Mark. Building deep dependency structures with a wide-coverage CCG parser. In: Proceedings of the 40th meeting of the ACL, Philadelphia, PA, 2002. p. 327–34.

[33] Hockenmaier Julia, Steedman Mark. Generative models for statistical parsing with Combinatory Categorial Grammar. In: Proceedings of the 40th meeting of the ACL, Philadelphia, PA, 2002. p. 335–42.

[34] Hockenmaier Julia. Parsing with generative models of predicate-argument structure. In: Proceedings of the 41st meeting of the ACL, Sapporo, Japan, 2003. p. 359–66.

[35] Clark Stephen, Curran James R. Parsing the WSJ using CCG and log-linear models. In: Proceedings of the 42nd meeting of the ACL, Barcelona, Spain, 2004. p. 104–11.

[36] Wood Mary McGee. Categorial grammars. London: Routledge; 1993.

[37] Bar-Hillel Yehoshua. A quasi-arithmetical notation for syntactic description. Language 1953;29:47–58.

[38] Curry Haskell B, Feys Robert. Combinatory logic, vol. I. Amsterdam: North Holland; 1958.

[39] Steedman Mark. Surface structure and interpretation. Cambridge, MA: The MIT Press; 1996.

[40] Hockenmaier Julia, Steedman Mark. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. Comput Linguist 2007;33(3):355–96.

[41] Ratnaparkhi Adwait. A maximum entropy model for part-of-speech tagging. In: Proceedings of the EMNLP conference, Philadelphia, PA, 1996. p. 133–42.

[42] Curran James R, Clark Stephen. Investigating GIS and smoothing for maximum entropy taggers. In: Proceedings of the 10th meeting of the EACL, Budapest, Hungary, 2003. p. 91–8.

[43] Clark Stephen. A supertagger for Combinatory Categorial Grammar. In: Proceedings of the TAG+ workshop, Venice, Italy, 2002. p. 19–24.

[44] Kasami J. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, MA, 1965.

[45] Younger D. Recognition and parsing of context-free languages in time n^3. Inf Control 1967;10(2):189–208.

[46] Ratnaparkhi Adwait. Maximum entropy models for natural language ambiguity resolution. Ph.D. Thesis, University of Pennsylvania; 1998.

[47] Lafferty John, McCallum Andrew, Pereira Fernando. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th international conference on machine learning, Williams College, MA, 2001. p. 282–9.

[48] King Tracy H, Crouch Richard, Riezler Stefan, Dalrymple Mary, Kaplan Ronald M. The PARC 700 dependency bank. In: Proceedings of the 4th international workshop on linguistically interpreted corpora, Budapest, Hungary, 2003.

[49] Briscoe Ted, Carroll John, Watson Rebecca. The second release of the RASP system. In: Proceedings of the interactive demo session of the joint conference of the international committee on computational linguistics and the association for computational linguistics (COLING/ACL-06), Sydney, Australia, 2006.

[50] Clark Stephen, Curran James R. Formalism-independent parser evaluation with CCG and DepBank. In: Proceedings of the 45th meeting of the ACL, Prague, Czech Republic, 2007. p. 248–55.

[51] Grover Claire, Matthews Michael, Tobin Richard. Tools to address the interdependence between tokenisation and standoff annotation. In: NLPXML, 2006.
[52] Grover Claire, Lascarides Alex, Lapata Mirella. A comparison of parsing technologies for the biomedical domain. Nat Lang Eng 2005;11:27–65.
[53] Tsuruoka Yoshimasa, Tateisi Yuka, Kim Jin-Dong, Ohta Tomoko, McNaught John, Ananiadou Sophia, et al. Developing a robust part-of-speech tagger for biomedical text. In: Advances in informatics – 10th Panhellenic conference on informatics, LNCS 3746, Volos, Greece, 2005. p. 382–92.
[54] Pyysalo Sampo, Ginter Filip, Heimonen Juho, Björne Jari, Boberg Jorma, Järvinen Jouni, et al. BioInfer: a corpus for information extraction in the biomedical domain. BMC Bioinform 2007;8:50.
[55] Pyysalo Sampo, Ginter Filip, Laippala Veronika, Haverinen Katri, Heimonen Juho, Salakoski Tapio. On the unification of syntactic annotations under the stanford dependency scheme: a case study on BioInfer and GENIA. In: ACL'07 workshop on biological, translational, and clinical language processing, Prague, Czech Republic, 2007. p. 25–32.
[56] de Marneffe Marie-Catherine, MacCartney Bill, Manning Christopher D. Generating typed dependency parses from phrase structure parses. In: Proceedings of the 5th LREC conference, Genoa, Italy, 2006. p. 449–54.
[57] Lin Dekang. A dependency-based method for evaluating broad-coverage parsers. In: Proceedings of IJCAI-95, Montreal, Canada, 1995. p. 1420–5.
[58] Carroll John, Briscoe Ted, Sanfilippo Antonio. Parser evaluation: a survey and a new proposal. In: Proceedings of the 1st LREC conference, Granada, Spain, 1998. p. 447–54.
[59] Clegg Andrew B, Shepherd Adrian J. Benchmarking natural-language parsers for biological applications using dependency graphs. BMC Bioinform 2007;8:24.
[60] Briscoe Ted, Carroll John. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In: Proceedings of the poster session of the joint conference of the international committee on computational linguistics and the association for computational linguistics (COLING/ACL-06), Sydney, Australia, 2006.
[61] Black E, Abney S, Flickinger D, Gdaniec C, Grishman R, Harrison P, et al. A procedure for quantitatively comparing the syntactic coverage of English grammars. In: Proceedings of the DARPA speech and natural language workshop, 1991. p. 306311.
[62] Kulick Seth, Bies Ann, Liberman Mark, Mandel Mark, McDonald Ryan, Palmer Martha, et al. Integrated annotation for biomedical information extraction. In: HLT/NAACL, 2004. p. 61–8.
[63] Sleator Daniel D, Temperley Davy. Parsing English with a link grammar. In: Third international workshop on parsing technologies, 1993. p. 277–91.
[64] Pyysalo Sampo, Salakoski Tapio, Aubin Sophie, Nazarenko Adeline. Lexical adaptation of link grammar to the biomedical sublanguage: a comparative evaluation of three approaches. BMC Bioinform 2006;7(Suppl. 3):S2.