Speculation and Negation: Rules, Rankers, and the Role of Syntax

Erik Velldal*
University of Oslo

Lilja Øvrelid*
University of Oslo

Jonathon Read*
University of Oslo

Stephan Oepen*
University of Oslo

This article explores a combination of deep and shallow approaches to the problem of resolving the scope of speculation and negation within a sentence, specifically in the domain of biomedical research literature. The first part of the article focuses on speculation. After first showing how speculation cues can be accurately identified using a very simple classifier informed only by local lexical context, we go on to explore two different syntactic approaches to resolving the in-sentence scopes of these cues. Whereas one uses manually crafted rules operating over dependency structures, the other automatically learns a discriminative ranking function over nodes in constituent trees. We provide an in-depth error analysis and discussion of various linguistic properties characterizing the problem, and show that although both approaches perform well in isolation, even better results can be obtained by combining them, yielding the best published results to date on the CoNLL-2010 Shared Task data. The last part of the article describes how our speculation system is ported to also resolve the scope of negation. With only modest modifications to the initial design, the system obtains state-of-the-art results on this task also.

1. Introduction

The task of providing a principled treatment of speculation and negation is a problem that has received increased interest within the NLP community during recent years. This is witnessed not only by this Special Issue, but also by the themes of several recent shared tasks and dedicated workshops. The Shared Task at the 2010 Conference on Natural Language Learning (CoNLL) has been of central importance in this respect, where the topic was speculation detection for the domain of biomedical research literature

* University of Oslo, Department of Informatics, PB 1080 Blindern, 0316 Oslo, Norway.
E-mail: {erikve,liljao,jread,oe}@ifi.uio.no.

Submission received: 5 April 2011; revised submission received: 30 September 2011; accepted for publication: 2 December 2011.

(Farkas et al. 2010). This particular area has been the focus of much current research, triggered by the release of the BioScope corpus (Vincze et al. 2008)—a collection of scientific abstracts, full papers, and clinical reports with manual annotations of words that signal speculation or negation (so-called **cues**), as well as of the **scopes** of these cues within the sentences. The following examples from BioScope illustrate how sentences are annotated with respect to speculation. Cues are here shown using angle brackets, with braces corresponding to their annotated scopes:

(1) {The specific role of the chromodomain is ⟨unknown⟩} but chromodomain swapping experiments in Drosophila {⟨suggest⟩ that they {⟨might⟩ be protein interaction modules}} [18].

(2) These data {⟨indicate that⟩ IL-10 and IL-4 inhibit cytokine production by different mechanisms}.

Negation is annotated in the same way, as shown in the following examples:

(3) Thus, positive autoregulation is {⟨neither⟩ a consequence ⟨nor⟩ the sole cause of growth arrest}.

(4) Samples of the protein pair space were taken {⟨instead of⟩ considering the whole space} as this was more computationally tractable.

In this article we develop several linguistically informed approaches to automatically identify cues and resolve their scope within sentences, as in the example annotations. Our starting point is the system developed by Velldal, Øvrelid, and Oepen (2010) for the CoNLL-2010 Shared Task challenge. This system implements a two-stage hybrid approach for resolving speculation: First, a binary classifier is applied for identifying cues, and then their in-sentence scope is resolved using a small set of manually defined rules operating on dependency structures.

In the current article we present several important extensions to the initial system design of Velldal, Øvrelid, and Oepen (2010): First, in Section 5, we present a simplified approach to **cue classification**, greatly reducing the model size and complexity of our Support Vector Machine (SVM) classifier while at the same time giving better accuracy. Then, after reviewing the manually defined **dependency-based scope rules** (Section 6.1), we show how the scope resolution task can be handled using an alternative approach based on learning a **discriminative ranking function** over subtrees of HPSG-derived constituent trees (Section 6.2). Moreover, by *combining* this empirical ranking approach with the manually defined rules (Section 6.3), we are able to obtain the best published results so far (to the best of our knowledge) on the CoNLL-2010 Shared Task evaluation data. Finally, in Section 7, we show how our speculation system can be ported to also resolve the scope of *negation*. Only requiring modest modifications, the system also obtains state-of-the-art results on this task. Rather than merely presenting the implementation details of the new approaches we develop, we also provide in-depth error analyses and discussion on the linguistic properties of the phenomena of both speculation and negation.

Before turning to the details of our approach, however, we start by presenting the relevant data sets and the resources used for pre-processing in Section 2, followed by a presentation of the various evaluation measures we will use in Section 3. We also provide a brief review of relevant previous work in Section 4.

2. Data Sets and Preprocessing

Our experiments center on the biomedical abstracts, full papers, and clinical reports of the BioScope corpus (Vincze et al. 2008). This comprises 20,924 sentences (or other root-level utterances), annotated with respect to both negation and speculation. Some basic descriptive statistics for the data sets are provided in Table 1. We see that roughly 18% of the sentences are annotated as uncertain, and 13% contain negations. Note that, for our speculation experiments, we will be using only the abstracts and the papers for training, corresponding to the official CoNLL-2010 Shared Task training data. Moreover, we will be using the Shared Task version of this data, in which certain annotation errors had been corrected. The Shared Task task organizers also provided a set of newly annotated biomedical articles for evaluation purposes, constituting an additional 5,003 utterances. This latter data set (also detailed in Table 1) will be used for held-out testing of our speculation models. We will be using the following abbreviations when referring to the various parts of the data: **BSA** (BioScope abstracts), **BSP** (full papers), **BSE** (the held-out evaluation data), and **BSR** (clinical reports). Note that, when we get to the *negation* task we will be using the *original version* of the BioScope data. Furthermore, as BSE does not annotate negation, we instead follow the experimental set-up of Morante and Daelemans (2009b) for the negation task, reporting 10-fold cross validation on BSA and held-out testing on BSP and BSR.

2.1 Tokenization

The BioScope data (and other data sets in the CoNLL-2010 Shared Task), are provided sentence-segmented only, and otherwise non-tokenized. Unsurprisingly, the GENIA tagger (Tsuruoka et al. 2005) has a central role in our pre-processing set-up. We found that its tokenization rules are not always optimally adapted for the type of text in BioScope, however. For example, GENIA unconditionally introduces token boundaries for some punctuation marks that can also occur token-internally, thus incorrectly splitting tokens like *390,926*, *methlycobamide:CoM*, or *Ca(2+)*. Conversely, GENIA fails to isolate some kinds of opening single quotes, because the quoting conventions assumed in BioScope differ from those used in the GENIA Corpus, and it mis-tokenizes LaTeX-style n- and m-dashes. On average, one in five sentences in the CoNLL training data

Table 1
The top three rows summarize the components of the BioScope corpus—abstracts (BSA), full papers (BSP), and clinical reports (BSR)—annotated for speculation and negation. The bottom row details the held-out evaluation data (BSE) provided for the CoNLL-2010 Shared Task. Columns indicate the total number of sentences and their average length, the number of hedged/negated sentences, the number of cues, and the number of multiword cues. (Note that BSE is not annotated for negation, and we do not provide speculation statistics for BSR as this data set will only be used for the negation experiments.

			Speculation			Negation		
	Sentences	**Length**	**Sentences**	**Cues**	**MWCs**	**Sentences**	**Cues**	**MWCs**
BSA	11,871	26.1	2,101	2,659	364	1,597	1,719	86
BSP	2,670	25.7	519	668	84	339	376	23
BSR	6,383	7.7	–	–	–	865	870	8
BSE	5,003	27.6	790	1,033	87	–	–	–

exhibited GENIA tokenization problems. Our pre-processing approach thus deploys a cascaded finite-state tokenizer (borrowed and adapted from the open-source English Resource Grammar: Flickinger [2002]), which aims to implement the tokenization decisions made in the Penn Treebank (Marcus, Santorini, and Marcinkiewicz 1993)—much like GENIA, in principle—but more appropriately treating corner cases like the ones noted here.

2.2 PoS Tagging and Lemmatization

For part-of-speech (PoS) tagging and lemmatization, we combine GENIA (with its built-in, occasionally deviant tokenizer) and TnT (Brants 2000), which operates on pre-tokenized inputs but in its default model is trained on financial news from the Penn Treebank. Our general goal here is to take advantage of the higher PoS accuracy provided by GENIA in the biomedical domain, while using our improved tokenization and producing inputs to the parsers that as much as possible resemble the conventions used in the original training data for the (dependency) parser (the Penn Treebank, once again).

To this effect, for the vast majority of tokens we can align the GENIA tokenization with our own, and in these cases we typically use GENIA PoS tags and lemmas (i.e., base-forms). For better normalization, we downcase all lemmas except for proper nouns. GENIA does not make a PoS distinction between proper vs. common nouns (as assumed in the Penn Treebank), however, and hence we give precedence to TnT outputs for tokens tagged as nominal by both taggers. Finally, for the small number of cases where we cannot establish a one-to-one correspondence between GENIA tokens and our own tokenization, we rely on TnT annotation only.

2.3 A Methodological Caveat

Unsurprisingly, the majority of previous work on BioScope seems to incorporate information from the GENIA tagger in one way or another, whether it regards tokenization, lemmatization, PoS information, or named entity chunking. Using the GENIA tagger for pre-processing introduces certain dependencies to be aware of, however, as the abstracts in BioScope are in fact also part of the GENIA corpus (Collier et al. 1999) on which the GENIA tagger is trained. This means that the accuracy of the information provided by the tagger on this subset of BioScope cannot be expected to be representative of the accuracy on other texts. Moreover, this effect might of course also carry over to any downstream components using this information.

For the experiments described in this article, GENIA supplies lemmas for the *n*-gram features used by the cue classifiers, as well as PoS tags used in the input to both the dependency parser and the Head-driven Phrase Structure Grammar (HPSG) parser (which in turn provide the inputs to our various scope resolution components). For the HPSG parser, a subset of the GENIA corpus was also used as part of the training data for estimating an underlying statistical parse selection model, producing *n*-best lists of ranked candidate parses (MacKinlay et al. 2011). When reporting final test results on the full papers (BSP or BSE) or the clinical reports (BSR), no such dependencies between information sources exists. It does mean, however, that we can reasonably expect to see some extra drop in performance when going from development results on data that includes the BioScope abstracts to the test results on these other data sets.

3. Evaluation Measures

In this section we seek to clarify the type of measures we will be using for evaluating both the *cue detection* components (Section 3.1) and the *scope resolution* components (Section 3.2). Essentially, we here follow the evaluation scheme established by the CoNLL-2010 Shared Task on speculation detection, also applying this when evaluating results for the negation task.

3.1 Evaluation Measures for Cue Identification

For the approaches presented for cue detection in this article (for both speculation and negation), we will be reporting precision, recall, and F_1 for three different levels of evaluation; the *sentence-level*, the *token-level*, and the *cue-level*. The *sentence-level* scores correspond to Task 1 in the CoNLL-2010 Shared Task, that is, correctly identifying whether a sentence contains uncertainty or not. The scores at the *token-level* measure the number of individual tokens within the span of a cue annotation that the classifier has correctly labeled as a cue. Finally, the stricter *cue-level* scores measure how well a classifier succeeds in identifying entire cues (which will in turn provide the input for the downstream components that later try to resolve the scope of the speculation or negation within the sentence). A true positive at the cue-level requires that the predicted cue exactly matches the annotation in its entirety (full multiword cues included).

For assessing the statistical significance of any observed differences in performance, we will be using a **two-tailed sign-test** applied to the token-level predictions. This is a standard non-parametric test for paired samples, which in our setting considers how often the predictions of two given classifiers differ. Note that we will only be performing significance testing for the token-level evaluation (unless otherwise stated), as this is the level that most directly corresponds to the classifier decisions. We will be assuming a significance level of $\alpha = 0.05$, but also reporting actual p-values in cases where differences are not found to be significant.

3.2 Evaluation Measures for Scope Resolution

When evaluating scope resolution we will be following the methodology of the CoNLL-2010 Shared Task, also using the scoring software made available by the task organizers.[1] We have modified the software trivially so that it can also be used to evaluate negation labeling. As pointed out by Farkas et al. (2010), this way of evaluating scope is rather strict: A **true positive** (TP) requires an exact match for both the entire cue and the entire scope. On the other hand, a **false positive** (FP) can be incurred by three different events; (1) incorrect cue labeling with correct scope boundaries, (2) correct cue labeling with incorrect scope boundaries, or (3) incorrectly labeled cue and scope. Moreover, conditions (1) and (2) will give a double penalty, in the sense that they also count as **false negatives** (FN) given that the gold-standard cue or scope is missed (Farkas et al. 2010). Finally, false negatives are of course also incurred by cases where the gold-standard annotations specify a scope but the system makes no such prediction.

Of course, the evaluation scheme outlined here corresponds to an *end-to-end* evaluation of the overall system, where the cue detection performance carries over to the

1 The Java code for computing the scores can be downloaded from the CoNLL-2010 Shared Task Web site: `http://www.inf.u-szeged.hu/rgai/conll2010st/`.

scope-level performance. In order to better assess the performance of a scope resolution component in isolation, we will also report scope results against *gold-standard cues*. Note that, when using gold-standard cues, the number of false negatives and false positives will always be identical, meaning that the scope-level figures for recall, precision, and F_1 will all be identical as well, and we will therefore only be reporting the latter in this set-up. (The reason for this is that, when assuming gold-standard cues, only error condition (2) can occur, which will in turn always count both a false positive and a false negative, making the two figures identical.)

Exactly how to define the paired samples that form the basis of the statistical significance testing is less straightforward for the end-to-end scope-level predictions than for the cue identification. It is also worth noting that the CoNLL-2010 Shared Task organizers themselves refrained from including any significance testing when reporting the official results. In this article we follow a recall-centered approach: For each cue/scope pair in the gold standard, we simply note whether it is correctly identified or not by a given system. The sequence of boolean values that results (FP = 0, TP = 1) can be directly paired with the corresponding sequence for a different system so that the sign-test can be applied as above.

Note that our modified scorer for negation is available from our Web page of supplemental materials,[2] together with the system output (in XML following the BioScope DTD) for all end-to-end runs with our final model configurations.

4. Related Work on Speculation Labeling

Although there exists a body of earlier work on identifying uncertainty on the *sentence level*, (Light, Qiu, and Srinivasan 2004; Medlock and Briscoe 2007; Szarvas 2008), the task of resolving the *in-sentence scope* of speculation cues was first pioneered by Morante and Daelemans (2009a). In this sense, the CoNLL-2010 Shared Task (Farkas et al. 2010) entered largely uncharted territory and contributed to an increased interest for this task.

Virtually all systems for resolving speculation scope implement a two-stage architecture: First there is a component that identifies the speculation *cues* and then there is a component for resolving the in-sentence *scopes* of these cues. In this section we provide a brief review of previous work on this problem, putting emphasis of the best performers from the two corresponding subtasks of the CoNLL-2010 Shared Task, cue detection (Task 1) and scope resolution (Task 2).

4.1 Related Work on Identifying Speculation Cues

The top-ranked system for Task 1 in the official CoNLL-2010 Shared Task evaluation approached cue identification as a *sequence labeling problem* (Tang et al. 2010). Similarly to the decision-tree approach of Morante and Daelemans (2009a), Tang et al. (2010) set out to label tokens according to a BIO-scheme; indicating whether they are at the Beginning, Inside, or Outside of a speculation cue. In the "cascaded" system architecture of Tang et al. (2010), the predictions of both a Conditional Random Field (CRF) sequence classifier and an SVM-based Hidden Markov Model (HMM) are both combined in a second CRF.

In terms of the overall approach, namely, viewing the problem as a sequence labeling task, Tang et al. (2010) are actually representative of the majority of the Shared Task participants for Task 1 (Farkas et al. 2010), including the top three performers on

2 Supplemental materials; `http://www.velldal.net/erik/modneg/`.

the official held-out data. Many participants instead approached the task as a word-by-word *token classification problem*, however. Examples of this approach are the systems of Velldal, Øvrelid, and Oepen (2010) and Vlachos and Craven (2010), sharing the fourth rank position (out of 24 submitted systems) for Task 1.

In both the sequence- and token-classification approaches, sentences are labeled as uncertain if they are found to contain a cue. In contrast to this, a third group of systems instead label sentences directly, typically using bag-of-words features. Such *sentence classifiers* tended to achieve a somewhat lower relative rank in the official Task 1 evaluation (Farkas et al. 2010).

4.2 Related Work on Resolving Speculation Scope

As mentioned earlier, the task of resolving the scope of speculation was first introduced in Morante and Daelemans (2009a), where a system initially designed for negation scope resolution (Morante, Liekens, and Daelemans 2008) was ported to speculation. Their general approach treats the scope resolution task in much the same way as the cue identification task: as a sequence labeling task and using only token-level, lexical information. Morante, van Asch, and Daelemans (2010) then extended on this system by also adding syntactic features, resulting in the top performing system of the CoNLL-2010 Shared Task at the scope-level (corresponding to the second subtask). It is interesting to note that all the top performers use various types of syntactic information in their scope resolution systems: The output from a dependency parser (MaltParser) (Morante, van Asch, and Daelemans 2010; Velldal, Øvrelid, and Oepen 2010), a tag sequence grammar (RASP) (Rei and Briscoe 2010), as well as constituent analysis in combination with dependency triplets (Stanford lexicalized parser) (Kilicoglu and Bergler 2010). The majority of systems perform classification at the token level, using some variant of machine learning with a BIO classification scheme and a post-processing step to assemble the full scope (Farkas et al. 2010), although several of the top performers employ manually constructed rules (Kilicoglu and Bergler 2010; Velldal, Øvrelid, and Oepen 2010) or even combinations of machine learning and rules (Rei and Briscoe 2010).

5. Identifying Speculation Cues

We now turn to look at the details of our own system, starting in this section with describing a simple yet effective approach to identifying **speculation cues**. A cue is here taken to mean the words or phrases that signal the attitude of uncertainty or speculation. As noted by Farkas et al. (2010), most hedge cues typically fall in the following categories; adjectives or adverbs (*probable*, *likely*, *possible*, *unsure*, etc.), auxiliaries (*may*, *might*, *could*, etc.), conjunctions (*either...or*, etc.), or verbs of speculation (*suggest*, *suspect*, *suppose*, *seem*, etc.). Judging by the examples in the Introduction, it might at first seem that the speculation cues can be identified merely by consulting a pre-compiled list. Most, if not all, words that can function as cues can also occur as non-cues, however. More than 85% of the cue lemmas observed in the BioScope corpus also have non-cue occurrences. To give just one example, a hedge detection system needs to correctly discriminate between the use of *appear* as a cue in Example (5), and as a non-cue in Example (6):

(5) In 5 patients the granulocytes {⟨appeared⟩ polyclonal} [...]

(6) The effect appeared within 30 min and returned to basal levels after 2 h.

In the approach of Velldal, Øvrelid, and Oepen (2010), a binary token classifier was applied in a way that labeled each and every word as *cue* or *non-cue*. We will refer to this mode of classification as **word-by-word classification** (**WbW**). The follow-up experiments described by Velldal (2011) showed that comparable results could be achieved using a **filtering approach** that ignores words not occurring as cues in the training data. This greatly reduces both the number of relevant training examples and the number of features in the model, and in the current article we simplify this "disambiguation approach" even further. In terms of modeling framework, we implement our models as linear SVM classifiers, estimated using the SVM^{light} toolkit (Joachims 1999). We also include results for a very simple baseline model, however—to wit, a WbW approach classifying each word simply based on its observed majority usage as a cue or non-cue in the training data. Then, as for all our models, if a given sentence is found to contain a cue, the entire sentence is subsequently labeled uncertain. Before turning to the individual models, however, we first describe how we deal with the issue of *multiword cues*.

5.1 Multiword Cues

In the BioScope annotations, it is possible for a speculation cue to span multiple tokens (e.g., *raise an intriguing hypothesis*). As seen from Table 1, about 13.5% of the cues in the training data are such **multiword cues** (**MWCs**). The distribution of these cues is very skewed, however. For instance, although the majority of MWCs are very infrequent (most of them occurring only once), the pattern *indicate that* accounts for more than 70% of the cases alone. Exactly which cases are treated as MWCs often seems somewhat arbitrary and we have come across several inconsistencies in the annotations. We therefore choose to not let the classifiers we develop in this article be sensitive to the notion of multiword cues. A given word token is considered a cue as long as it falls within the span of a cue annotation. Multiword cues are instead treated in a separate post-processing step, applying a small set of heuristic rules that aim to capture only the most frequently occurring patterns observed in the training data. For example, if we find that *indicate* is classified as a cue and it is followed by *that*, a rule will fire that ensures we treat these tokens as a single cue. (Note that the rules are only applied to sentences that have already been labeled uncertain by the classifier.) Table 2 lists the lemma patterns currently covered by our rules.

5.2 Reformulating the Classification Problem: A Filtered Model

Before detailing our approach, we start with some general observations about the data and the task. An error analysis of the initial WbW classifier developed by Velldal,

Table 2
Patterns covered by our rules for multiword speculation cues.

cannot {be}? exclude
either .+ or
indicate that
may ,? or may not
no {evidence | proof | guarantee}
not {known | clear | evident | understood | exclude}
raise the .* {possibility | question | issue | hypothesis}
whether or not

Øvrelid, and Oepen (2010) revealed it was not able to generalize to new speculation cues beyond those observed during training. On the other hand, only a rather small fragment of the test cues are actually unseen: Using a 10-fold split for the development data, the average ratio of test cues that also occur as cues in training is more than 90%.

Another important observation we can take into account is that although it seems reasonable to assume that any word occurring as a cue can also occur as a non-cue (recall that more than 85% of the observed cues also have non-cue occurrences in the training data), the converse is less likely. Whereas the training data contains a total of approximately 17,600 unique base forms, only 143 of these ever occur as speculation cues.

As a consequence of these observations, Velldal (2011) proposed that one might reasonably treat the set of cue words as a near-closed class, at least for the biomedical data considered in this study. This means reformulating the problem as follows. Instead of approaching the task as a classification problem defined for all words, we only consider words that have a base form observed as a speculation cue in the training material. By restricting the classifier to only this subset of words, we can simplify the classification problem tremendously. As we shall see, it also has the effect of leveling out the initial imbalance between negative and positive examples in the data, acting as a (selective rather than random) downsampling technique.

One reasonable fear here, perhaps, might be that this simplification comes at the expense of recall, as we are giving up on generalizing our predictions to any previously unseen cues. As noted earlier, however, the initial WbW model of Velldal, Øvrelid, and Oepen (2010) already failed to make any such generalizations, and, as we shall see, this reformulation comes without any loss in performance and actually leads to an increase in recall compared to a full WbW model using the same feature set.

Note that although we will approach the task as a "disambiguation problem," it is not feasible to train separate classifiers for each individual base form. The frequency distribution of the cue words in the training material is rather skewed with most cues being very rare—many occurring as a cue only once ($\approx 40\%$, constituting less than 1.5% of the total number of cue word instances). (Most of these words also have many additional occurrences in the training data as non-cues, however.) For the majority of the cue words, then, it seems we cannot hope to gather enough reliable information to train individual classifiers. Instead, we want to be able to draw on information from the more frequently occurring cues also when classifying or disambiguating the less frequent ones. Consequently, we will still train a single global classifier.

Extending on the approach of Velldal (2011), we include a final simple step to reduce the set of relevant training examples even further. As pointed out in Section 5.1, any token occurring within a cue annotation is initially regarded as a cue word. Many multiword cues also include function words, punctuation, and so forth, however. In order to filter out such spurious but high-frequency "cues," we compiled a small stoplist on the basis of the MWCs in training data (containing just a dozen tokens, namely, *a*, *an*, *as*, *be*, *for*, *of*, *that*, *the*, *to*, *with*, ',', and '-').

5.2.1 Features. After experimenting with a wide range of different features, Øvrelid, Velldal, and Oepen (2010) concluded that syntactic features appeared unnecessary for the cue classification task, and that simple sequence-oriented *n*-gram features recording immediate lexical context based on lemmas and surface forms is what gave the best performance.

Initially, the *n*-gram feature templates we use in the current article record neighbors for up to three positions left/right of the focus word. For increased generality, we also include non-lexicalized variants, that is, recording only the neighbors while excluding

the focus word itself. After a grid search across the various configurations of these features, the best performance was found for a model recording *n*-grams of *lemmas* up to three positions left and right of the focus word, and *n*-grams of *surface forms* up to two positions to the right.

Table 3 shows the performance of the filtering model when using this feature configuration and testing by 10-fold cross-validation on the training data (BSA and BSP), also contrasting performance with the majority usage baseline. Achieving a sentence-level F_1 of 92.04 (compared to 89.07 for the baseline), a token-level score of 89.57 (baseline = 86.42), and a cue-level score of 89.11 (baseline = 85.57), it performs significantly better than the baseline. Applying the sign-test as described in Section 3.1, the token-level differences were found to be significant for $p < 0.05$. It is also clear, however, that the simple baseline appears to be fairly strong.

As discussed previously, part of the motivation for introducing the filtering scheme is to create a model that is as simple as possible without sacrificing performance. In addition to the evaluation scores, therefore, it is also worth noting some statistics related to the classifier and the training data itself. Before looking into the properties of the filtering set-up though, let us start, for the sake of comparison, by considering some properties of a learning set-up based on full WbW classification like the model of Velldal, Øvrelid, and Oepen (2010), assuming an identical feature configuration as used for the given filtering model. The row titled WbW in Table 3 lists the development results for this model, and we see that they are slightly lower than for the filtering model (with the differences being significant for $\alpha = 0.05$). Although precision is slightly higher, recall is substantially lower. Assuming a 10-fold cross-validation scheme like this, the number of training examples presented to the WbW learner in each fold averages roughly 340,000, corresponding to the total number of word tokens. Among these training examples, the ratio of positive to negative examples (cues vs. non-cues) is roughly 1:100. In other words, the data is initially very skewed when it comes to class balance. In terms of the size of the feature set, the average number of distinct feature types per fold, assuming the given feature configuration, would be roughly 2,600,000 under a WbW set-up.

Turning now to the filtering model, the average number of training examples presented to the learner in each fold is reduced from roughly 340,000 to just 10,000. Correspondingly, the average number of distinct feature types is reduced from well above 2,600,000 to roughly 100,000. The *class balance* among the tokens given to the learner is also much less skewed, with positive examples now averaging 30%, compared to 1% for the WbW set-up. Finally, we observe that the *complexity* of the model in terms of how many training examples end up as **support vectors** (SVs) defining the separating hyperplane is also considerably reduced: Although the average number of SVs in each fold corresponds to roughly 14,000 examples for the WbW model, this is down to roughly 5,000 for the final filtered model. Note that for the SVM regularization

Table 3
Development results for detecting speculation CUES: Averaged 10-fold cross-validation results for the cue classifiers on both the abstracts and full papers in the BioScope training data (BSA and BSP).

	Sentence Level			Token Level			Cue Level		
Model	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**
Baseline	91.07	87.21	89.07	91.61	81.85	86.42	90.49	81.16	85.57
WbW	95.01	88.03	91.37	95.29	82.78	88.58	94.65	82.26	88.02
Filtering	94.52	89.72	**92.04**	94.88	84.86	**89.57**	94.13	84.60	**89.11**

parameter *C*, governing the trade-off between training error and margin size, we will always be using the default value set by SVM^{light}. This value is analytically determined from the training data, and further empirical tuning has in general not led to improvements on our data sets.

5.2.2 The Effect of Data Size. Given how the filtered classifier treats the set of cues as a closed class, a reasonable concern is its sensitivity to the size of the training set. In order to further assess this effect, we computed learning curves showing how classifier performance on the development data changes as we incrementally include more training examples (see Figure 1). For reference we also include learning curves for the word-by-word classifier using the identical feature configuration, as well as the majority usage baseline.

As expected, we see that classifier performance steadily improves as more training data is included. Although additional data would no doubt be beneficial, we reassuringly observe that the curve seems to start gradually flattening out somewhat. If we instead look at the performance curve for the WbW classifier we find that, while having roughly the same shape as that of the filtered classifier, although consistently lower, it nonetheless appears to be more sensitive to the size of the training set. Interestingly, we see that the baseline model seems to be the one that is least affected by data size. It actually outperforms the standard WbW model for the first three increments, but at the same time it seems unable to benefit much at all from additional data.

5.2.3 Error Analysis. When looking at the distribution of errors at the cue-level (totaling just below 700 across the 10-fold run), we find that roughly 74% are false negatives. Rather than being caused by legitimate cue words being filtered out during training, however, the FNs mostly pertain to a handful of high-frequency words that are also highly ambiguous. When sorted according to error frequency, the top four candidates alone constitute almost half the total number of FNs: *or* (24% of the FNs), *can* (10%), *could* (7%), and *either* (6%). Looking more closely at the distribution of these words in

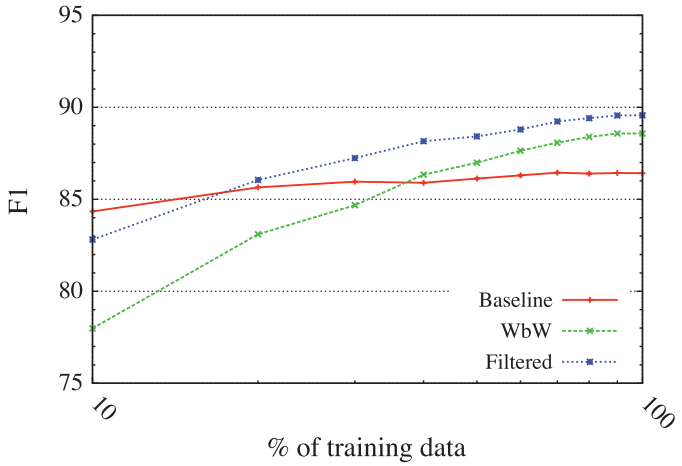

Figure 1
Learning curves showing the effect on token-level F_1 for speculation cues when withdrawing some portion of the training partitions across the 10-fold cycles. The size of the training set is shown on a logarithmic scale to better see whether improvements are constant for *n*-fold increases of data.

the training data, it is easy to see how they pose a challenge for the learner. For example, whereas *or* has a total of 1,215 occurrences, only 153 of these are annotated as a cue. Distinguishing the different usages from each other can sometimes be difficult even for a human eye, as testified also by the many inconsistencies we observed in the gold-standard annotation of these cases.

Turning our attention to the other end of the tail, we find that just over 40 (8%) of the FNs involve tokens for which there is only a *single* occurrence as a cue in the training data. In other words, these would first appear to be exactly the tokens that we could never get right, given our filtering scheme. We find, however, that most of these cases regard tokens whose one and only appearance as a cue is as part of a multiword cue, although they typically have a high number of other *non*-cue occurrences as well. For example, although *number* occurs a total of 320 times, its one and only occurrence as a cue is in the multiword cue *address a number of questions*. Given that this and several other equally rare patterns are not currently covered by our MWC rules in the first place, we would not have been able to get them right even if all the individual tokens had been classified as cues (recall that a true positive at the cue-level requires an exact match of the entire span). In total we find that 16% of the cue-level FNs corresponds to multiword cues.

When looking at the frequency of multiword cues among the false positives, we find that they only make up roughly 5% of the errors. Furthermore, a manual inspection reveals that they can all be argued to be instances of annotation errors, in that we believe these should actually be counted as true positives. Most of them involve *indicate that* and *not known*, as in the following examples (where the cues assigned by our system are not annotated as cues in BioScope):

(7) In contrast, levels of the transcriptional factor AP-1, which is ⟨not known⟩ to be important in B cell Ig production, were reduced by TGF-beta.

(8) Analysis of the nuclear extracts [...] ⟨indicated that⟩ the composition of NF-kappa B was similar in neonatal and adult cells.

All in all, the errors in the FP category make up 26% of the total number of errors. Just as for the FNs, the frequency distribution of the cues involved is quite skewed, with a handful of highly frequent and highly ambiguous cue words accounting for the bulk of the errors: The modal *could* (20%), and the adjectives *putative* (11%), *possible* (6%), *potential* (6%), and *unknown* (5%). After manually inspecting the full set of FPs, however, we find that at least 60% of them should really be counted as true positives. The following are just a few examples where cues predicted by our classifier are not annotated as such in BioScope and therefore counted as FPs.

(9) IEF-1, a pancreatic beta-cell type-specific complex ⟨believed⟩ to regulate insulin expression, is demonstrated to consist of at least two distinct species, [...]

(10) We ⟨hypothesize⟩ that a mutation of the hGR glucocorticoid-binding domain is the cause [...]

(11) Antioxidants have been ⟨proposed⟩ to be anti-atherosclerotic agents; [...]

(12) Finally, matDCC might be further stabilized by the addition of roX1 RNA, which could interact with several of the MSLs and ⟨perhaps⟩ roX2 RNA as well.

One interesting source of real FPs concerns "anti-hedges," which in the training data appear with a negation and as part of a multiword cue, for example *no proof*. During testing, the classifier will sometimes wrongly predict a word like *proof* to be a speculation cue, even when it is not negated. Because we already have MWC rules for cases like this (see Section 5.1) it would be easy to also include a check for "negative context," making sure that such tokens are not classified as cues if the required multiword context is missing.

Before rounding off this section, a brief look at the BioScope inter-annotator agreement rates may offer some further perspective on the results discussed here. Note that when creating the BioScope data, the decisions of two independent annotators were merged by a third expert linguist who resolved any differences. The F_1 of each set of annotations toward the final gold-standard cues are reported by Vincze et al. (2008) to be 83.92 / 92.05 for the abstracts and 81.49 / 90.81 for the full papers. (Recall from Table 3 that our cue-level F_1 for the cross-validation runs on the abstracts and papers is 89.11.) When instead comparing the decisions of the two annotators directly, the F_1 is reported to be 79.12 for the abstracts and 77.60 for the papers.

5.3 Held-Out Results for Identifying Speculation Cues

Table 4 presents the final evaluation of the various cue classifiers developed in this section, as applied to the held-out BSE test data. In addition to the evaluation results for our own classifiers, Table 4 also includes the official test results for the system described by Tang et al. (2010). The sequence classifier developed by Tang et al. (2010)—combining a CRF classifier and a large-margin HMM model—obtained the best results for the official Shared Task evaluation for Task 1 (i.e., sentence-level uncertainty detection), as well as the highest cue-level scores.

As seen from Table 4, although the model of Tang et al. (2010) still achieves a slightly higher F_1 (81.34) than our filtered disambiguation model for the cue-level, our model achieves a slightly higher F_1 (86.58) for the sentence-level (yielding the best-published result for this task so far, to the best of our knowledge). The differences are not deemed statistically significant by a two-tailed sign-test, however ($p = 0.37$). It is interesting to note, however, that the two approaches appear to have somewhat different strengths and weaknesses: Whereas our filtering classifier consistently shows stronger precision (and the WbW model even more so), the model of Tang et al. (2010) is stronger on recall. The sentence-level recall of our filtered classifier is still better than any of the remaining 23 systems submitted for the Shared Task evaluation, however, and, more interestingly, it improves substantially on the recall of the full WbW classifier.

Table 4
Held-out results for identifying speculation cues: Applying the cue classifiers to the 5,003 sentences in BSE— the biomedical papers provided for the CoNLL-2010 Shared Task evaluation.

	Sentence Level			Token Level			Cue Level		
Model	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**
Baseline	77.59	81.52	79.51	77.16	72.39	74.70	75.15	72.49	73.80
WbW	**89.28**	83.29	86.18	**87.62**	73.95	80.21	**86.33**	74.21	79.82
Filtering	87.87	85.32	**86.58**	86.46	**76.74**	**81.31**	84.79	77.17	80.80
Tang et al. 2010	85.03	**87.72**	86.36	n/a	n/a	n/a	81.70	**80.99**	**81.34**

We find that, just as for the development data, the reformulation of the cue classification task as a simple disambiguation problem improves F_1 across all evaluation levels, consistently outperforming the WbW classifiers. When computing a two-tailed signed-test for the token-level decisions (where the WbW and filtering model achieves an F_1 of 80.21 and 81.31, respectively) the differences are not found to be significant ($p = 0.12$). As discussed in Section 5.2, however, it is important to bear in mind that the size and complexity of the filtered "disambiguation" model is greatly reduced compared to the WbW model, using a much smaller number of features and relevant training examples.

While on the topic of model complexity, it is also worth noting that many of the systems participating in the CoNLL-2010 Shared Task challenge used fairly complex and resource-heavy feature types, being sensitive to properties of document structure, grammatical relations, deep syntactic structure, and so forth (Farkas et al. 2010). The fact that comparable or better results can be obtained using a relatively simplistic approach as developed in this section, with surface-oriented features that are only sensitive to the immediate lexical context, is an interesting result in its own right. In fact, even the simple majority usage baseline classifier proves to be surprisingly competitive: Comparing its sentence-level F_1 to those of the official Shared Task evaluation, it actually outranks 7 of the 24 submitted systems.

A final point that deserves some discussion is the drop in F_1 that we observe when going from the development results to the held-out results. There are several reasons for this drop. Section 2.3 discussed how certain overfitting effects might be expected from the GENIA-based pre-processing. In addition to this, it is likely that there are MWC patterns in the held-out data that were not observed in the training data, and that are therefore not covered by our MWC rules. Another factor that may have slightly inflated the development results is the fact that we used a sentence-level rather than a document-level partitioning of the data for cross-validation.

6. Resolving the Scope of Speculation Cues

Once the speculation cue has been determined using the cue detection system described here, we go on to determine the scope of the speculation within the sentence. This task corresponds to Task 2 of the CoNLL-2010 Shared Task. Example (13), which will be used as a running example throughout this section, shows a scope-resolved BioScope sentence where speculation is signaled by the modal verb *may*.

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

The exact scope will vary quite a lot depending on linguistic properties of the cue in question, and in our approaches to scope resolution we rely heavily on syntactic information. We experiment with two different approaches to syntactic analysis; **data-driven dependency parsing** and **grammar-driven phrase structure parsing**. Because scope determination in BioScope makes reference to subtle and fine-grained linguistic distinctions (e.g., passivization or subject raising), in both cases we choose parsing systems that make available comparatively "deep" syntactic analyses. In the following we present three different systems; a rule-based approach using dependency structures (Section 6.1), a data-driven approach using an SVM ranker for selecting appropriate subtrees in constituent structures (Section 6.2), and finally a hybrid approach combining the rules and the ranker (Section 6.3).

6.1 A Rule-Based Approach Using Dependency Structures

Øvrelid, Velldal, and Oepen (2010) applied a small set of heuristic rules operating over syntactic dependency structures to define the scope for each cue. In the following we will provide a detailed description of these rules and the syntactic generalizations they provide for the scope of speculation (Section 6.1.2). We will evaluate their performance using both gold-standard cues and cues predicted by our cue classifier (Section 6.1.3), in addition to providing an in-depth manual error analysis (Section 6.1.5). We start out, however, by presenting some specifics about the processing of the data; introducing the stacked dependency parser that produces the input to our rules (Section 6.1.1) and quantifying the effect of using a domain-adapted PoS tagger (Section 6.1.4).

6.1.1 Stacked Dependency Parsing. For syntactic analysis we use the open-source MaltParser (Nivre, Hall, and 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 Lexical-Functional Grammar parser. A technique dubbed **parser stacking** enables the data-driven parser to learn from the output of another parser, in addition to gold-standard treebank annotations (Martins et al. 2008; Nivre and McDonald 2008). This technique has been shown to provide significant improvements in accuracy for both English and German (Øvrelid, Kuhn, and Spreyer 2009), and a similar set-up using an HPSG grammar has been shown to increase domain independence in data-driven dependency parsing (Zhang and Wang 2009). The stacked parser used here is identical to the parser described in Øvrelid, Kuhn, and Spreyer (2009), except for the preprocessing in terms of tokenization and PoS tagging, which is performed as detailed in Sections 2.1–2.2. The 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 to draw upon for the speculation 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 history in order to guide parsing and may easily be extended to take into account additional features. 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 et al. 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. For a description of the parse model features and the dependency substructures proposed by XLE for each word token, see Nivre and McDonald (2008). For further background on the conversion and training procedures, see Øvrelid, Kuhn, and Spreyer (2009).

Table 5 shows the enhanced dependency representation for the sentence in Example (13). For each token, the parsed data contains information on the word form, lemma, and PoS, as well as the head and dependency relation (last two columns). The added XLE information resides in the *Features* column and in the XLE-specific head and dependency columns (*XHead* and *XDep*). Parser outputs, which in turn form the basis for our scope resolution rules, also take this same form. The parser used in this work is trained on the Wall Street Journal Sections 2–24 of the Penn Treebank (PTB), converted

Table 5
Stacked dependency representation of the sentence in Example (13), lemmatized and annotated with GENIA PoS tags, Malt parses (Head, DepRel), and XLE parses (XHead, XDep), as well as other morphological and lexical semantic features extracted from the XLE analysis (Features).

Id	Form	PoS	Features	XHead	XDep	Head	DepRel
1	The	DT	_	4	SPECDET	4	NMOD
2	unknown	JJ	degree:attributive	4	ADJUNCT	4	NMOD
3	amino	JJ	degree:attributive	4	ADJUNCT	4	NMOD
4	acid	NN	pers:3|case:nom|num:sg|ntype:common	3	SUBJ	5	SBJ
5	may	MD	mood:ind|subcat:MODAL|tense:pres|clauseType:decl	0	ROOT	0	ROOT
6	be	VB	_	7	PHI	5	VC
7	used	VBN	subcat:V-SUBJ-OBJ|vtype:main|passive:+	5	XCOMP	6	VC
8	by	IN	_	9	PHI	7	LGS
9	these	DT	deixis:proximal	10	SPECDET	10	NMOD
10	species	NNS	num:pl|pers:3|case:obl|common:count|ntype:common	7	OBL-AG	8	PMOD
11	.	.	_	0	PUNC	5	P

to dependency format (Johansson and Nugues 2007) and extended with XLE features, as described previously. 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, which is lower than the current state-of-the-art for transition-based dependency parsers (to wit, the 91.8 score of Zhang and Nivre 2011, although not directly comparable given that they test exclusively on WSJ Section 23), but with the advantage of providing us with the deep linguistic information from the XLE.

6.1.2 Rule Overview. Our scope resolution rules take as input a parsed sentence that has been further tagged with speculation cues. We assume the **default scope** to start at the cue word and span to the end of the sentence (modulo punctuation), and this scope also provides the baseline when evaluating our rules.

In developing the 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. In the following, we discuss broad classes of rules, organized by categories of speculation cues. An overview is also provided in Table 6, detailing the source of the syntactic information used by the rule; MaltParser (M) or XLE (X). Note that, as there is no explicit representation of phrase or clause boundaries in our dependency universe, we assume a set of functions over dependency graphs, for example, finding the left- or rightmost (direct) *dependent* of a given node, or recursively 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, that is, start scope is set to the leftmost dependent of the head of the coordination, and end scope is set to its rightmost dependent (conjunct). This analysis provides us with coordinations at various syntactic levels, such as NP and $\overline{\text{N}}$, AP and AdvP, or VP as in Example (14):

(14) [...] the binding interfaces are more often {kept ⟨or⟩ even reused} rather than lost in the course of evolution.

Table 6
Overview of dependency-based scope rules with information source (MaltParser or XLE), organized by the triggering PoS of the cue.

PoS	Description	Source
CC	Coordinations scope over their conjuncts	M
IN	Prepositions scope over their argument with its descendants	M
JJ_{attr}	Attributive adjectives scope over their nominal head and its descendants	M
JJ_{pred}	Predicative adjectives scope over referential subjects and clausal arguments, if present	M, X
MD	Modals inherit subject-scope from their lexical verb and scope over their descendants	M, X
RB	Adverbs scope over their heads with its descendants	M
VB_{pass}	Passive verbs scope over referential subjects and the verbal descendants	M, X
VB_{rais}	Raising verbs scope over referential subjects and the verbal descendants	M, X
*	For multiword cues, the head determines scope for all elements	
*	Back off from final punctuation and parentheses	

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 Example (15):

(15) The {⟨possible⟩ selenocysteine residues} are shown in red, [...]

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 Example (16). The scope does not, however, include expletive subjects, as in Example (17).

(16) Therefore, {the unknown amino acid, if it is encoded by a stop codon, is ⟨unlikely⟩ to exist in the current databases of microbial genomes}.

(17) [...] 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 Example (18), unless it is an expletive pronoun, as in Example (19).

(18) {Genomes of plants and vertebrates ⟨seem⟩ to be free of any recognizable Transib transposons} (Figure 1).

(19) 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 speculation cue, specifically modals, subject arguments are included in scopes where the controller heads a passive construction or a raising verb, as in our running Example (13).

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.

Prepositions and Adverbs. Cues that are tagged as prepositions (including some complementizers) take scope over their argument, with all its descendants, Example (20). Adverbs take scope over their head with all its (non-subject) syntactic descendants Example (21).

(20) {⟨Whether⟩ the codon aligned to the inframe stop codon is a nonsense codon or not} was neglected [...]

(21) These effects are {⟨probably⟩ mediated through the 1,25(OH)2D3 receptor}.

Multiword Cues. In the case of multiword cues, such as *indicate that* or *either...or*, we set the scope of the unit as a whole to the maximal scope encompassing the scopes of both units.

As an illustration of processing by the rules, consider our running Example (13), with its syntactic analysis as shown in Table 5 and the dependency graph depicted in Figure 2. This example invokes a variety of syntactic properties, including parts of speech, argumenthood, voice, and so on. Initially, the scope of the speculation cue is set to default scope. Then the subject control rule is applied, it checks the properties of the verbal argument *used*, going through a chain of verbal dependents (VC) from the modal verb *may* (indicated in red in Figure 2). Because it is marked as passive in the LFG analysis (+pass), the start scope is set to include the subject of the cue word (the leftmost descendant [NMOD] of its SBJ dependent, indicated in green in Figure 2).

6.1.3 Evaluating the Rules. Table 7 summarizes scope resolution performance (viewed as a subtask in isolation) against both the CoNLL-2010 shared task training data (BSA and BSP) and held-out evaluation data (BSE), using *gold-standard cues*. First of all, we note that the default scope baseline, that is, unconditionally extending the scope of a cue to the end of the sentence, yields much better results for the abstracts than the full papers. The main reason is simply that the abstracts contain almost no cases of sentence final bracketed expressions (e.g., citations and in-text references). Our scope rules improve on the baseline by only 3.8 percentage points on the BSA data (F_1 up from 69.84 to

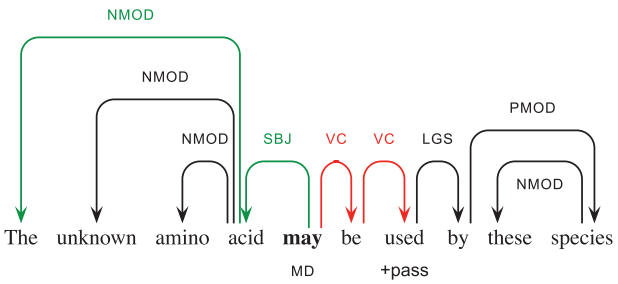

Figure 2
Dependency representation for Example (13), indicating rule processing of the cue word *may*.

Table 7
Resolving the scope of gold-standard speculation cues in the development and held-out data using the dependency rules. For *Default*, the scope for each cue is always taken to span rightwards to the end of the sentence.

Data	Configuration	F_1
BSA	Default	69.84
	Dependency Rules	73.67
BSP	Default	45.21
	Dependency Rules	72.31
BSE	Default	46.95
	Dependency Rules	66.60

73.67). For BSP, however, we find that the rules improve on the baseline by as much as 27 points (up from 45.21 to 72.31). Similarly for the papers in the held-out BSE data, the rules improve the F_1 by 19.7 points (F_1 up from 46.95 to 66.60).

Comparing to the result on the training data, we observe a substantial drop in performance on the held-out data. There are several possible explanations for this effect. First of all, there may well be some degree of overfitting of our rules to the training data. The held-out data may contain speculation 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 Section 6.1.5). The overfitting effects caused by the data dependencies introduced by the various GENIA-based domain adaptation steps, as described in Section 2.3, must also be taken into account.

6.1.4 PoS Tagging and Domain Variation. As mentioned in Section 6.1.1, an advantage of stacking with a general-purpose LFG parser is that it can be expected to aid domain portability. Nonetheless, substantial differences in domain and genre are bound to negatively affect syntactic analysis (Gildea 2001), and our parser is trained on financial news. MaltParser presupposes that inputs have been PoS tagged, however, leaving room for variation in preprocessing. In this article we have aimed, on the one hand, to make parser inputs conform as much as possible to the conventions established in its PTB training data, while on the other hand taking advantage of specialized resources for the biomedical domain.

To assess the impact of improved, domain-adapted inputs on our scope resolution rules, we contrast two configurations: Running the parser in the exact same manner as Øvrelid, Kuhn, and Spreyer (2009)—the first configuration uses TreeTagger (Schmid 1994) and its standard model for English (trained on the PTB) for preprocessing. In the second configuration the parser input is provided by the refined GENIA-based preprocessing described in Section 2.2. Evaluating the two modes of preprocessing on the BSP subset of BioScope using gold-standard speculation cues, our scope resolution rules achieve an F_1 of 66.31 when using TreeTagger parser inputs, and 72.31 (see Table 7) using our GENIA-based tagging and tokenization combination. These results underline the importance of domain adaptation for accurate syntactic analysis.

6.1.5 Error Analysis. In Section 5.2.3 we discussed BioScope inter-annotator agreement rates for the cue-level. Focusing only on the cases where the annotators agree with the final gold-standard cues (as resolved by the chief annotator), Vincze et al. (2008) report

the *scope-level* F_1 of the two annotators toward the gold standard to be 66.72 / 89.67 for BSP. Comparing the decisions of the two annotators directly (i.e., treating one of the annotations as gold-standard) yields an F_1 of 62.50.

Using gold-standard cues, our scope resolution rules fail to exactly replicate the target annotation in 185 (of 668) cases in the papers portion of the training material (BSP), corresponding to an F_1 of 72.31 as seen in Table 7. Two of the authors, who are both trained linguists, performed a manual error analysis of these 185 cases. They classify 156 (84 %) as genuine system errors, 22 (12 %) as likely[3] annotation errors, and the remaining 7 cases as involving controversial or seemingly arbitrary decisions (Øvrelid, Velldal, and Oepen 2010). Out of the 156 system errors, 85 (55%) were deemed as resulting from missing or defective rules, and 71 system errors (45%) resulted from parse errors. The latter were annotated as parse errors even in cases where there was also a rule error.

The two most frequent classes of system errors pertain to (a) the recognition of phrase and clause boundaries and (b) not dealing successfully with relatively superficial properties of the text. Examples (22) and (23) 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.

(22) [. . .] {the reverse complement |mR of m will be ⟨considered⟩ to . . .]|}

(23) 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. In Example (22), for instance, the parser has failed to correctly locate the head of the subject. Example (23), however, is specifically indicative of a key challenge in this task, where adverbials of condition, reason, or contrast frequently attach within the dependency domain of a speculation cue, yet are rarely included in the scope annotation. For these system errors, the syntactic analysis may well be correct, although additional information is required to resolve the scope.

Example (24) 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 (e.g., Examples [18] or [19]); 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 cases such as Example (24), however, syntax does not make explicit the contrast between an in-text reference versus another type of parenthetical.

(24) More specifically, {|the bristle and leg phenotypes are ⟨likely⟩ to result from reduced signaling by Dl| (and not by Ser)}.

3 In some cases, there is no doubt that annotation is erroneous, that is, 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 our own generalizations (e.g., assumptions about syntactic analyses implicit in the BioScope annotations). Furthermore, selecting items for manual analysis that do not align with the 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.

Moving on to apparent annotation errors, the rules for inclusion (or not) of the subject in the scope of verbal speculation cues and decisions on boundaries (or internal structure) of nominals seem problematic—as illustrated in Examples (25) and (26).[4]

(25) [...] and |this is also {⟨thought⟩ to be true for the full protein interaction networks we are modeling}|.

(26) [...] |redefinition of {one of them is ⟨feasible⟩}|.

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

6.2 A Data-Driven Approach Using an SVM Constituent Ranker

The error analysis indicated that it is often difficult to use dependency paths to define phenomena that actually correspond to syntactic constituents. Furthermore, we felt that the factors governing scope resolution would be better expressed in terms of soft constraints instead of absolute rules, thus enabling the scope resolver to consider a range of relevant (potentially competing) contextual properties. In this section we describe experiments with a novel approach to determining the in-sentence scope of speculation that, rather than using manually defined heuristics operating on dependency structures, instead uses a *data-driven* approach, ranking candidate scopes on the basis of *constituent trees*. More precisely, our parse trees are licensed by the LinGO **English Resource Grammar** (ERG; Flickinger [2002]), a general-purpose, wide-coverage grammar couched in the framework of an HPSG (Pollard and Sag 1987, 1994). The approach rests on two main assumptions: Firstly, that the annotated scope of a speculation cue corresponds to a syntactic constituent and secondly, that we can automatically learn a ranking function that selects the correct constituent.

Our ranking approach to scope resolution is abstractly related to statistical *parse selection*, and in particular work on discriminative parse selection for unification based grammars, such as those by Johnson et al. (1999), Riezler et al. (2002), Malouf and van Noord (2004), and Toutanova et al. (2005). The overall goal is to learn a function for ranking syntactic structures, based on training data that annotates which tree(s) are correct and incorrect for each sentence. In our case, however, rather than discriminating between complete analyses for a given sentence, we want to learn a ranking function over candidate *subtrees* (i.e., constituents) within a parse (or possibly even within several parses). Figure 3 presents an example derivation tree that represents a complete HPSG analysis. Starting from the cue and working through the tree bottom–up, there are three candidate constituents to determine scope (marked in bold), each projecting onto a substring of the full utterance, and each including at least the cue. Note that in the case of multiword cues the intersection of each word's candidates is selected, ensuring that all cues appear within the scope projected by the candidate constituents.

The training data is then defined as follows. Given a parsed BioScope sentence, the subtree that corresponds to the annotated scope for a given speculation cue will

4 As 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.

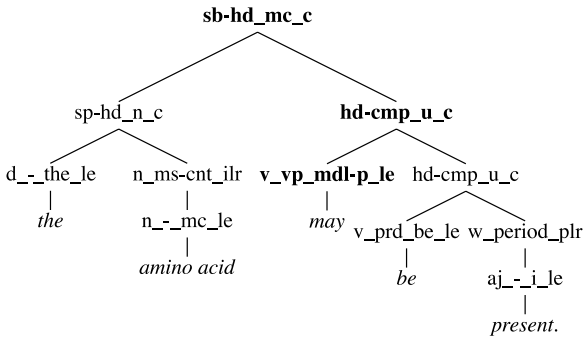

Figure 3
An example derivation tree. Internal nodes are labeled with ERG rule identifiers; common HPSG constructions near the top (e.g., subject-head, head-complement, adjunct-head), and lexical rules (e.g., passivization of verbs or plural formation of nouns) closer to the leaves. The preterminals are so-called LE types, corresponding to fine-grained parts of speech and reflecting close to a thousand lexical distinctions.

be labeled as correct. Any other remaining constituents that also span the cue are labeled as incorrect. We then attempt to learn a linear SVM-based scoring function that reflects these preferences, using the implementation of ordinal ranking in the SVM^{light} toolkit (Joachims 2002). Our definition of the training data, however, glosses over two important details.

Firstly, the grammar will usually license not just one, but thousands or even hundreds of thousands of different parses for a given sentence which are ranked by an underlying parse selection model. Some parses may not necessarily contain a subtree that aligns with the annotated scope. We therefore experiment with defining the training data relative to *n*-best lists of available parses. Secondly, the rate of alignment between annotated scopes and constituents of parsing results indicates the upper-bound performance: For inputs where no constituents align with the correct scope substring, a correct prediction will not be possible. Searching the *n*-best parses for alignments enables additional instances of scope to be presented to the learner, however.

In the following, Section 6.2.1 summarizes the general parsing setup for the ERG, as well as our rationale for the use of HPSG. Section 6.2.2 provides an empirical assessment of the degree to which ERG analyses can be aligned with speculation scopes in BioScope and reviews some frequent sources of alignment failures. After describing our feature types for representing candidate constituents in Section 6.2.3, Section 6.2.4 details the tuning of feature configurations and other ranker parameters. Finally, Section 6.2.5 provides an empirical assessment of stand-alone ranker performance, before we discuss the integration of the dependency rules with the ranking approach in Section 6.3.

6.2.1 Basic Set-up: Parsing Biomedical Text Using the ERG. At some level of abstraction, the approach to grammatical analysis embodied in the ERG is quite similar to the LFG parser that was "stacked" with our data-driven dependency parser in Section 6.1.1—both are commonly considered comparatively "deep" (and thus costly) approaches to syntactic analysis. Judging from the BioScope annotation guidelines, subtle grammatical distinctions are at play when determining scopes, for example, different types of control verbs, expletives, or passivization (Vincze et al. 2008). In contrast to the LFG framework (with its distinction between so-called constituent and functional structures), the analyses provided by the ERG offer the convenience of a single syntactic

representation—HPSG derivation trees, as depicted in Figure 3—where all contextual information that we expect to be relevant for scope resolution is readily accessible.

For parsing biomedical text using the ERG, we build on the same preprocessing pipeline as described in Section 2. A lattice of tokens annotated with parts of speech and named entity hypotheses contributed by the GENIA tagger is input to the PET HPSG parser (Callmeier 2002), a unification-based chart parser that first constructs a packed forest of candidate analyses and then applies a discriminative parse ranking model to selectively enumerate an *n*-best list of top-ranked candidates (Zhang, Oepen, and Carroll 2007). To improve parse selection for this kind of data, we re-trained the discriminative model following the approach of MacKinlay et al. (2011), combining gold-standard out-of-domain data from existing ERG treebanks with a fully automated procedure seeking to take advantage of syntactic annotations in the GENIA Treebank. Although we have yet to pursue domain adaptation in earnest and have not systematically optimized the parse selection component for biomedical text, model re-training contributed about a one-point F_1 improvement in stand-alone ranker performance over the parsed subset of BSP (compare to Table 8).

As the ERG has not previously been adapted to the biomedical domain, unknown word handling in the parser plays an important role. Here we build on a set of somewhat underspecified "generic" lexical entries for common open-class categories provided by the ERG (thus complementing the 35,000-entry lexicon that comes with the grammar), which are activated on the basis of PoS and NE annotation from preprocessing. Other than these, there are no robustness measures in the parser, such that syntactic analysis will fail in a number of cases, to wit, when the ERG is unable to derive a complete, well-formed syntactic structure for the full input string. In this configuration, the parser returns at least one derivation for 91.2% of all utterances in BSA, and 85.6% and 81.4% for BSP and BSE, respectively.

6.2.2 Alignment of Constituents and Scopes. The constituent ranking approach makes explicit an assumption that is also present at the core of our dependency-based heuristics (viz., the expectation that scope boundaries align with the boundaries of syntactically meaningful units). This assumption is motivated by general BioScope annotation principles, as Vincze et al. (2008) suggest that the "scope of a keyword can be determined on the basis of syntax." To determine the degree to which ERG analyses conform to this expectation, we computed the ratio of alignment between scopes and constituents (over parsed sentences) in BioScope, considering various sizes of *n*-best lists of parses. To improve alignment we also apply a small number of **slackening** heuristics. These rules allow (a) minor adjustments of scope boundaries around punctuation marks

Table 8
Ranker optimization on BSP: Showing ranker performance for various feature type combinations compared with a random-choice baseline, only considering instances where the gold-standard scope aligns to a constituent within the 1-best parse.

Features	**F1**
Baseline	26.76
Path	78.10
Path+Surface	79.93
Path+Linguistic	83.72
Path+Surface+Linguistic	85.30

(specifically, utterance-final punctuation is never included in BioScope annotations, yet the ERG analyzes most punctuation marks as pseudo-affixes on lexical tokens; see Figure 3). Furthermore, the slackening rules (b) reduce the scope of a constituent to the right when it includes a citation (see the discussion of parentheticals in Section 6.1.5); (c) reduce the scope to the left when the left-most terminal is an adverb and is not the cue; and (d) ensure that the scope starts with the cue when the cue is a noun. Collectively, these rules improve alignment (over parsed sentences) in BSP from 74.10% to 80.54%, when only considering the syntactic analysis ranked most probable by the parse selection model. Figure 4 further depicts the degree of alignment between speculation scopes and constituents in the *n*-best derivations produced by the parser, again after application of the slackening rules. Alignment when inspecting only the top-ranked parse is 84.37% for BSA and 80.54% for BSP. Including the top 50-best derivations improves alignment to 92.21% and 88.93%, respectively. Taken together with an observed parser coverage of 85.6% for BSP, these results mean that for only about 76% of all utterances in BSP can the ranker potentially identify a constituent matching the gold-standard scope.

To shed some light on the cases where we *fail* to find an alignment, we manually inspected all utterances in the BSP segment for which there were (a) syntactic analyses available from the ERG and (b) no candidate constituents in any of the top-fifty parses that mapped onto the gold-standard scope (after the application of the slackening rules). The most interesting cases from this non-alignment analysis are ones judged as "non-syntactic" (25% of the total mismatches), which we interpret as violating the assumption of the annotation guidelines under any possible interpretation of syntactic structure. Following are select examples in this category:

(27) This allows us to {⟨address a number of questions⟩: what proportion of each organism's protein interaction network [...] can be attributed to a known domain-domain interaction}?

(28) As {⟨suggested⟩ in 18, by making more such data sets available, it will be possible to [...] determine the most likely human interactions}.

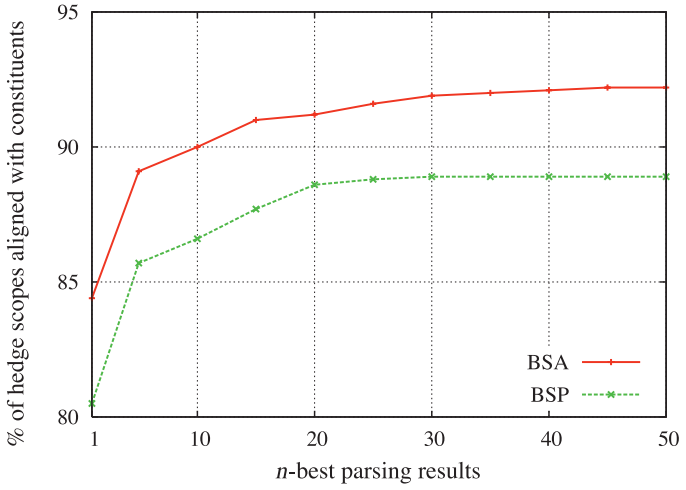

Figure 4
The effect of incrementally including additional derivations from the *n*-best list when searching for an alignment between a speculation scope and a constituent. Plots are shown for the BSA and BSP subsets of the training data.

(29) The {lack of specificity ⟨might⟩ be attributed to a number of reasons, such as the absence of other MSL components, the presence of other RNAs interacting with MOF}, or worse [...].

(30) [...] thereby making {the use of this objective function — and exploration of other complex objective functions — ⟨possible⟩}.

Example (27) is representative of a handful of similar cases, where complete sentences are (implicitly or overtly) conjoined, yet the scope annotation encompasses only part of one of the sentences. Example (28) is in a similar spirit, only in this case a topicalized prepositional phrase (and hence an integral constituent) is only partially included in the gold-standard scope. Although our slackening heuristics address a number of cases of partial noun phrases (with a left scope boundary right before the head noun or a pre-head attributive adjective), another handful of non-syntactic scopes are of the type exemplified by Example (29), a class observed earlier already in the error analysis of our dependency-based scope resolution rules (see Section 6.1.5). Finally, Example (30) demonstrates one of many linguistically subtle corner cases: The causative *make* in standard analyses of the resultative construction takes two arguments, namely, an NP (*the use of this objective function...*) and a predicative phrase (*possible*).

Alongside cases like these, our analysis considered 16% of mismatches owed to divergent syntactic theories (i.e., structures that in principle can be analyzed in a manner compatible with the BioScope gold-standard annotations, yet do not form matching constituents in the ERG analyses). The by far largest class of mismatches was attributed to parse ranking deficiencies: In close to 40% of cases, the ERG is capable of deriving a constituent structure compatible with the scope annotations, but no such analysis was available within the top 50 parses. Somewhat reassuringly, less than 6% of all mismatches were classified as BioScope annotation errors, whereas a majority of remaining mismatches are owed to the recurring issue of parentheticals and bibliographic references (see examples in Section 6.1.5).

6.2.3 Features of Candidate Scopes. We use three families of features to describe candidate constituents. Given our working hypothesis that scopes are aligned with syntactic constituents, the most natural features to use are the location of constituents within trees. We define these in terms of the paths from speculation cues to candidate constituents. For example, the correct candidate in Figure 3 has the feature `v_vp_mdl-p_le\hd-cmp_u_c\sb-hd_mc_c`. We include both lexicalized and unlexicalized versions of this feature. As traversal from the cue to the candidate can involve many nodes, we also include a more general version recording only the cue and the root of the candidate constituent (rather than the full path including all intermediate nodes). In a similar spirit we also generate bigram features for each path node and its parent.

In addition to the given path features, we also exploit features describing the surface properties of scope candidates. These include the enumeration of bigrams of the preterminal lexical types, the cue position within the candidate (in tertile bins relative to the candidate length), and the candidate size (in quartile bins relative to the sentence length). Because punctuation may also be informative for scope resolution, we also record whether punctuation was present at the end of the terminal preceding the candidate or at the end of its right-most terminal.

The third family of features is concerned with specific linguistic phenomena described in the BioScope annotation guidelines (Vincze et al. 2008) or observed when

developing the rules in Section 6.1. These include detection of passivization, subject control verbs occurring with passivized verbs, subject raising verbs, and predicative adjectives. Furthermore, these features are only activated when the subject of the construction is not an expletive pronoun, and they are represented by appending the type of phenomenon observed to the path features described here.

6.2.4 Ranker Optimization. We conducted several experiments designed to find an optimal configuration of features. Table 8 lists the results of combinations of the feature families on the BSP data set when using gold-standard cues, reporting 10-fold cross-validated F_1 scores with respect to only the instances where the gold-standard speculation scope aligns with constituents (i.e., the "ideal circumstances" for the ranker). The table also lists results for a random-choice baseline, calculated as the mean ambiguity of each instance (i.e., the averaged reciprocal of the number of candidates). The feature optimization results indicate that each feature family is informative, and that the best result can be obtained by using all three in conjunction. The comparatively largest improvement in ranker performance is obtained from the "rule-like" linguistic feature family, which is noteworthy in two respects: First, our current system includes only four such features, and second, these features parallel some of the dependency-based rules of Section 6.1.2—suggesting that subtle syntactic configurations are an important component also in our data-driven approach to scope resolution.

As discussed in Section 6.2.2 and depicted in Figure 4, searching the best-ranked parses can greatly increase the number of aligned constituents and thus improve the upper-bound potential of the ranker. We therefore experimented with training using the first aligned constituent in n-best derivations. At the same time we varied the m-best derivations used during testing, using features from all m derivations. We found that performance did not vary greatly, but that the best result was achieved when $n = 1$ and $m = 3$ (note, however, that such optimization over n-best lists of ERG parses will play a much greater role in the hybrid approach to scope resolution developed in Section 6.3). As explained in Section 5.2.1, all experiments use the SVM^{light} default value for the regularization parameter, determined analytically from the training data.

A cursory error analysis conducted over aligned items in BSP indicated similar errors to those discussed in connection with the dependency rules (see Section 6.1.5). There are a number of instances where the predicted scope is correct according to the BioScope annotation guidelines, but the annotated scope is incorrect. We also note some instances where the rule-like linguistic features are activated on the correct constituent, but the ranker nevertheless selects a different candidate. In a strictly rule-based system, these features would act as hard constraints and yield superior results in these cases. Therefore, these instances seem a prime source of inspiration for further improvements to the ranker in future work.

6.2.5 Evaluating the Ranker. Table 9 summarizes the performance of the constituent ranker (coupled with the default scope baseline in the case of unparsed items) compared with the dependency rules, resolving the scope of both gold-standard and predicted speculation cues. We note that the constituent ranker performs slightly superior to the dependency rules on BSA but inferior (though well above the default scope baseline) on BSP and BSE. Applying the sign-test (in the manner described in Section 3.2) to the scope-level performance of the ranker and the rules on the held-out BSE data (using gold-standard cues), the differences are found to be statistically significant.

Table 9
Resolving the scope of speculation cues using the dependency rules, the constituent ranker, and their combination. Whereas table (a) shows results for gold-standard cues, table (b) shows end-to-end results for the cues predicted by the classifier of Section 5.2. Results are shown both for the BioScope development data (for which both the scope ranker and the cue classifier is applied using 10-fold cross-validation) and the CoNLL-2010 Shared Task evaluation data.

Data	System	F_1
BSA	Rules	73.67
	Ranker	75.48
	Combined	**79.56**
BSP	Rules	72.31
	Ranker	66.17
	Combined	**75.15**
BSAP	Rules	73.40
	Ranker	73.61
	Combined	**78.69**
BSE	Rules	66.60
	Ranker	58.37
	Combined	**69.60**

(a) Resolving Gold-Standard Cues

Data	System	Prec	Rec	F_1
BSA	Rules	72.47	66.42	69.31
	Ranker	74.27	68.07	71.04
	Combined	77.80	71.31	**74.41**
BSP	Rules	69.87	62.13	65.77
	Ranker	62.63	55.69	58.95
	Combined	72.05	64.07	**67.83**
BSAP	Rules	71.97	65.56	68.61
	Ranker	71.99	65.59	68.64
	Combined	76.67	69.85	**73.11**
BSE	Rules	58.95	54.21	56.48
	Ranker	51.68	47.53	49.52
	Combined	62.00	57.02	**59.41**

(b) Resolving Predicted Cues

Again we also observe a drop in performance for the results on the held-out data compared with the development data. We attribute this drop partly to overfitting caused by using GENIA abstracts to adapt the parse ranker to the biomedical domain (see Section 2.3), but primarily to reduced parser coverage and constituent alignment in the latter data sets. Improving these aspects should result in substantive gains in ranker performance. Finally, note that the performance of the default baseline (which is much better for the abstracts than the full papers of BSP and BSE) also carries over to ranker performance for the cases where we do not have a parse.

6.3 Combining the Constituent Ranker and the Dependency Rules

Although both the constituent ranker and dependency rules perform well in isolation, they do not necessarily perform well on the same test items. Consequently, we investigated the effects of combining their predictions. When ERG parses are available for a given sentence, the dependency rules may be combined with the information used by the constituent ranker. We implement this coupling by adding features that record whether the (slackened) span of a candidate constituent matches the span of the scope predicted by the rules (either exactly or just at one of its boundaries). When an ERG parse is not available we simply revert to the prediction of the dependency rules.

Adding the rule prediction features may influence the effectiveness of considering multiple parses, by compensating for the extra ambiguity. We therefore repeated our examination of the effects of using the best-ranked parses for training and testing the ranker. Figure 5 plots the effect on F_1 for parsed sentences in BSP when including constituents from the n-best derivations in training, and from the m-best derivations in testing. We see that, when activating the dependency prediction features, the constituent ranker performs best for $n = 5$ and $m = 20$.

Looking at the performance summaries of Table 9, we see that the combined approach consistently outperforms both the dependency rules and the constituent ranker

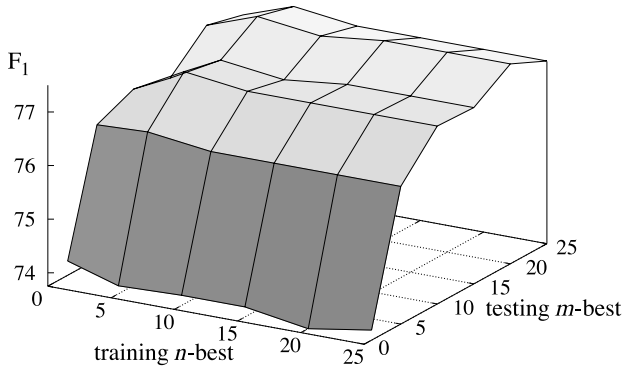

Figure 5
Cross-validated F_1 scores of the ranker combined with the dependency rules over gold cues for parsed sentences from BSP, varying the maximum number of parse results employed for training and testing.

in isolation, and the improvements are deemed significant with respect to both of them (comparing results for BSE using gold-standard cues).

Comparing the combined approach to the plain ranker runs, there are two sources for the improvements: the addition of the rule prediction features and the fact that we fall back on using the rule predictions directly (rather than the default scope) when we do not have an available constituent tree. To isolate the contribution of these factors, we applied the ranker without the rule-prediction features (as in the initial ranker set-up), but still using the rules as our fall-back strategy (as in the combined set-up). Testing on BSP using gold-standard cues this gives an F_1 of 69.61, meaning that the 8.98-percentage-point improvement of the combined model over the plain ranker owes 3.44 points to the rule-based fall-back strategy and 5.54 to the new rule-based features.

As discussed in Section 6.2.2, an important premise of the success of our ranking approach is that scopes align with constituents. Indeed, we find that the performance of both the ranker in isolation and the combined approach is superior on BSA, which is the data set that exhibits the greatest proportion of aligned instances. We can therefore expect that any improvements in our alignment procedure, as well as in the domain-adapted ERG parse selection model, will also carry through to improve the overall performance of our subtree ranking.

As a final evaluation of speculation resolution, Table 10 compares the end-to-end performance of our combined approach with the best end-to-end performer in the CoNLL-2010 Shared Task. In terms of both precision and recall, our cue classifier using the combination of constituent ranking and dependency rules for scope resolution achieves superior performance on BSE compared with the system of Morante, van Asch, and Daelemans (2010), improving on the overall F_1 by more than 2 percentage points. Whereas the token-level differences for cue classification are found to be significant, the end-to-end scope-level differences are not ($p = 0.39$).

7. Porting the Speculation System to Negation

Dealing with negation in natural language has been a long-standing topic and there has been work attempting to resolve the scope of negation in particular within the area of sentiment analysis (Moilanen and Pulman 2007), where treatment of negation clearly constitutes an important subtask and has been shown to provide improved sentiment

Table 10
Final end-to-end results for scope resolution: Held-out testing on BSE, using the cue classifier described in Section 5.2 while combining the dependency rules and the constituent ranker for scope resolution. The results are compared to the system with the best end-to-end performance in the CoNLL-2010 Shared Task (Morante, van Asch, and Daelemans 2010).

	Cue Level			Scope Level		
System Configuration	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**
Cue classifier + Scope Rules & Ranking	84.79	77.17	**80.80**	62.00	57.02	**59.41**
Morante et al. 2010	78.75	74.69	76.67	59.62	55.18	57.32

analysis (Councill, McDonald, and Velikovich 2010). The BioScope corpus (Vincze et al. 2008), being annotated with negation as well as speculation, has triggered work on negation detection in the biomedical domain as well. In this setting, there are a few previous studies where the same system architecture has been successfully applied for both speculation and negation. For example, whereas Morante and Daelemans (2009a) try to resolve the scope of speculation using a system initially developed for negation (Morante, Liekens, and Daelemans 2008), Zhu et al. (2010) develop a system targeting both tasks. In this section we investigate to what degree our speculation system can be ported to also deal with negation, hoping that the good results obtained for speculation will carry over to the negation task at a minimal cost in terms of adaption and modification. We start by describing our experiments with porting the cue classifier to negation in Section 7.1, and then present our modified set of dependency rules for resolving the scope of the negation cues in Section 7.2. Section 7.3 presents the adaptation of the constituent ranker, as well as the final end-to-end results when combining the ranker and the rules, paralleling what we did for speculation. The relation to other relevant work is discussed as we go along.

Some summarizing statistics for the negation annotations in BioScope were given in Table 1. Note, however, that the additional evaluation data that we used for *held-out testing* of the speculation system, does *not* contain negation annotations. For this reason, and in order to be able to compare our results to those obtained in previous studies, we here follow the partitioning established by Morante and Daelemans (2009b), reporting 10-fold cross-validation (for the cue classifier and the subtree ranker) on the abstracts (BSA) and using the full papers (BSP) for held-out and cross text-type testing. Note that for the development results using cross-validation, we partition the data on the sentence-level, just as in Morante and Daelemans (2009b).

7.1 Identifying Negation Cues

Several previous approaches to detecting negation cues have been based on pre-compiled lexicons, either alone (Councill, McDonald, and Velikovich 2010) or in combination with a learner (Morante and Daelemans 2009b). For the purpose of the current article we wanted to investigate whether the "filtered classification" approach that we applied for detecting speculation cues would directly carry over to negation. Drawing heavily on much of the discussion previously given for the speculation cue classifiers in Section 5, the small modifications made to implement a classifier for negation cues are described in Section 7.1.1. We then provide some discussion of the results in Section 7.1.2, including comparison to previous work on negation cue detection by Morante and Daelemans (2009b) and Zhu et al. (2010) in Section 7.1.3.

7.1.1 Classifier Description. Apart from re-tuning the feature configuration, the only modifications that we made with respect to the speculation classifier regard the rules for multiword cues (as described for speculation in Section 5.1) and the corresponding stop-list (Section 5.2). The overall approach, however, is the same: We train and apply a linear SVM classifier that only considers words whose lemma has been observed as a negation cue in the training data. Note that roughly 82% of the negation tokens are ambiguous in the training data, in the sense that they have both cue and non-cue occurrences. Based on the most frequently occurring MWC patterns observed in the abstracts we defined post-processing rules to cover the cases shown in Table 11. Furthermore, and again based on the MWCs, we compiled a small stop-list so that the classifier ignores certain "spurious" tokens (namely, *can*, *could*, *notable*, *of*, *than*, *the*, *with*, and '('). Although this of course means that the classifier will never label any such word as a cue, they will typically be captured by the MWC rules instead.

When re-tuning the feature configuration based on the *n*-gram templates previously described in Section 5.2.1, we find that the best performer for negation is the combination that records lemmas two positions to the left and the right of the target word, and surface forms one position to the right.

7.1.2 Development Results. The performance of this model, evaluated by 10-fold cross-validation on the BioScope abstracts, is shown in Table 12. Just as for speculation, we also contrast the performance with a simple WbW majority usage baseline, classifying each and every word according to its most frequent usage (cue vs. non-cue) in the training data. Although this baseline proved to be surprisingly strong for speculation, it is even stronger for negation: Evaluated at the token-level (though after the application of the MWC rules) the baseline achieves an F_1 of 93.60. Applying the filtering model further improves this score to 96.00. The differences are found to be statistically significant (according to the testing scheme described in Section 3.1), and the filtering classifier also improves greatly with respect to the sentence-, and cue-level evaluations as well, in particular with respect to the precision.

Recall that, when looking at the distribution of error types for the token-level mistakes made by the *speculation classifier* (see Section 5.2.3), we found that almost 75% were false negatives. The distribution of error types for the *negation cue classifier* is very different: Almost 85% of the errors are false positives. After inspecting the actual cues involved, we find the same situation as reported by Morante and Daelemans (2009b), namely, that a very high number of the errors concern cases where *not* is labeled as a cue by the classifier but not in the annotations. The same is true for the cue word *absence*, and many of these cases appear to be annotation errors.

The class balance among tokens in the BioScope data is extremely skewed, with the positive examples of negation constituting only 0.5% of the total number of examples.

Table 11
Patterns covered by our post-processing rules for multiword negation cues.

rather than
{can|could} not
no longer
instead of
with the * exception of
neither * nor
{no(t?)|neither} * nor

Table 12
Results for negation cue detection, including the systems of Morante et al. (2009b) and Zhu et al. (2010). Whereas the scores for BSA are obtained by 10-fold cross validation, the scores on BSP and BSR represent held-out testing using a model trained on all the abstracts. The latter scores thereby serves as a test of generalization performance across different text types within the same domain.

		Sentence Level			Token Level			Cue Level		
Data	**Model**	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**
BSA (10-Fold)	Baseline	90.34	98.81	94.37	89.28	98.40	93.60	88.92	97.78	93.14
	Filtering	94.19	98.87	**96.45**	93.46	98.73	**96.00**	93.19	98.12	**95.59**
	Morante	n/a	n/a	n/a	84.72	98.75	91.20	94.15	90.67	92.38
	Zhu	n/a	n/a	n/a	94.35	94.99	94.67	n/a	n/a	n/a
BSP (Held-out)	Baseline	79.48	99.41	88.34	75.96	99.00	85.96	74.55	98.41	84.84
	Filtering	86.75	98.53	**92.27**	85.22	98.25	**91.27**	84.06	97.62	**90.33**
	Morante	n/a	n/a	n/a	87.18	95.72	91.25	85.55	78.31	81.77
	Zhu	n/a	n/a	n/a	87.47	90.48	88.95	n/a	n/a	n/a
BSR (Held-out)	Baseline	96.64	96.42	96.53	96.12	96.01	96.06	95.87	95.98	95.93
	Filtering	96.97	96.30	**96.64**	96.44	95.90	96.17	96.20	95.87	**96.03**
	Morante	n/a	n/a	n/a	97.33	98.09	**97.71**	96.38	91.62	93.94
	Zhu	n/a	n/a	n/a	88.54	86.81	87.67	n/a	n/a	n/a

In terms of the tokens actually considered by our filtering model, however, the numbers look much healthier, with the negative examples actually being slightly outweighed by the positives (just above 50%). Moreover, the average number of distinct *n*-gram features instantiated across the 10-folds is approximately 17,500. The small size of the feature set is of course due to the small number of training examples considered by the learner: Whereas a WbW approach (like the majority usage baseline) would consider every token in training data (just below 300,000 in each fold), this number is reduced by almost 99% for the filtered disambiguation model. In effect, we can conclude that the proposed approach manages to combine very good results with very low computational cost.

Figure 6 shows learning curves for both the word-by-word baseline and the filtering model, plotting token-level F_1 against percentages of data included in training. Compared to the learning curves previously shown for speculation detection (Figure 1), the curve for the filtering model seems to be somewhat flatter for negation. Looking at the curve for the WbW unigram baseline, it again seems unable to benefit much from any additional data after the first few increments.

7.1.3 Comparison to Related Work. To the best of our knowledge, the systems currently achieving state-of-the-art results for detecting negation cues are those described by Morante and Daelemans (2009b), Zhu et al. (2010), and Councill, McDonald, and Velikovich (2010). Although the latter work does not offer separate evaluation of the cue detection scheme in isolation, Morante and Daelemans (2009b) and Zhu et al. (2010) provide cue evaluation for the data splits listed in Table 12; 10-fold cross-validation experiments (with sentence-level partitioning) on the BioScope abstracts, and held-out testing on the full papers and the clinical reports (with a model trained on the abstracts).

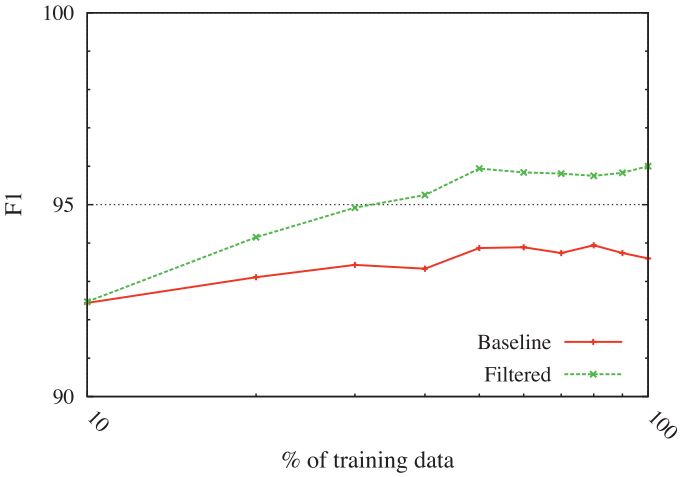

Figure 6
Learning curves for both baseline and the filtered "disambiguation" model showing the effect on token-level negation cue F_1 when including larger percentages (shown on a logarithmic scale) of the training data across the 10-fold cycles on BSA.

The results[5] reported by Morante and Daelemans (2009b) and Zhu et al. (2010) are token-level precision, recall, and F_1. Having obtained the system output of Morante and Daelemans (2009b), however, we also computed cue-level scores for their system.

Morante and Daelemans (2009b) identify cues using a small list of unambiguous cue words compiled from the abstracts in combination with applying a decision tree classifier to the remaining words. Their features record information about neighboring word forms, PoS, and chunk information from GENIA. Zhu et al. (2010) train an SVM to classify tokens according to a BIO-scheme using surface-oriented *n*-gram features in addition to various syntactic features extracted using the Berkley parser (Petrov and Klein 2007) trained on the GENIA treebank. Looking at the results in Table 12, we see that the performance of our cue classifier compares favorably with the systems of both Morante and Daelemans (2009b) and Zhu et al. (2010), achieving a higher cue-level F_1 across all data sets (with differences in classifier decisions with respect to Morante and Daelemans [2009b] being statistically significant for all of them).

For the 10-fold run, the biggest difference concerns token-level precision, where both the system of Zhu et al. (2010) and our own achieves a substantially higher score than that of Morante and Daelemans (2009b). Turning to the cross-text experiment, however, the precision of our system and that of Zhu et al. (2010) suffers a large drop, whereas the system of Morante and Daelemans (2009b) actually obtains a higher precision than for the 10-fold run. These effects are reversed for recall, however, where our system still maintains the higher score, also resulting in a higher F_1. Looking at the cue-level scores, we find that the precision of our system and that of Morante and Daelemans (2009b) drops by an equal amount for the BSP cross-text testing. In terms of recall, however, the cue-level scores of Morante and Daelemans (2009b) suffers a much larger drop than that of our filtered classifier.

5 As the results reported by Morante and Daelemans (2009b) were inaccurate, we instead refer to values obtained from personal communication with the authors.

The drop in performance when going from cross-validation to held-out testing can largely be attributed to the same factors discussed in relation to speculation cues in Section 5.3 (e.g., GENIA-based pre-processing, sentence-level partitioning in cross-validation, and unobserved MWCs). In addition, looking at the BioScope inter-annotator agreement rates for negation cues it is not surprising that we should observe a drop in results going from BSA to BSP: Measured as the F_1 of one of the annotators with respect to the other, it is reported as 91.46 for BSA, compared with 79.42 for BSP (Vincze et al. 2008). Turning to the F_1-scores of each annotator with respect to the final gold standard, the numbers are 91.71/98.05 for BSA and 86.77/91.71 for BSP.

The agreement rates for the clinical reports, on the other hand, are much closer to those of the abstracts (Vincze et al. 2008), and the held-out scores we observe on this data set are generally also much better, not the least for the simple majority usage baseline. In general the baseline again proves to be surprisingly competitive, most notably with respect to recall where it actually outperforms all the other systems for both the cross-text experiments. (Recall that the baseline scores also reflect the application of the MWC rules, though.)

7.2 Adapting the Dependency Rules for Resolving Negation Scope

There have been several previous studies on resolving the scope of negation based on the BioScope corpus. For example, Morante and Daelemans (2009b) present a meta-learning approach that combines the output from three learners—a memory-based model, an SVM classifier, and a CRF classifier—using lexical features, such as PoS and chunk tags. Councill, McDonald, and Velikovich (2010) use a CRF learner with features based on dependency parsing (e.g., detailing the PoS of the head and the dependency path to the negation cue).

The annotation of speculation and negation in BioScope was performed using a common set of principles. It therefore seems reasonable to assume our dependency-based scope resolution rules for speculation should be general enough to allow porting to negation with fairly limited efforts. On the other hand, negation is expressed linguistically using quite different syntactic structures from speculation, so it is clear that some modifications will be necessary as well.

As we recall, the dependency rules for speculation scope are triggered by the PoS of the cue. Several of the same parts-of-speech (verbs, adverbs) also express negation. As an initial experiment, therefore, we simply applied the speculation rules to negation unmodified. As before, taking default scope to start at the cue word and spanning to the end of the sentence provides us with a baseline system. We find that applying our speculation scope rules directly to the task of negation scope resolution offers a fair improvement over the baseline. For BSA and BSP, the default scope achieves F_1 scores of 52.24 and 31.12, respectively, and the speculation rules applied directly without modifications achieve 48.67 and 56.25.

In order to further improve on these results, we introduce a few new rules to account specifically for negation. The general rule machinery is identical to the speculation scope rules described in Section 6.1: The rules are triggered by the part of speech of the cue and operate over the dependency representations output by the stacked dependency parser described in Section 6.1.1. In developing the rules we consulted the BioScope guidelines (Vincze et al. 2008), as well as a descriptive study of negation in the BioScope corpus (Morante 2010).

Table 13
Additional dependency-based scope rules for negation, with information source (MaltParser or XLE), organized by PoS of the cue.

PoS	Description	Source
DT	Determiners scope over their head node and its descendants	M
NN	Nouns scope over their descendants	M
$\texttt{NN}_{none}$	*none* take scope over entire sentence if subject and otherwise over its descendants	M
VB	Verbs scope over their descendants	M
$\texttt{RB}_{vb}$	Adverbs with verbal head scope over the descendants of the lexical verb	M, X
$\texttt{RB}_{other}$	Adverbs scope over the descendants of the head	M, X

7.2.1 Rule Overview. The added rules are presented in Table 13 and are described in more detail subsequently, organized by the triggering PoS of the negation cue.

Determiners. Determiner cue words in BioScope are largely realized by the negative determiner *no*. These take scope over their nominal head and its descendants, as seen in Example (31):

(31) The finding that dexamethasone has {⟨no⟩ effect on TPA-induced activation of PKC} suggests [...]

Nouns. Nominal cues take scope over their descendants (i.e., the members of the noun phrase), as shown in Example (32).

(32) This unresponsiveness occurs because of a {⟨lack⟩ of expression of the beta-chain (accessory factor) of the IFN-gamma receptor}, while at the same time [...]

The negative pronoun *none* is tagged as a noun by our system, but deviates from regular nouns in their negation scope: If the pronoun is a subject, it scopes over the remaining sentence, as in Example (33), whereas in object function it simply scopes over the noun phrase (Morante 2010). These are therefore treated specifically by our system.

(33) Similarly, {⟨none⟩ of SCOPE's component algorithms outperformed the other ten programs on this data set by a statistically significant margin}.

Adverbs. Adverbs constitute the majority of negation cues and are largely realized by the lexical item *not*. Syntactically, however, adverbs are a heterogeneous category. They may modify a number of different head words and their scope will thus depend largely on properties of the head. For instance, when an adverb is a nominal modifier, as in Example (34), it has a narrow scope which includes only the head noun (34) and its possible conjuncts.

(34) This report directly demonstrates that OTF-2 but {⟨not⟩ OTF-1} regulates the DRA gene

Verbal adverbs scope over the clause headed by the verbal head. As shown by Figure 2, the parser's analysis of verbal chains has the consequence that preverbal arguments and modifiers, such as subjects and adverbs, are attached to the finite verb and postverbal

arguments and modifiers are attached to the lexical verb, in cases where there is an auxiliary. This rule thus locates the lexical verb (e.g., *affect* in Example [35]), in the dependency path from the auxiliary head verb and defines scope over the descendants of this verb. In cases where the lexical verb is passive, the subject is included in the scope of the adverb, as in Example (36).

(35) IL-1 did {⟨not⟩ affect the stability of the c-fos and c-jun transcripts}.

(36) {Levels of RNA coding for the receptor were ⟨not⟩ modulated by exposure to high levels of ligand}.

7.2.2 Evaluating the Negation Rules. The result of resolving the scope of gold-standard negation cues using the new set of dependency rules (i.e., the speculation rules extended with the negation specific rules of Table 13), are presented in Table 14, along with the performance of the default scope baseline. First of all, we note that the baseline scores provided by assigning default scope to all cues differ dramatically between the data sets, ranging from an F_1 of 52.24 for BSA, 31.12 for BSP, and 91.43 for BSR. In comparison, the performance of the rules is fairly stable across BSA and BSP, and for both data sets they improve substantially on the baseline (up by roughly 18.5 and 34.5 percentage points on BSA and BSP, respectively). On BSR, however, the default scope baseline is substantially stronger than for the other data sets, and even performs slightly better than the rules. Recall from Table 1 that the average sentence length in the clinical reports is substantially lower (7.7) than for the other data sets (average of 26), a property which will make the default scope much more likely to succeed.

In order to shed more light on the performance of the rules on BSR, a manual error analysis was performed, once again by two trained linguists working together. We found that out of the total of 74 errors, 30 (40.5%) were parse errors, 29 (39.2%) were rule errors, 8 (10.8%) were annotation errors, and 4 (5.4%) were undecided. Although it is usually the case that short sentences are easier to parse, the reports contain a substantial proportion of ungrammatical structures, such as missing subjects, dropped auxiliaries, and bare noun phrases, as in Example (37), which clearly lead to lower parse quality, resulting in 40% parse errors. There are also constructions, such as so-called run-on constructions, as in Example (38), for which there is simply no correct analysis available

Table 14
Scope resolution for gold-standard negation cues across the BioScope sub-corpora.

Data	Configuration	F_1
BSA	Default	52.24
	Dependency Rules	70.91
	Constituent Ranker	68.35
	Combined	74.35
BSP	Default	31.12
	Dependency Rules	65.69
	Constituent Ranker	60.90
	Combined	70.21
BSR	Default	91.43
	Dependency Rules	90.86
	Constituent Ranker	89.59
	Combined	90.74

within the dependency framework (which, for instance, requires that graphs should be connected). In addition, the annotations of the reports data contain some idiosyncrasies which the rules fail to reproduce. Twenty-four percent of the errors are found with the same cue, namely, the adjective *negative*. The rules make attributive adjectives scope over their nominal heads, whereas the BSR annotations define the scope to only cover the cue word itself; see Example (37). The annotation errors were very similar to the ones observed in the earlier error analysis of Section 6.1.5.

(37) |{⟨Negative⟩} chest radiograph|.

(38) |{⟨No⟩ focal pneumonia}, normal chest radiograph|.

7.3 Adapting the Constituent Ranker for Negation

Adapting the SVM-based discriminative constituent ranker of Section 6.2 to also predict the scope of negation is a straightforward procedure, requiring only minor modifications: Firstly, we developed a further slackening heuristic to ensure that predicted scope does not begin with an auxiliary. Secondly, we augmented the family of linguistic features to also record the presence of adverb cues with verbal heads (as specified by the dependency-based scope rules in Table 13). Finally, we repeated the parameter tuning for training with n-best and testing with m-best parses (as described in Section 6.2.4). Performing 10-fold cross-validation on BSA using gold-standard negation cues, we found that the optimal values for the ranker in isolation were $n = 10$ and $m = 1$. When paralleling the combined approach developed in Section 6.3 (adding the rule-predictions as a feature in the ranker while falling back on rule-predicted scope for cases where we do not have an ERG parse) the optimal values were found to be $n = 15$ and $m = 5$. Examining the coverage of the parser and the alignment of constituents with negation scope (considering the 50-best parses), we found that the upper-bound of the constituent ranker (disregarding any fall-back strategy) on the BSA development set is 79.4% (compared to 83.6% for speculation).

Table 14 lists the performance of both the constituent ranker in isolation and the combined approach when resolving the scope of gold-standard negation cues (reporting 10-fold cross-validation results for BSA, while using BSP and BSR for held-out testing). We see that the dependency rules perform consistently better than the constituent ranker, although the differences are not found to be statistically significant (the p-values for BSA, BSP, and BSR are 0.06, 0.11, and 0.25, respectively). The combined approach again outperforms the dependency rules on both BSA and BSP (and by a much larger margin than we observed for speculation), however, with the improvements on both data sets being significant. Just as we observed for the dependency rules in Section 7.2.2, neither the constituent ranker nor the combined approach are effective in BSR.

7.4 End-to-End Evaluation with Comparison to Related Work

We now turn to evaluating our end-to-end negation system with SVM-based cue classification and scope resolution using the combination of constituent ranking and dependency-based rules. To put the evaluation in perspective we also compare our results against the results of other state-of-the-art approaches to negation detection.

Comparison to previous work is complicated slightly by the fact that different data splits and evaluation measures have been used across various studies. A commonly reported measure in the literature on resolving negation scope is the **percentage of**

correct scopes (PCS) as used by Morante and Daelemans (2009b), and Councill, McDonald, and Velikovich (2010), among others. Councill, McDonald, and Velikovich (2010) define PCS as the number of correct spans divided by the number of true spans. It therefore corresponds roughly to the scope-level recall as reported in the current article. The PCS notion of a correct scope, however, is less strict than in our set-up (Section 3.2): Whereas we require an exact match of both the cue and the scope, Councill, McDonald, and Velikovich (2010) do not include the cue identification in their evaluation.

Moreover, whereas the work of both Morante and Daelemans (2009b) and Councill, McDonald, and Velikovich (2010) is based on the BioScope corpus, only Morante and Daelemans (2009b) follow the same set-up assumed in the current article. Councill, McDonald, and Velikovich (2010), on the other hand, evaluate by 5-fold cross-validation on the papers alone, reporting a PCS score of 53.7%. When running our negation cue classifier and constituent ranker (in the hybrid mode using the dependency features) by 5-fold cross-validation on the papers we achieve a scope-level recall of 68.62 (and an F_1 of 64.50).

Table 15 shows a comparison of our negation scope resolution system with that of Morante and Daelemans (2009b). Rather than using the PCS measure reported by Morante and Daelemans (2009b), we have re-scored the output of their system according to the CoNLL-2010 shared task scoring scheme, and it should therefore be kept in mind that the system of Morante and Daelemans (2009b) originally was optimized with respect to a slightly different metric.

For the cross-validated BSA experiments we find the results of the two systems to be fairly similar, although the F_1 achieved by our system is higher by more than 5 percentage points, mostly due to higher recall. For the cross-text experiments, the differences are much more pronounced, with the F_1 of our system being more than 22 points higher on BSP and more than 17 points higher on BSR. Again, the largest differences are to be found for recall—even though this is the score that most closely corresponds to the PCS metric used by Morante and Daelemans (2009b)—but as seen in Table 15 there are substantial differences in precision as well. The scope-level differences between the two systems are found to be statistically significant across all the three BioScope sub-corpora.

Table 15
End-to-end results for our negation system, using the SVM cue classifier and the combination of subtree ranking and dependency-based rules for scope resolution, comparing with Morante et al. (2009b).

		Scope Level		
Data	**Configuration**	**Prec**	**Rec**	**F_1**
BSA 10-Fold	Morante et al. (2009b)	66.31	65.27	65.79
	Cue classifier & Scope Rules + Ranking	69.30	72.89	**71.05**
BSP Held-out	Morante et al. (2009b)	42.49	39.10	40.72
	Cue classifier & Scope Rules + Ranking	58.58	68.09	**62.98**
BSR Held-out	Morante et al. (2009b)	74.03	70.54	72.25
	Cue classifier & Scope Rules + Ranking	89.62	89.41	**89.52**

To some degree, some of the differences are to be expected, perhaps, at least with respect to BSP. For example, the BSP evaluation represents a held-out setting for both the cue and scope component in the machine learned system of Morante and Daelemans (2009b). While also true for our cue classifier and subtree ranker, it is not strictly speaking the case for the dependency rules, and so the potential effect of any overfitting during learning might be less visible. The small set of manually defined rules are general in nature, targeting the general syntactic constructions expressing negation, as shown in Table 13. In addition to being based on the BioScope annotation guidelines, however, both the abstracts and the full papers were consulted for patterns, and the fact that rule development has included intermediate testing on BSP (although mostly during the development of the initial set of speculation rules from which the negation rules are derived) has likely made our system more tailored to the peculiarities of this data set. When comparing the errors made by our system to those of Morante and Daelemans (2009b), the most striking example of this is the inclusion of post-processing rules in our system for "backing off" from bracketed expressions (as discussed in Section 6.1). Although making little difference on the abstracts, this has a huge impact when evaluating the full papers, where bracketed expressions (citations, references to figures and tables, etc.) are much more common, and the system output of Morante and Daelemans (2009b) seems to suffer from the lack of such robustness measures. In relation to the clinical reports, one should bear in mind that, although our combined system outperforms that of Morante and Daelemans (2009b) by a large margin, this result would still be rivaled by simply using our default scope baseline, as is clear from Table 14.

The scope results of Zhu et al. (2010) are unfortunately not currently directly comparable to ours, due to differences in evaluation methodologies. Whereas we perform an *exact match* evaluation at the *scope-level*, as described in Section 3, Zhu et al. (2010) use a much less strict *token-level* evaluation even for their scopes in their end-to-end evaluation. Nevertheless, our results appear to be highly competitive, because even with the strict exact match criterion underlying our scope-level evaluation, our scores are actually still higher for both the papers and the reports. (Zhu et al. [2010] report an F_1 of 78.50 for the 10-fold runs on the abstracts, and 57.22 and 81.41 for held-out testing on the papers and reports, respectively.)

8. Conclusion

This article has explored several linguistically informed approaches to the problem of resolving the scope of speculation and negation within sentences. Our point of departure was the system developed by Velldal, Øvrelid, and Oepen (2010) for the CoNLL-2010 Shared Task challenge on resolving speculation in biomedical texts, where a binary maximum entropy cue classifier was used in combination with a small set of manually crafted scope resolution rules operating over dependency structures. In the current article we have introduced several major extensions and improvements to this initial system design.

First we presented a greatly simplified approach to cue identification using a linear SVM classifier. The classifier only considers features of the immediate lexical context of a target word, and it only aims to "disambiguate" words that have already been observed as speculation cues in the training data. The filtering imposed by this latter "closed class" assumption greatly reduces the size and complexity of the model while increasing classifier accuracy, yielding state-of-the-art performance on the CoNLL-2010 Shared Task evaluation data.

We then presented a novel approach to the problem of resolving the scopes of cues within a sentence. As an alternative to using the manually defined dependency rules of our initial system, we showed how an SVM-based discriminative ranking function can be learned for choosing subtrees from HPSG-based constituent structures. An underlying assumption of the ranking approach is that annotated scopes actually align with constituents, and we provided in-depth discussion and analysis of this issue.

Furthermore, while both the dependency rules and the constituent ranker achieve good performance on their own, we showed how even better results can be achieved by combining the two, as the errors they make are not always overlapping. The combined approach uses the dependency rules for all cases where we do not have an available HPSG parse, and for the cases where we do, the scope predicted by the rules is included as a feature in the constituent ranker model. Together with the reformulation of our cue classifier, this combined model for scope resolution obtains the best published results so far on the CoNLL-2010 Shared Task evaluation data (to the best of our knowledge).

Finally, we have showed how all components of our speculation system are easily ported to also handle the problem of resolving the scope of negation. With only modest modifications, the system obtains state-of-the-art results also on the negation task. The system outputs corresponding to the end-to-end experiments with our final model configurations, for both speculation and negation, are made available online (see footnote 2) together with the relevant evaluation software.

Acknowledgments

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 also want to thank Buzhou Tang (HIT Shenzhen Graduate School) and Roser Morante (University of Antwerp), together with their colleagues, for providing us with the raw XML output of their negation and speculation systems in order to enable system comparisons. Andrew MacKinlay (Melbourne University) and Dan Flickinger (Stanford University) were of invaluable help in adapting ERG parse selection to the biomedical domain. We thank our colleagues at the University of Oslo for their comments and support during our original participation in the 2010 CoNLL Shared Task, as well as more recently in preparing this manuscript. Large-scale experimentation and engineering was made possible though access to the TITAN high-performance computing facilities at the University of Oslo, and we are grateful to the Scientific Computation staff at UiO, as well as to the Norwegian Metacenter for Computational Science. Last but not least, we are indebted to the anonymous reviewers for their careful reading and insightful comments.

References

Brants, Thorsten. 2000. TnT. A statistical Part-of-Speech tagger. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 224–231, Seattle, WA.

Butt, Miriam, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of the COLING Workshop on Grammar Engineering and Evaluation*, pages 1–7, Taipei.

Callmeier, Ulrich. 2002. Preprocessing and encoding techniques in PET. In Stephan Oepen, Daniel Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering. A Case Study in Efficient Grammar-based Processing*. CSLI Publications, Stanford, CA, pages 127–143.

Collier, Nigel, Hyun S. Park, Norihiro Ogata, Yuka Tateishi, Chikashi Nobata, Tomoko Ohta, Tateshi Sekimizu, Hisao Imai, Katsutoshi Ibushi, and Jun I. Tsujii. 1999. The GENIA project: Corpus-based knowledge acquisition and information extraction from genome research papers. In *Proceedings of the 9th Conference of the European Chapter of the ACL*, pages 271–272, Bergen.

Councill, Isaac G., Ryan McDonald, and Leonid Velikovich. 2010. What's great and what's not: Learning to classify the scope of negation for improved sentiment

analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*, pages 51–59, Uppsala.
Crouch, Dick, Mary Dalrymple, Ron Kaplan, Tracy King, John Maxwell, and Paula Newman. 2008. XLE documentation. Palo Alto Research Center, Palo Alto, CA.
Farkas, Richard, Veronika Vincze, Gyorgy Mora, Janos Csirik, and György Szarvas. 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*, pages 1–12, Uppsala.
Flickinger, Dan. 2002. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering: A Case Study in Efficient Grammar-based Processing*. CSLI Publications, Stanford, CA, pages 1–17.
Gildea, Daniel. 2001. Corpus variation and parser performance. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 167–202, Pittsburgh, PA.
Joachims, Thorsten. 1999. Making large-scale SVM learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT Press, Cambridge, MA, pages 41–56.
Joachims, Thorsten. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, Alberta.
Johansson, Richard and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proceedings of 16th Nordic Conference of Computational Linguistics*, pages 105–112, Tartu.
Johnson, Mark, Stuart Geman, Stephen Canon, Zhiyi Chi, and Stefan Riezler. 1999. Estimators for stochastic unification-based grammars. In *Proceedings of the 37th Meeting of the Association for Computational Linguistics*, pages 535–541, College Park, MD.
Kilicoglu, Halil and Sabine Bergler. 2010. A high-precision approach to detecting hedges and their scopes. In *Proceedings of the 14th Conference on Natural Language Learning*, pages 70–77, Uppsala.
Light, Marc, Xin Ying Qiu, and Padmini Srinivasan. 2004. The language of bioscience: Facts, speculations, and statements in between. In *Proceedings of the HLT-NAACL 2004 Workshop: Biolink 2004, Linking Biological Literature, Ontologies and Databases*, pages 17–24, Boston, MA.
MacKinlay, Andrew, Rebecca Dridan, Dan Flickinger, Stephan Oepen, and Timothy Baldwin. 2011. Treeblazing: Using external treebanks to filter parse forests for parse selection and treebanking. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 246–254, Chiang Mai.
Malouf, Robert and Gertjan van Noord. 2004. Wide coverage parsing with stochastic attribute value grammars. In *Proceedings of the IJCNLP Workshop Beyond Shallow Analysis*, Hainan.
Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English. The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
Martins, Andre F. T., Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Waikiki, HI.
Medlock, Ben and Ted Briscoe. 2007. Weakly supervised learning for hedge classification in scientific literature. In *Proceedings of the 45th Meeting of the Association for Computational Linguistics*, pages 992–999, Prague.
Moilanen, Karo and Stephen Pulman. 2007. Sentiment composition. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 378–382, Borovets.
Morante, Roser. 2010. Descriptive analysis of negation cues in biomedical texts. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, pages 1429–1436, Valletta.
Morante, Roser and Walter Daelemans. 2009a. Learning the scope of hedge cues in biomedical texts. In *Proceedings of the BioNLP 2009 Workshop*, pages 28–36, Boulder, CO.
Morante, Roser and Walter Daelemans. 2009b. A metalearning approach to processing the scope of negation. In *Proceedings of the 13th Conference on Natural Language Learning*, pages 21–29, Boulder, CO.
Morante, Roser, Anthony Liekens, and Walter Daelemans. 2008. Learning the scope of negation in biomedical texts.

In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 715–724, Waikiki, HI.

Morante, Roser, Vincent van Asch, and Walter Daelemans. 2010. Memory-based resolution of in-sentence scope of hedge cues. In *Proceedings of the 14th Conference on Natural Language Learning*, pages 40–47, Uppsala.

Nivre, Joakim, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 2216–2219, Genoa.

Nivre, Joakim and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of the 46th Meeting of the Association for Computational Linguistics*, pages 950–958, Columbus, OH.

Øvrelid, Lilja, Jonas Kuhn, and Kathrin Spreyer. 2009. Cross-framework parser stacking for data-driven dependency parsing. *TAL special issue on Machine Learning for NLP*, 50(3):109–138.

Øvrelid, Lilja, Erik Velldal, and Stephan Oepen. 2010. Syntactic scope resolution in uncertainty analysis. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1379–1387, Beijing.

Petrov, Slav and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 404–411, Rochester, NY.

Pollard, Carl and Ivan A. Sag. 1987. *Information-Based Syntax and Semantics. Vol. 1: Fundamentals*. CSLI Lecture Notes # 13. CSLI Press, Stanford, CA.

Pollard, Carl and Ivan A. Sag. 1994. *Head-driven Phrase Structure Grammar*. The University of Chicago Press and CSLI Publications, Chicago, IL.

Rei, Marek and Ted Briscoe. 2010. Combining manual rules and supervised learning for hedge cue and scope detection. In *Proceedings of the 14th Conference on Natural Language Learning*, pages 56–63, Uppsala.

Riezler, Stefan, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, and Mark Johnson. 2002. Parsing the Wall Street Journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics*, pages 271–278, Philadelphia, PA.

Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, pages 44–49, Manchester.

Szarvas, György. 2008. Hedge classification in biomedical texts with a weakly supervised selection of keywords. In *Proceedings of the 46th Meeting of the Association for Computational Linguistics*, pages 281–289, Columbus, OH.

Tang, Buzhou, Xiaolong Wang, Xuan Wang, Bo Yuan, and Shixi Fan. 2010. A cascade method for detecting hedges and their scope in natural language text. In *Proceedings of the 14th Conference on Natural Language Learning*, pages 13–17, Uppsala.

Toutanova, Kristina, Christopher D. Manning, Dan Flickinger, and Stephan Oepen. 2005. Stochastic HPSG parse disambiguation using the Redwoods corpus. *Research on Language and Computation*, 3(1):83–105.

Tsuruoka, Yoshimasa, Yuka Tateishi, Jin-Dong Kim, Tomoko Ohta, John McNaught, Sophia Ananiadou, and Jun'ichi Tsujii. 2005. Developing a robust Part-of-Speech tagger for biomedical text. In P. Bozanis and E. Houstis, editors, *Advances in Informatics*. Springer, Berlin, pages 382–392.

Velldal, Erik. 2011. Predicting speculation: A simple disambiguation approach to hedge detection in biomedical literature. *Journal of Biomedical Semantics*, 2(Suppl 5):S7.

Velldal, Erik, Lilja Øvrelid, and Stephan Oepen. 2010. Resolving speculation: MaxEnt cue classification and dependency-based scope rules. In *Proceedings of the 14th Conference on Natural Language Learning*, pages 48–55, Uppsala.

Vincze, Veronika, György Szarvas, Richárd Farkas, György Móra, and János Csirik. 2008. The BioScope corpus: Biomedical texts annotated for uncertainty, negation and their scopes. *BMC Bioinformatics*, 9 (Suppl. 11).

Vlachos, Andreas and Mark Craven. 2010. Detecting speculative language using syntactic dependencies and logistic regression. In *Proceedings of the 14th*

Conference on Natural Language Learning, pages 18–25, Uppsala.

Zhang, Yi, Stephan Oepen, and John Carroll. 2007. Efficiency in unification-based n-best parsing. In *Proceedings of the 10th International Conference on Parsing Technologies*, pages 48–59, Prague.

Zhang, Yi and Rui Wang. 2009. Cross-domain dependency parsing using a deep linguistic grammar. In *Proceedings of the 47th Meeting of the Association for Computational Linguistics*, pages 378–386, Singapore.

Zhang, Yue and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Meeting of the Association for Computational Linguistics*, pages 188–193, Portland, OR.

Zhu, Qiaoming, Junhui Li, Hongling Wang, and Guodong Zhou. 2010. A unified framework for scope learning via simplified shallow semantic parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 714–724, Cambridge, MA.