

# Syntactic Scope Resolution in Uncertainty Analysis

Lilja Øvrelid<sup>✦✦</sup> and Erik Velldal<sup>✦</sup> and Stephan Oepen<sup>✦</sup>

<sup>✦</sup> University of Oslo, Department of Informatics

<sup>✦</sup> Universität Potsdam, Institut für Linguistik

ovrelid@uni-potsdam.de and erikve@ifi.uio.no and oe@ifi.uio.no

## Abstract

We show how the use of syntactic structure enables the resolution of hedge scope in a hybrid, two-stage approach to uncertainty analysis. In the first stage, a Maximum Entropy classifier, combining surface-oriented and syntactic features, identifies cue words. With a small set of hand-crafted rules operating over dependency representations in stage two, we attain the best overall result (in terms of both combined ranks and average  $F_1$ ) in the 2010 CoNLL Shared Task.

## 1 Background—Motivation

Recent years have witnessed an increased interest in the analysis of various aspects of sentiment in natural language (Pang & Lee, 2008). The subtask of *hedge resolution* deals with the analysis of uncertainty as expressed in natural language, and the linguistic means (so-called hedges) by which speculation or uncertainty are expressed. Information of this kind is of importance for various mining tasks which aim at extracting factual data. Example (1), taken from the BioScope corpus (Vincze, Szarvas, Farkas, Móra, & Csirik, 2008), shows a sentence where uncertainty is signaled by the modal verb *may*.<sup>1</sup>

- (1) {The unknown amino acid ⟨may⟩ be used by these species}.

The topic of the Shared Task at the 2010 Conference for Natural Language Learning (CoNLL) is hedge detection in biomedical literature—in a sense ‘zooming in’ on one particular aspect of the broader BioNLP Shared Task in 2009 (Kim, Ohta, Pyysalo, Kano, & Tsujii, 2009). It involves two subtasks: Task 1 is described as *learning to detect*

<sup>1</sup>In examples throughout this paper, angle brackets highlight hedge cues, and curly braces indicate the scope of a given cue, as annotated in BioScope.

*sentences containing uncertainty*; the objective of Task 2 is *learning to resolve the in-sentence scope of hedge cues* (Farkas, Vincze, Mora, Csirik, & Szarvas, 2010). The organizers further suggest: *This task falls within the scope of semantic analysis of sentences exploiting syntactic patterns [...]*.

The utility of syntactic information within various approaches to sentiment analysis in natural language has been an issue of some debate (Wilson, Wiebe, & Hwa, 2006; Ng, Dasgupta, & Arifin, 2006), and the potential contribution of syntax clearly varies with the specifics of the task. Previous work in the hedging realm has largely been concerned with cue detection, i.e. identifying uncertainty cues such as *may* in (1), which are predominantly individual tokens (Medlock & Briscoe, 2007; Kilicoglu & Bergler, 2008). There has been little previous work aimed at actually resolving the scope of such hedge cues, which presumably constitutes a somewhat different and likely more difficult problem. Morante and Daelemans (2009) present a machine-learning approach to this task, using token-level, lexical information only. To this end, CoNLL 2010 enters largely uncharted territory, and it remains to be seen (a) whether syntactic analysis indeed is a necessary component in approaching this task and, more generally, (b) to what degree the specific task setup can inform us about the strong and weak points in current approaches and technology.

In this article, we investigate the contribution of syntax to hedge resolution, by reflecting on our experience in the CoNLL 2010 task.<sup>2</sup> Our CoNLL system submission ranked fourth (of 24) on Task 1 and third (of 15) on Task 2, for an overall best average result (there appears to be very limited overlap among top performers for the two subtasks).

<sup>2</sup>It turns out, in fact, that all the top-performing systems in Task 2 of the CoNLL Shared Task rely on syntactic information provided by parsers, either in features for machine learning or as input to manually crafted rules (Morante, Asch, & Daelemans, 2010; Rei & Briscoe, 2010).

	Sentences	Hedged Sentences	Cues	Multi-Word Cues	Tokens	Cue Tokens
<b>Abstracts</b>	11871	2101	2659	364	309634	3056
<b>Articles</b>	2670	519	668	84	68579	782
<b>Total</b>	14541	2620	3327	448	378213	3838

Table 1: Summary statistics for the Shared Task training data.

This article transcends our CONLL system description (Velldal, Øvrelid, & Oepen, 2010) in several respects, presenting updated and improved cue detection results (§ 3 and § 4), focusing on the role of syntactic information rather than on machine learning specifics (§ 5 and § 6), providing an analysis and discussion of Task 2 errors (§ 7), and generally aiming to gauge the value of available annotated data and processing tools (§ 8). We present a hybrid, two-level approach for hedge resolution, where a statistical classifier detects cue words, and a small set of manually crafted rules operating over syntactic structures resolve scope. We show how syntactic information—produced by a data-driven dependency parser complemented with information from a ‘deep’, hand-crafted grammar—contributes to the resolution of in-sentence scope of hedge cues, discussing various types of syntactic constructions and associated scope detection rules in considerable detail. We furthermore present a manual error analysis, which reveals remaining challenges in our scope resolution rules as well as several relevant idiosyncrasies of the preexisting BioScope annotation.

## 2 Task, Data, and System Basics

### Task Definition and Evaluation Metrics

Task 1 is a binary sentence classification task: identifying utterances as being *certain* or *uncertain*. Following common practice, this subtask is evaluated in terms of precision, recall, and  $F_1$  for the ‘positive’ class, i.e. *uncertain*. In our work, we approach Task 1 as a byproduct of the full hedge resolution problem, labeling a sentence as *uncertain* if it contains at least one token classified as a hedge cue. In addition to the sentence-level evaluation for Task 1, we also present precision, recall, and  $F_1$  for the cue-level.

Task 2 comprises two subtasks: cue detection and scope resolution. The official CONLL eval-

uation does not tease apart these two aspects of the problem, however: Only an exact match of both the cue and scope bracketing (in terms of substring positions) will be counted as a success, again quantified in terms of precision, recall, and  $F_1$ . Discussing our results below, we report cue detection and scope resolution performance separately, and further put scope results into perspective against an upper bound based on the gold-standard cue annotation.

Besides the primary biomedical domain data, some annotated Wikipedia data was provided for Task 1, and participating systems are classified as *in-domain* (using exclusively the domain-specific data), *cross-domain* (combining both types of training data), or *open* (utilizing additional uncertainty-related resources). In our work, we focus on the interplay of syntax and the more challenging Task 2; we ignored the Wikipedia track in Task 1. Despite our using general NLP tools (see below), our system falls into the most restrictive, *in-domain* category.

**Training and Evaluation Data** The training data for the CONLL 2010 Shared Task is taken from the BioScope corpus (Vincze et al., 2008) and consists of 14,541 ‘sentences’ (or other root-level utterances) from biomedical abstracts and articles (see Table 1).<sup>3</sup> The BioScope corpus provides annotation for hedge cues as well as their scope. According to the annotation guidelines (Vincze et al., 2008), the annotation adheres to a principle of minimalism when it comes to hedge cues, i.e. the minimal unit expressing hedging is annotated. The inverse is true of scope annotations, which adhere to a principle of maximal scope—meaning that scope should be set to the largest syntactic

<sup>3</sup>As it was known beforehand that evaluation would draw on full articles only, we put more emphasis on the article subset of the training data, for example in cross validation testing and manual diagnosis of errors.

ID	FORM	LEMMA	POS	FEATS	HEAD	DEPREL	XHEAD	XDEP
1	The	the	DT	—	4	NMOD	4	SPECDET
2	unknown	unknown	JJ	degree:attributive	4	NMOD	4	ADJUNCT
3	amino	amino	JJ	degree:attributive	4	NMOD	4	ADJUNCT
4	acid	acid	NN	pers:3 case:nom num:sg ntype:common	5	SBJ	3	SUBJ
5	may	may	MD	mood:ind subcat:MODAL tense:pres clauseType:decl	0	ROOT	0	ROOT
6	be	be	VB	—	5	VC	7	PHI
7	used	use	VBN	subcat:V-SUBJ-OBJ vtype:main passive:+	6	VC	5	XCOMP
8	by	by	IN	—	7	LGS	9	PHI
9	these	these	DT	deixis:proximal	10	NMOD	10	SPECDET
10	species	specie	NNS	num:pl pers:3 case:obl common:count ntype:common	8	PMOD	7	OBL-AG
11	.	.	.	—	5	P	0	PUNC

Table 2: Stacked dependency representation of example (1), with MaltParser and XLE annotations.

unit possible.

For evaluation purposes, the task organizers provided newly annotated biomedical articles, following the same general BioScope principles. The CoNLL 2010 evaluation data comprises 5,003 additional utterances (138,276 tokens), of which 790 are annotated as hedged. The data contains a total of 1033 cues, of which 87 are so-called multiword cues (i.e. cues spanning multiple tokens), comprising 1148 cue tokens altogether.

**Stacked Dependency Parsing** For syntactic analysis we employ the open-source MaltParser (Nivre, Hall, & Nilsson, 2006), a platform for data-driven dependency parsing. For improved accuracy and portability across domains and genres, we make our parser incorporate the predictions of a large-scale, general-purpose LFG parser—following the work of Øvrelid, Kuhn, and Spreyer (2009). A technique dubbed *parser stacking* enables the data-driven parser to learn, not only from gold standard treebank annotations, but from the output of another parser (Nivre & McDonald, 2008). This technique has been shown to provide significant improvements in accuracy for both English and German (Øvrelid et al., 2009), and a similar setup employing an HPSG grammar has been shown to increase domain independence in data-driven dependency parsing (Zhang & Wang, 2009). The stacked parser combines two quite different approaches—data-driven dependency parsing and ‘deep’ parsing with a hand-crafted grammar—and thus provides us with a broad range of different types of linguistic information for the hedge resolution task.

MaltParser is based on a deterministic parsing strategy in combination with treebank-induced classifiers for predicting parse transitions. It supports a rich feature representation of the parse his-

tory in order to guide parsing and may easily be extended to take additional features into account. The procedure to enable the data-driven parser to learn from the grammar-driven parser is quite simple. We parse a treebank with the XLE platform (Crouch et al., 2008) and the English grammar developed within the ParGram project (Butt, Dyvik, King, Masuichi, & Rohrer, 2002). We then convert the LFG output to dependency structures, so that we have two parallel versions of the treebank—one gold standard and one with LFG annotation. We extend the gold standard treebank with additional information from the corresponding LFG analysis and train MaltParser on the enhanced data set.

Table 2 shows the enhanced dependency representation of example (1) above, taken from the training data. For each token, the parsed data contains information on the word form, lemma, and part of speech (PoS), as well as on the head and dependency relation in columns 6 and 7. The added XLE information resides in the FEATS column, and in the XLE-specific head and dependency columns 8 and 9. Parser outputs, which in turn form the basis for our scope resolution rules discussed in Section 5, also take this same form. The parser employed in this work is trained on the Wall Street Journal sections 2–24 of the Penn Treebank (PTB), converted to dependency format (Johansson & Nugues, 2007) and extended with XLE features, as described above. Parsing uses the arc-eager mode of MaltParser and an SVM with a polynomial kernel. When tested using 10-fold cross validation on the enhanced PTB, the parser achieves a labeled accuracy score of 89.8.

**PoS Tagging and Domain Variation** Our parser is trained on financial news, and although stacking with a general-purpose LFG parser is ex-

pected to aid domain portability, substantial differences in domain and genre are bound to negatively affect syntactic analysis (Gildea, 2001). MaltParser presupposes that inputs have been PoS tagged, leaving room for variation in preprocessing. On the one hand, we aim to make parser inputs maximally similar to its training data (i.e. the conventions established in the PTB); on the other hand we wish to benefit from specialized resources for the biomedical domain.

The GENIA tagger (Tsuruoka et al., 2005) is particularly relevant in this respect (as could be the GENIA Treebank proper<sup>4</sup>). However, we found that GENIA tokenization does not match the PTB conventions in about one out of five sentences (for example wrongly splitting tokens like ‘390,926’ or ‘Ca(2+)’); also in tagging proper nouns, GENIA systematically deviates from the PTB. Hence, we adapted an in-house tokenizer (using cascaded finite-state rules) to the CoNLL task, run two PoS taggers in parallel, and eclectically combine annotations across the various preprocessing components—predominantly giving precedence to GENIA lemmatization and PoS hypotheses.

To assess the impact of improved, domain-adapted inputs on our hedge resolution system, we contrast two configurations: first, running the parser in the exact same manner as Øvrelid, Kuhn, and Spreyer (2010), we use TreeTagger (Schmid, 1994) and its standard model for English (trained on the PTB) for preprocessing; second, we give as inputs to the parser our refined tokenization and merged PoS tags, as described above. When evaluating the two modes of preprocessing on the articles subset of the training data, and using gold-standard cues, our system for resolving cue scopes (presented in §5) achieves an  $F_1$  of 66.31 with TreeTagger inputs, and 72.30 using our refined tokenization and tagger combination. These results underline the importance of domain adaptation for accurate syntactic analysis, and in the following we assume our hybrid in-house setup.

<sup>4</sup>Although the GENIA Treebank provides syntactic annotation in a form inspired by the PTB, it does not provide function labels. Therefore, our procedure for converting from constituency to dependency requires non-trivial adaptation before we can investigate the effects of retraining the parser against GENIA.

### 3 Stage 1: Identifying Hedge Cues

For the task of identifying hedge cues, we developed a binary maximum entropy (MaxEnt) classifier. The identification of cue words is used for (a) classifying sentences as certain/uncertain (Task 1), and (b) providing input to the syntactic rules that we later apply for resolving the in-sentence scope of the cues (Task 2). We also report evaluation scores for the sub-task of cue detection in isolation.

As annotated in the training data, it is possible for a hedge cue to span multiple tokens, e.g. as in *whether or not*. The majority of the multi-word cues in the training data are very infrequent, however, most occurring only once, and the classifier itself is not sensitive to the notion of multi-word cues. Instead, the task of determining whether a cue word forms part of a larger multi-word cue, is performed in a separate post-processing step (applying a heuristic rule targeted at only the most frequently occurring patterns of multi-word cues in the training data).

During development, we trained cue classifiers using a wide variety of feature types, both syntactic and surface-oriented. In the end, however, we found  $n$ -gram-based lexical features to have the greatest contribution to classifier performance. Our best-performing classifier so far (see ‘Final’ in Table 3) includes the following feature types:  $n$ -grams over forms (up to 2 tokens to the right),  $n$ -grams over base forms (up to 3 tokens left and right), PoS (from GENIA), subcategorization frames (from XLE), and phrase-structural coordination level (from XLE). Our CoNLL system description includes more details of the various other feature types that we experimented with (Vellidal et al., 2010).

### 4 Cue Detection Evaluation

Table 3 summarizes the performance of our MaxEnt hedge cue classifier in terms of precision, recall and  $F_1$ , computed using the official Shared Task scorer script. The sentence-level scores correspond to Task 1 of the Shared Task, and the cue-level scores are based on the exact-match counts for full hedge cues (possibly spanning multiple tokens).

Configuration	Sentence Level			Cue Level		
	Prec	Rec	F <sub>1</sub>	Prec	Rec	F <sub>1</sub>
Baseline, Development	79.25	79.45	79.20	77.37	71.70	74.43
Final, Development	91.39	86.78	89.00	90.18	79.47	84.49
Final, Held-Out	85.61	85.06	85.33	81.97	76.41	79.10

Table 3: Isolated evaluation of the hedge cue classifier.

As the CoNLL test data was known beforehand to consist of articles only, in 10-fold cross validation for classifier development we tested exclusively against the articles segment, while always including all sentences from the abstracts in the training set. This corresponds to the development results in Table 3, while the held-out results are for the official Shared Task evaluation data (training on all the available training data). A model using only unigram features serves as a baseline.

## 5 Stage 2: Resolving Scope

Hedge scope may vary quite a lot depending on linguistic properties of the cue in question. In our approach to scope resolution we rely heavily on syntactic information, taken from the dependency structures proposed by both MaltParser and XLE, as well as on various additional features relating to specific syntactic constructions.

We constructed a small set of heuristic rules which define the scope for each cue detected in Stage 1. In developing these rules, we made use of the information provided by the guidelines for scope annotation in the BioScope corpus (Vincze et al., 2008), combined with manual inspection of the training data in order to further generalize over the phenomena discussed by Vincze et al. (2008) and work out interactions of constructions for various types of cues.

The rules take as input a parsed sentence which has been further tagged with hedge cues. They operate over the dependency structures and additional features provided by the parser. Default scope is set to start at the cue word and span to the end of the sentence (modulo punctuation), and this scope also provides the baseline for the evaluation of our rules. In the following, we discuss broad classes of rules, organized by categories of hedge cues. As there is no explicit representation of phrase or clause boundaries in our depen-

ency universe, we assume a set of functions over dependency graphs, for example finding the left- or rightmost (direct) *dependent* of a given node, or transitively selecting left- or rightmost *descendants*.

**Coordination** The dependency analysis of coordination provided by our parser makes the first conjunct the head of the coordination. For cues that are coordinating conjunctions (PoS tag CC), such as *or*, we define the scope as spanning the whole coordinate structure, i.e. start scope is set to the leftmost dependent of the head of the coordination, e.g., *roX* in (2), and end scope is set to its rightmost dependent (conjunct), e.g., *RNAs* in (2). This analysis provides us with coordinations at various syntactic levels, such as NP and  $\bar{N}$  (2), AP and AdvP, or VP (3):

- (2) [...] the {roX genes ⟨or⟩ RNAs} recruit the entire set of MSL proteins [...]
- (3) [...] the binding interfaces are more often {kept ⟨or⟩ even reused} rather than lost in the course of evolution.

**Adjectives** We distinguish between adjectives (JJ) in *attributive* (NMOD) function and adjectives in *predicative* (PRD) function. Attributive adjectives take scope over their (nominal) head, with all its dependents, as in (4) and (5):

- (4) The {(possible) selenocysteine residues} are shown in red, [...]
- (5) Extensive analysis of the flanks failed to show any hallmarks of {(putative) transposons that might be associated with this RAG1-like protein}, [...]

For adjectives in a predicative function the scope includes the subject argument of the head verb (the copula), as well as a (possible) clausal argument, as in (6). The scope does not, however, include expletive subjects, as in (7).

- (6) Therefore, {the unknown amino acid, if it is encoded by a stop codon, is ⟨unlikely⟩ to exist in the current databases of microbial genomes}.
- (7) For example, it is quite {⟨likely⟩ that there exists an extremely long sequence that is entirely unique to U}.

**Verbs** The scope of verbal cues is a bit more complex and depends on several factors. In our rules, we distinguish *passive* usages from active usages, *raising* verbs from non-raising verbs, and the presence or absence of a subject-control embedding context. The scopes of both passive and raising verbs include the subject argument of their head verb, as in (8) and (9), unless it is an expletive pronoun, as in (10).

- (8) {Interactions determined by high-throughput methods are generally ⟨considered⟩ to be less reliable than those obtained by low-throughput studies} 1314 and as a consequence [...]
- (9) {Genomes of plants and vertebrates ⟨seem⟩ to be free of any recognizable Transib transposons} (Figure 1).
- (10) It has been {⟨suggested⟩ that unstructured regions of proteins are often involved in binding interactions, particularly in the case of transient interactions} 77.

In the case of subject control involving a hedge cue, specifically modals, subject arguments are included in scopes where the controller heads a passive construction or a raising verb, as in example (1) above, repeated here for convenience:

- (11) {The unknown amino acid ⟨may⟩ be used by these species}.

In general, the end scope of verbs should extend over the minimal clause that contains the verb in question. In terms of dependency structures, we define the clause boundary as comprising the chain of descendants of a verb which is not intervened by a token with a higher attachment in the graph than the verb in question. In example (8) for instance, the sentence-level conjunction *and* marks the end of the clause following the cue *considered*.

**Prepositions and Adverbs** Cues that are tagged as prepositions (including some complementizers) take scope over their argument, with all its descendants, (12). Adverbs take scope over their head with all its (non-subject) syntactic descendants (13).

	Configuration	F <sub>1</sub>
BSP	Default, Gold Cues	45.21
	Rules, Gold Cues	72.31
	Rules, System Cues	64.77
BSE	Rules, Gold Cues	66.73
	Rules, System Cues	55.75

Table 4: Evaluation of scope resolution rules.

- (12) {⟨Whether⟩ the codon aligned to the inframe stop codon is a nonsense codon or not} was neglected at this stage.
- (13) These effects are {⟨probably⟩ mediated through the 1,25(OH)2D3 receptor}.

**Multi-Word Cues** In the case of multi-word cues, such as *indicate that* or *either ... or*, we need to determine the head of the multi-word unit. We then set the scope of the whole unit to the scope of the head token.

As an illustration of rule processing, consider our running example (11), with its syntactic analysis as shown in Table 2 above. This example invokes a variety of syntactic properties, including parts of speech, argumenthood, voice etc. Initially, the scope of the hedge cue is set to default scope. Then the subject control rule is applied, which checks the properties of the verbal argument *used*, going through a chain of verbal dependents from the modal verb. Since it is marked as passive in the LFG analysis, the start scope is set to include the subject of the cue word (the leftmost descendant in its *SUBJ* dependent).

## 6 Rule Evaluation

Table 4 summarizes scope resolution performance (viewed as a an isolated subtask) for various configurations, both against the articles section of the CONLL training data (dubbed BSP) and against the held-out evaluation data (BSE). First of all, we note that the ‘default scope’ baseline is quite strong: unconditionally extending the scope of a cue to the end of the sentence yields an F<sub>1</sub> of 45.21. Given gold standard cue information, our scope rules improve on the baseline by 27 points on the articles section of the data set, for an F<sub>1</sub> of 72.31; with system-assigned hedge cues, our rules still

achieve an  $F_1$  of 64.77. Note that scope resolution scores based on classified cues also yield the end-to-end system evaluation for Task 2.

The bottom rows of Table 4 show the evaluation of scope rules on the CoNLL held-out test data. Using system cues, scope resolution on the held-out data scores at 55.75  $F_1$ . Comparing to the result on the (articles portion of the) training data, we observe a substantial drop in performance (of six points with gold-standard cues, nine points with system cues). There are several possible explanations for this effect. First of all, there may well be a certain degree of overfitting of our rules to the training data. The held-out data may contain hedging constructions that are not covered by our current set of scope rules, or annotation of parallel constructions may in some cases differ in subtle ways (see § 7 below). Moreover, scope resolution performance is of course influenced by cue detection (see Table 3). The cue-level  $F_1$  of our system on the held-out data set is 79.10, compared to 84.49 (using cross validation) on the training data. This drop in cue-level performance appears to affect classification precision far more than recall. Of course, given that our heuristics for identifying multi-word cues were based on patterns extracted from the training data, some loss in the cue-level score was expected.

## 7 Error Analysis

To start shedding some light on the significance of our results, we performed a manual error analysis on the article portion of the training material (BSP), with two of the authors (trained linguists) working in tandem. Using gold-standard cues, our scope resolution rules fail to exactly replicate the target annotation in 185 (of 668) cases, corresponding to 72.31  $F_1$  in Table 4 above. Our evaluators reviewed and discussed these 185 cases, classifying 156 (84%) as genuine system errors, 22 (12%) as likely<sup>5</sup> annotation errors, and a re-

<sup>5</sup>In some cases, there is no doubt that annotation is erroneous, i.e. in violation of the available annotation guidelines (Vincze et al., 2008) or in conflict with otherwise unambiguous patterns. In other cases, however, judgments are necessarily based on generalizations made by the evaluators, i.e. assumptions about the underlying system and syntactic analyses implicit in the BioScope annotations. Furthermore, selecting items for manual analysis that do not align with the

maintaining seven cases as involving controversial or seemingly arbitrary decisions.

The two most frequent classes of system errors pertain (a) to the recognition of phrase and clause boundaries and (b) to not dealing successfully with relatively superficial properties of the text. Examples (14) and (15) illustrate the first class of errors, where in addition to the gold-standard annotation we use vertical bars (‘|’) to indicate scope predictions of our system.

- (14) [...] {the reverse complement |mR of m will be  
<considered> to be [...]}  
(15) This |{(might) affect the results} if there is a  
systematic bias on the composition of a protein  
interaction set|.

In our syntax-driven approach to scope resolution, system errors will almost always correspond to a failure in determining constituent boundaries, in a very general sense. However, specifically example (15) is indicative of a key challenge in this task, where adverbials of condition, reason, or contrast frequently attach within the dependency domain of a hedge cue, yet are rarely included in the scope annotation.

Example (16) demonstrates our second frequent class of system errors. One in six items in the BSP training data contains a sentence-final parenthesized element or trailing number, as for example (2), (9), or (10) above; most of these are bibliographic or other in-text references, which are never included in scope annotation. Hence, our system includes a rule to ‘back out’ from trailing parentheticals; in examples like (16), however, syntax does not make explicit the contrast between an in-text reference vs. another type of parenthetical.

- (16) More specifically, {|the bristle and leg phenotypes are  
<likely> to result from reduced signaling by D| (and  
not by Ser)}.

Moving on to apparent annotation errors, the rules for inclusion (or not) of the subject in the scope of verbal hedge cues and decisions on boundaries (or internal structure) of nominals

predictions made by our scope resolution rules is likely to bias our sample, such that our estimated proportion of 12% annotation errors cannot be used to project an overall error rate.

seem problematic—as illustrated in examples (17) to (22).<sup>6</sup>

- (17) [...] and |this is also {⟨thought⟩ to be true for the full protein interaction networks we are modeling}|.
- (18) [...] {Neur |⟨can⟩ promote Ser signaling}|.
- (19) |Some of the domain pairs {⟨seem⟩ to mediate a large number of protein interactions, thus acting as reusable connectors}|.
- (20) One {⟨possible⟩ explanation| is functional redundancy with the mouse Neur2 gene}.
- (21) [...] |redefinition of {one of them is ⟨feasible⟩}|.
- (22) |The {Bcl-2 family ⟨appears⟩ to function [...]}|.

Finally, the difficult corner cases invoke non-constituent coordination, ellipsis, or NP-initial focus adverbs—and of course interactions of the phenomena discussed above. Without making the syntactic structures assumed explicit, it is often very difficult to judge such items.

## 8 Reflections — Outlook

Our combination of stacked dependency parsing and hand-crafted scope resolution rules proved adequate for the CoNLL 2010 competition, confirming the central role of syntax in this task. With a comparatively small set of rules (implemented in a few hundred lines of code), constructed through roughly two full weeks of effort (studying BioScope annotations and developing rules), our CoNLL system achieved an end-to-end  $F_1$  of 55.33 on Task 2.<sup>7</sup> The two submissions with better results (at 57.32 and 55.65  $F_1$ ) represent groups who have pioneered the hedge analysis task in previous years (Morante et al., 2010; Rei & Briscoe, 2010). Scores for other ‘in-domain’ participants range from 52.24 to 2.15  $F_1$ .

<sup>6</sup>Like in the presentation of system errors, we include scope predictions of our own rules here too, which we believe to be correct in these cases. Also in this class of errors, we find the occasional ‘uninteresting’ mismatch, for example related to punctuation marks and inconsistencies around parentheses.

<sup>7</sup>In § 4 and § 6 above, we report scores for a slightly improved version of our system, where (after the official CoNLL submission date) we eliminated a bug related to the treatment of sentence-initial whitespace in the XML annotations. At an end-to-end  $F_1$  of 55.75, this system would outrank the second best performer in Task 2.

Doubtless there is room for straightforward extension: for example retraining our parser on the GENIA Treebank, further improving the cue classifier, and refining scope resolution rules in the light of the error analysis above.

At the same time, we remain mildly ambivalent about the long-term impact of some of the specifics of the 2010 CoNLL task. Shared tasks (i.e. system bake-offs) have become increasingly popular in past years, and in some sub-fields (e.g. IE, SMT, or dependency parsing) high-visibility competitions can shape community research agendas. Hence, even at this early stage, it seems appropriate to reflect on the possible conclusions to be drawn from the 2010 hedge resolution task. First, we believe the harsh ‘exact substring match’ evaluation metric underestimates the degree to which current technology can solve this problem; furthermore, idiosyncratic, string-level properties (e.g. the exact treatment of punctuation or parentheses) may partly obscure the interpretation of methods used and corresponding system performance.

These effects are compounded by some concerns about the quality of available annotation. Even though we tried fine-tuning our cross validation testing to the nature of the evaluation data (comprising only articles), our system performs substantially worse on the newly annotated CoNLL test data, in both stages.<sup>8</sup> In our view, the annotation of hedge cues and scopes ideally would be overtly related to at least some level of syntactic annotation—as would in principle be possible for the segment of BioScope drawing on the abstracts of the GENIA Treebank.

## Acknowledgements

We are grateful to the organizers of the 2010 CoNLL Shared Task and creators of the BioScope resource; first, for engaging in these kinds of community service, and second for many in-depth discussions of annotation and task details. We thank our colleagues at the Universities of Oslo and Potsdam for their comments and support.

<sup>8</sup>We are leaving open the possibility to further refine our system; we have therefore abstained from an error analysis on the evaluation data so far.

## References

- Butt, M., Dyvik, H., King, T. H., Masuichi, H., & Rohrer, C. (2002). The Parallel Grammar Project. In *Proceedings of COLING workshop on grammar engineering and evaluation* (pp. 1–7). Taipei, Taiwan.
- Crouch, D., Dalrymple, M., Kaplan, R., King, T., Maxwell, J., & Newman, P. (2008). *XLE documentation*. Palo Alto, CA. (Palo Alto Research Center)
- Farkas, R., Vincze, V., Mora, G., Csirik, J., & Szarvas, G. (2010). The CoNLL 2010 Shared Task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the 14th Conference on Natural Language Learning*. Uppsala, Sweden.
- Gildea, D. (2001). Corpus variation and parser performance. In *Proceedings of the 2001 conference on Empirical Methods in Natural Language Processing* (pp. 167–202). Pittsburgh, PA.
- Johansson, R., & Nguens, P. (2007). Extended constituent-to-dependency conversion for English. In J. Nivre, H.-J. Kaalep, & M. Koit (Eds.), *Proceedings of NODALIDA 2007* (p. 105-112). Tartu, Estonia.
- Kilicoglu, H., & Bergler, S. (2008). Recognizing speculative language in biomedical research articles: A linguistically motivated perspective. In *Proceedings of the BioNLP 2008 Workshop*. Columbus, OH, USA.
- Kim, J.-D., Ohta, T., Pyysalo, S., Kano, Y., & Tsujii, J. (2009). Overview of BioNLP 2009 Shared Task on event extraction. In *Proceedings of the BioNLP 2009 workshop companion volume for shared task* (pp. 1–9). Boulder, CO: Association for Computational Linguistics.
- Medlock, B., & Briscoe, T. (2007). Weakly supervised learning for hedge classification in scientific literature. In *Proceedings of the 45th Meeting of the Association for Computational Linguistics* (pp. 992–999). Prague, Czech Republic: Association for Computational Linguistics.
- Morante, R., Asch, V. V., & Daelemans, W. (2010). Memory-based resolution of in-sentence scope of hedge cues. In *Proceedings of the 14th Conference on Natural Language Learning* (pp. 40–47). Uppsala, Sweden.
- Morante, R., & Daelemans, W. (2009). Learning the scope of hedge cues in biomedical texts. In *Proceedings of the BioNLP 2009 Workshop* (pp. 28–36). Boulder, Colorado.
- Ng, V., Dasgupta, S., & Arifin, S. M. N. (2006). Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. Sydney, Australia.
- Nivre, J., Hall, J., & Nilsson, J. (2006). MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation* (p. 2216-2219). Genoa, Italy.
- Nivre, J., & McDonald, R. (2008, June). Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Meeting of the Association for Computational Linguistics* (pp. 950–958). Columbus, Ohio.
- Øvrelid, L., Kuhn, J., & Spreyer, K. (2009). Improving data-driven dependency parsing using large-scale LFG grammars. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics* (pp. 37–40). Singapore.
- Øvrelid, L., Kuhn, J., & Spreyer, K. (2010). Cross-framework parser stacking for data-driven dependency parsing. *TAL 2010 special issue on Machine Learning for NLP*, 50(3).
- Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2).
- Rei, M., & Briscoe, T. (2010). Combining manual rules and supervised learning for hedge cue and scope detection. In *Proceedings of the 14th Conference on Natural Language Learning* (pp. 56–63). Uppsala, Sweden.
- Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *International conference on new methods in language processing* (p. 44-49). Manchester, England.
- Tsuruoka, Y., Tateishi, Y., Kim, J.-D., Ohta, T., McNaught, J., Ananiadou, S., et al. (2005). Developing a robust Part-of-Speech tagger for biomedical text. In *Advances in informatics* (pp. 382–392). Berlin, Germany: Springer.
- Velldal, E., Øvrelid, L., & Oepen, S. (2010). Resolving speculation: MaxEnt cue classification and dependency-based scope rules. In *Proceedings of the 14th Conference on Natural Language Learning*. Uppsala, Sweden.
- Vincze, V., Szarvas, G., Farkas, R., Móra, G., & Csirik, J. (2008). The BioScope corpus: Annotation for negation, uncertainty and their scope in biomedical texts. In *Proceedings of the BioNLP 2008 Workshop*. Columbus, OH, USA.
- Wilson, T., Wiebe, J., & Hwa, R. (2006). Recognizing strong and weak opinion clauses. *Computational Intelligence*, 22(2), 73–99.
- Zhang, Y., & Wang, R. (2009). Cross-domain dependency parsing using a deep linguistic grammar. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics*. Singapore.