Detecting Speculative Language Using Syntactic Dependencies and Logistic Regression

Andreas Vlachos and **Mark Craven**
Department of Biostatistics and Medical Informatics
University of Wisconsin-Madison
{vlachos,craven}@biostat.wisc.edu

Abstract

In this paper we describe our approach to the CoNLL-2010 shared task on detecting speculative language in biomedical text. We treat the detection of sentences containing uncertain information (Task1) as a token classification task since the existence or absence of *cues* determines the sentence label. We distinguish words that have speculative and non-speculative meaning by employing syntactic features as a proxy for their semantic content. In order to identify the *scope* of each cue (Task2), we learn a classifier that predicts whether each token of a sentence belongs to the scope of a given cue. The features in the classifier are based on the syntactic dependency path between the cue and the token. In both tasks, we use a Bayesian logistic regression classifier incorporating a sparsity-enforcing Laplace prior. Overall, the performance achieved is 85.21% F-score and 44.11% F-score in Task1 and Task2, respectively.

1 Introduction

The term *speculative language*, also known as *hedging*, refers to expressions of uncertainty over statements. Recognition of such statements is important for higher-level applications. For example, a multi-document summarization system can assign different weights to speculative and non-speculative statements when aggregating information on a particular issue.

The CoNLL-2010 shared task (Farkas et al., 2010) formulates speculative language detection as two subtasks. In the first subtask (Task1), systems need to determine whether a sentence contains uncertain information or not. In the second subtask (Task2), systems need to identify the hedge *cues* and their *scope* in the sentence. Table 1 provides an example from the training data.

The participants are provided with data from two domains: biomedical scientific literature (both abstracts and full articles) and Wikipedia. We choose to focus on the former. The training data for this domain are nine full articles and 1,273 abstracts from the BioScope corpus (Szarvas et al., 2008) and the test data are 15 full articles.

Our approach to speculative language detection relies on syntactic parsing and machine learning. We give a description of the techniques used in Sections 2 and 3. We treat the detection of sentences containing uncertain information (Task1) as a token classification task in which we learn a classifier to predict whether a token is a *cue* or not. In order to handle words that have speculative and non-speculative meaning (e.g. "indicating" in the example of Table 1), we employ syntactic features as a proxy for their semantic content (Section 4). For *scope* identification (Task2), we learn a classifier that predicts whether each token of the sentence belongs to the scope of a particular cue (Section 6). The features used are based on the syntactic dependency path between the cue and the token. We report results and perform error analysis for both tasks, pointing out annotation issues that could be ameliorated (Sections 5 and 7). Based on our experience we suggest improvements on the task definition taking into account work from the broader field (Section 8).

2 Syntactic parsing for the biomedical domain

The syntactic parser we chose for our experiments is the C&C Combinatory Categorial Grammar (CCG) parser adapted to the biomedical domain (Rimell and Clark, 2009). In this framework, parsing is performed in three stages: part-of-speech (PoS) tagging, CCG supertagging and parse selection. The parse selection module de-

The Orthology and Combined modules both have states that achieve likelihood ratios above 400 (as high as 1207 for the Orthology module and 613 for the Combined module), {**indicating that** both these modules {**can**, on their own, predict some interacting protein pairs with a posterior odds ratio above 1}}.

Table 1: Sentence annotated as speculative with two cues (in boldface) and their scopes (in brackets).

rives the actual parse tree using the information from the other two components. The intermediate CCG supertagging stage assigns each token to a lexical category which attempts to capture its syntactic role in the sentence. Lexical categories contain more information than PoS tags (mainly on subcategorization) and they are more numerous, thereby making their assignment a relatively difficult task. Therefore, the parse selection module takes into account multiple predictions per token which allows recovery from supertagging errors while still reducing the ambiguity propagated. An interesting aspect of this three-stage parsing approach is that, if the parse selection module fails to construct a parse tree for the sentence (a common issue when syntactic parsers are ported to new domains), the lexical categories obtained by the supertagger preserve some of the syntactic information that would not be found in PoS tags.

The adaptation to the biomedical domain by Rimell and Clark (2009) involved re-training the PoS tagger and the CCG supertagger using in-domain resources, while the parse selection component was left intact. As recent work in the BioNLP 2009 shared task has shown (Kim et al., 2009), domain-adapted parsing benefits information extraction systems.

The native output of the C&C parser is converted into the Stanford Dependency (SD) collapsed dependency format (de Marneffe and Manning, 2008). These dependencies define binary relations between tokens and the labels of these relations are obtained from a hierarchy. While the conversion is unlikely to be perfect given that the native C&C output follows a different formalism, we made this choice because it allows for the use of different parsers with minimal adaptation.

Finally, an important pre-processing step we take is tokenization of the original text. Since the PoS tagger is trained on the GENIA corpus which follows the Penn TreeBank tokenization scheme, we use the tokenization script provided by the treebank.[1]

[1]http://www.cis.upenn.edu/~treebank/tokenization.html

3 Bayesian logistic regression

In both tasks, we use a Bayesian logistic regression classifier incorporating a sparsity-enforcing Laplace prior (Genkin et al., 2006). Logistic regression models are of the form:

$$p(y = +1|\beta, x) = \frac{exp(x\beta^T)}{1 + exp(x\beta^T)} \qquad (1)$$

where $y \in \{+1, -1\}$ is a binary class label, x is the feature vector representation of the instance to be classified and β is the feature weight vector which is learnt from the training data. Since feature interactions are not directly represented, the interactions that are expected to matter for the task considered must be specified as additional features. In Bayesian logistic regression, a prior distribution on β is used which encodes our prior beliefs on the feature weights. In this work, we use the Laplace prior which encourages the feature weight vector to be sparse, reflecting our belief that most features will be irrelevant to the task.

4 Detecting sentences containing speculation

In Task1, systems need to determine whether a sentence contains uncertain information (labeled *uncertain*) or not (labeled *certain*). A sentence is *uncertain* if one or more of its tokens denote uncertainty. Such tokens are labeled as *cues* and they are provided by the organizers for training. If a cue is a present, any other (potentially "unhedging") token becomes irrelevant to the task. Therefore, we cast the task as a binary token classification problem and determine the sentence label from the token-level decisions.

Words used as speculative cues do not always denote speculation. For example, in BioScope "if" and "appear" are annotated as cues 4% and 83% of the times they are encountered. In order to gain better understanding of the task, we build a dictionary-based cue extractor. First we extract all the cues from the training data and use their lemmas, obtained using *morpha* (Minnen et al., 2001), to tag tokens in the test data. We keep only single-token cues in order to avoid non-indicative lem-

token=indicating
PoS=VBG
	lemma=indicate
	lemma+PoS=indicate+VBG

CCG=(S[ng]\NP)/S[em] lemma+CCG=indicate+(S[ng]\NP)/S[em]	

Table 2: Features extracted for the token "indicating" from the Example in Table 1. CCG supertag (S[ng]\NP)/S[em] denotes that "indicating" expects an embedded clause (S[em]) to its right (indicated by the forward slash /) and a noun phrase (NP) to its left (indicated by the backward slash \) to form a present participle (S[ng]).

mas entering the dictionary (e.g. "that" in "indicate that"). Since the test data consist of full articles only, we evaluate the performance of the dictionary-based approach using four-fold cross-validation on the nine full articles of the training data with the abstracts added as training data in every fold, but not used as test data. The recall achieved is 98.07%, but F-score is lower (59.53%) demonstrating that the single-token cues in the training data provide adequate coverage, but low precision. The restricted domain helps precision as it precludes some word meanings from appearing. For example "might" is unlikely to be encountered as a noun in the biomedical domain. Nevertheless, in order to achieve better performance it is important to further refine the cue identification procedure.

Determining whether a token is used as a speculative cue or not resembles supervised word sense disambiguation. The main difference is that instead of having an inventory of senses for each word, we have two senses applicable to all words. As in most word sense disambiguation tasks, the classification of a word as cue or not is dependent on the other words in the sentence, which we take into account using syntax. The syntactic context of words is a useful proxy to their semantics, as shown in recent work on verb sense disambiguation (Chen and Palmer, 2009). Furthermore, it is easy to obtain syntactic information automatically using a parser, even though there will be some noise due to parsing errors. Similar intuitions were exploited by Kilicoglu and Bergler (2008) in refining a dictionary of cues with syntactic rules.

In what follows, we present the features extracted for each token for our final system, along with an example of their application in Table 2. Where appropriate we give the relevant labels in the Stanford Dependency (SD) scheme in parentheses for reproducibility:

- We extract the token itself and its lemma as features.
- To handle cases where word senses are identifiable by the PoS of a token ("might result" vs "the might"), we combine the latter with the lemma and add it as a feature.
- We combine the lemma with the CCG supertag and add it as a feature in order to capture cases where the hedging function of a word is determined by its syntactic role in the sentence. For example, "indicating" in the example of Table 1 is followed by a clausal complement (a very reliable predictor of hedging function for epistemic verbs), which is captured by its CCG supertag. As explained in Section 2, this information can be recovered even in sentences where the parser fails to produce a parse.
- Passive voice is commonly employed to limit commitment to the statement made, therefore we add it as a feature combined with the lemma to verbs in that voice (*nsubjpass*).
- Modal auxiliaries are prime hedging devices but they are also among the most ambiguous. For example, "can" is annotated as a cue in 16% of its occurrences and it is the fifth most frequent cue in the full articles. To resolve this ambiguity, we add as features the lemma of the main verb the auxiliary is dependent on (*aux*) as well as the lemmas of any dependents of the main verb. Thus we can capture some stereotypical speculative expressions in scientific articles (e.g "could be due"), while avoiding false positives that are distinguished by the use of first person plural pronoun and/or reference to objective enabling conditions (Kilicoglu and Bergler, 2008).
- Speculation can be expressed via negation of a word expressing certainty (e.g. "we do not know"), therefore we add the lemma of the token prefixed with "not" (*neg*).
- In order to capture stereotypical hedging expressions such as "raises the question" and "on the assumption" we add as features the direct object of verbs combined with the lemma of their object (*dobj*) and the preposition for nouns in a prepositional relation (*prep_**).
- In order to capture the effect of adverbs on the hedging function of verbs (e.g. "theoretically

features	Recall	Precision	F-score
tokens, lemmas	75.92	81.07	78.41
+PoS, CCG	78.23	83.71	80.88
+syntax	81.00	81.31	81.15
+combs	79.58	84.98	82.19

Table 3: Performance of various feature sets on Task1 using cross-validation on full articles.

features	Recall	Precision	F-score
tokens, lemmas	79.19	80.43	79.81
+PoS, CCG	81.12	85.22	83.12
+syntax	83.43	84.57	84.00
+combs	85.16	85.99	85.58

Table 4: Performance of various feature sets on Task1 using cross-validation on full articles incorporating the abstracts as training data.

considered”) we add the lemma of the adverb as a feature to the verb (*advmod*).

- To distinguish the probabilistic/numerical sense from the hedging sense of adjectives such as “possible”, we add the lemma and the number of the noun they modify as features (*amod*), since plural number tends to be associated with the probabilistic/numerical sense (e.g. “all possible combinations”).

Finally, given that this stage is meant to identify cues in order to recover their scopes in Task2, we attempt to resolve multi-token cues in the training data into single-token ones. This agrees with the *minimal* strategy for marking cues stated in the corpus guidelines (Szarvas et al., 2008) and it simplifies scope detection. Therefore, during training multi-token cues are resolved to their syntactic head according to the dependency output, e.g. in Table 1 “indicate that” is restricted to “indicate” only. There were two cases in which this process failed; the cues being “cannot” (S3.167) and “not clear” (S3.269). We argue that the former is inconsistently annotated (the sentence reads “cannot be defined. . . ” and it would have been resolved to “defined”), while the latter is headed syntactically by the verb “be” which is preceding it.

5 Task1 results and error analysis

Initially we experiment using the full-articles part of the training data only divided in four folds. The reason for this choice is that the language of the abstracts is relatively restricted and phenomena that appear only in full papers could be obscured by the abstracts, especially since the latter consist of more sentences in total (11,871 vs. 2,670). Such phenomena include language related to figures and descriptions of probabilistic models.

Each row in Table 3 is produced by adding extra features to the feature set represented on the row directly above it. First we consider using only the tokens and their lemmas as features which amounts to a weighted dictionary but which achieves reasonable performance. The inclusion of PoS tags and CCG supertags improves performance, whereas syntactic context increases recall while decreasing precision slightly. This is due to the fact that logistic regression does not represent feature interactions and the effect of these features varies across words. For example, clausal complements affect epistemic verbs but not other words (“indicate” vs. “state” in the example of Table 1) and negation affects only words expressing certainty. In order to ameliorate this limitation we add the lexicalized features described in Section 4, for example the combination of the lemma with the negation syntactic dependency. These additional features improved precision from 81.31% to 84.98%.

Finally, we add the abstracts to the training data which improves recall but harms precision slightly (Table 4) when only tokens and lemmas are used as features. Nevertheless, we decided to keep them as they have a positive effect for all other feature representations.

A misinterpretation of the BioScope paper (Szarvas et al., 2008) led us to believe that five of the nine full articles in the training data were annotated using the guidelines of Medlock and Briscoe (2007). After the shared task, the organizers clarified to us that all the full articles were annotated using the BioScope guidelines. Due to our misinterpretation, we change our experimental setup to cross-validate on the four full articles annotated in BioScope only, considering the other five full articles and the abstracts only as training data. We keep this setup for the remainder of the paper.

We repeat the cross-validation experiments with the full feature set and this new experimental setup and report the results in Table 5. Using the same feature set, we experiment with the Gaussian prior instead of the sparsity-enforcing Laplace prior which results in decreased precision and F-score,

	Recall	Precision	F-score
cross-Laplace	80.33	84.21	82.23
cross-Gaussian	81.59	80.58	81.08
test	84.94	85.48	85.21

Table 5: Performance of the final system in Task1.

therefore confirming our intuition that most features extracted are irrelevant to the task and should have zero weight. Finally, we report our performance on the test data using the Laplace prior.

6 Detecting the scope of the hedges

In Task2, the systems need to identify speculative cues and their respective scopes. Since our system for Task1 identifies cues, our discussion of Task2 focuses on identifying the scope of a given cue. It is a non-trivial task, since scopes can be nested and can span over a large number of tokens of the sentence.

An initial approach explored was to associate each cue with the token representing the syntactic head of its scope and then to infer the scope using syntactic parsing. In order to achieve this, we resolved the (almost always multi-token) scopes to their syntactic heads and then built a classifier whose features are based on syntactic dependency paths. Multi-token scopes which were not headed syntactically by a single token (according to the parser) were discarded in order to obtain a cleaner dataset for training. This phenomenon occurs rather frequently, therefore reducing the training instances. At testing, the classifier identifies the syntactic head of the scope for each cue and we infer the scope from the syntactic parser's output. If more than one scope head is identified for a particular cue, then the scopes are concatenated.

The performance of this approach turned out to be very low, 10.34% in F-score. We identified two principal reasons for this. First, relying on the syntactic parser's output to infer the scope is unavoidably affected by parsing errors. Second, the scope annotation takes into account semantics instead of syntax. For example bibliographic references are excluded based on their semantics.

In order to handle these issues, we developed an approach that predicts whether each token of the sentence belongs to the scope of a given cue. The overall scope for that cue becomes the string enclosed by the left- and right-most tokens predicted to belong to the scope. The features used by the classifier to predict whether a token belongs to the scope of a particular cue are based on the shortest syntactic dependency path connecting them, which is found using Dijkstra's algorithm. If no such path is found (commonly due to parsing failure), then the token is classified as not belonging to the scope of that cue. The features we use are the following:

- The dependency path between the cue and the token, combined with both their lemmas.
- According to the guidelines, different cues have different preferences in having their scopes extended to their left or to their right. For example modal auxiliaries like "can" in Table 1 extend their scope to their right. Therefore we add the dependency path feature defined above refined by whether the token is on the left or the right of the cue in question.
- We combine the dependency path and the lemmas of the cue and the token with their PoS tags and CCG supertags, since these tags refine the syntactic function of the tokens.

The features defined above are very sparse, especially when longer dependency paths are involved. This can affect performance substantially, as the scopes can be rather long, in many cases spanning over the whole sentence. An unseen dependency path between a cue and a token during testing results in the token being excluded from the scope of that cue. In turn, this causes predicted scopes to be shorter than they should be. We attempt to alleviate this sparsity in two stages. First, we make the following simplifications to the labels of the dependencies:

- Adjectival, noun compound, relative clause and participial modifiers (*amod*, *nn*, *rcmod*, *partmod*) are converted to generic modifiers (*mod*).
- Passive auxiliary (*auxpass*) and copula (*cop*) relations are converted to auxiliary relations (*aux*).
- Clausal complement relations with internal/external subject (*ccomp/xcomp*) are converted to complement relations (*comp*).
- All subject relations in passive or active voice (*nsubj*, *nsubjpass*, *csubj*, *csubjpass*) are converted to subjects (*subj*).
- Direct and indirect object relations (*iobj*, *dobj*) are converted to objects (*obj*).

- We de-lexicalize conjunct (*conj_**) and prepositional modifier relations (*prep_**).

Second, we shorten the dependency paths:

- Since the SD collapsed dependencies format treats conjunctions asymmetrically (*conj*), we propagate the subject and object dependencies of the head of the conjunction to the dependent. We process appositional and abbreviation modifiers (*appos*, *abbrev*) in the same way.
- Determiner and predeterminer relations (*det*, *predet*) in the end of the dependency path are removed, since the determiners (e.g. "the") and predeterminers (e.g. "both") are included in/excluded from the scope following their syntactic heads.
- Consecutive modifier and dependent relations (*mod*, *dep*) are replaced by a single relation of the same type.
- Auxiliary relations (*aux*) that are not in the beginning or the end of the path are removed.

Despite these simplifications, it is still possible during testing to encounter dependency paths unseen in the training data. In order to ameliorate this issue, we implement a backoff strategy that progressively shortens the dependency path until it matches a dependency path seen in the training data. For example, if the path from a cue to a token is *subj-mod-mod* and it has not been seen in the training data, we test if *subj-mod* has been seen. If it has, we consider it as the dependency path to define the features described earlier. If not, we test for *subj* in the same way. This strategy relies on the assumption that tokens that are likely to be included in/excluded from the scope following the tokens they are syntactically dependent on. For example, modifiers are likely to follow the token being modified.

7 Task2 results and error analysis

In order to evaluate the performance of our approach, we performed four-fold cross-validation on the four BioScope full articles, using the remaining full articles and the abstracts as training data only. The performance achieved using the features mentioned in Section 6 is 28.62% F-score, while using the simplified dependency paths instead of the path extracted from the parser's output improves it to 34.35% F-score. Applying the back-off strategy for unseen dependency paths to the simplified paths results in 35.37% F-score (Table 6).

features	Recall	Precision	F-score
standard	27.54	29.79	28.62
simplified	33.11	35.69	34.35
+backoff	34.10	36.75	35.37
+post	40.98	44.17	42.52

Table 6: Performance on Task2 using cross-validation on BioScope full articles.

Our system predicts only single token cues. This agrees in spirit with the minimal cue annotation strategy stated in the BioScope guidelines. The guidelines allow for multi-token cues, referred to as *complex keywords*, which are defined as cases where the tokens of a phrase cannot express uncertainty independently. We argue that this definition is rather vague, and combined with the requirement for contiguity, results in cue instances such as "indicating that" (multiple occurrences), "looks as" (S4.232) and "address a number of questions" (S4.36) annotated as cues. It is unclear why "suggesting that" or "appears that" are not annotated as cues as well, or why "that" contributes to the semantic content of "indicate". "that" does help determine the sense of "indicate", but we argue that it should not be part of the cue as it does not contribute to its semantic content. "indicate that" is the only consistent multi-token cue pattern in the training data. Therefore, when our system identifies as a cue a token with the lemma "indicate", if this token is followed by "that", "that" is added to the cue. Given the annotation difficulties multi-token cues present, it would be useful during evaluation to relax cue matching in the same way as in the BioNLP 2009 shared task, i.e. considering as correct those cues predicted within one token of the gold standard annotation.

As explained in Section 6, bibliographic references are excluded from scopes and cannot be recognized by means of syntactic parsing only. Additionally, in some cases the XML formatting does not preserve the parentheses and/or brackets around numerical references. We employ two post-processing steps to deal with these issues. First, if the ultimate token of a scope happens to be the penultimate token of the sentence and a number, then it is removed from the scope. This step can have a negative effect when the last token of the scope and penultimate token of the sen-

		Recall	Precision	F-score
Cues	cross	74.52	81.63	77.91
	test	74.50	81.85	78.00
Task2	cross	40.98	44.17	42.52
	test	42.40	45.96	44.11

Table 7: Performance on cue identification and cue/scope identification in Task2.

tence happens to be a genuine number, as in Figure 1. In our experiments however, this heuristic always increased performance. Second, if a scope contains an opening parenthesis but not its closing one, then the scope is limited to the token immediately before the opening one. Note that the training data annotation allows for partial parenthetical statements to be included in scopes, as a result of terminating scopes at bibliographic references which are not the only tokens in a parentheses. For example, in S7.259: "expressed (ED, unpublished)" the scope is terminated after "ED". These post-processing steps improved the performance substantially to 42.52% F-score (Table 6).

The requirement for contiguous scope spans which include their cue(s) is not treated appropriately by our system, since we predict each token of the scope independently. Combined with the fact that the guidelines frequently define scopes to extend either to the left or to the right of the cue, an approach based on sequential tagging and/or predicting boundaries could perform better. However, as mentioned in the guidelines, the contiguity requirement sometimes forced the inclusion of tokens that should have been excluded given the purpose of the task.

Our final performance on the test data is 44.11% in F-score (Table 7). This is higher than the one reported in the official results (38.37%) because we subsequently increased the coverage of the C&C parser (parse failures resulted in 63 cues not receiving a scope), the addition of the back-off strategy for unseen dependency paths and the clarification on the inclusion of bibliographic references in the scopes which resulted in improving the parentheses post-processing steps.

8 Related work

The shared task uses only full articles for testing while both abstracts and full articles are used for training. We argue that this represents a realistic scenario for system developers since annotated resources consist mainly of abstracts, while most information extraction systems are applied to full articles. Also, the shared task aimed at detecting the scope of speculation, while most previous work (Light et al., 2004; Medlock and Briscoe, 2007; Kilicoglu and Bergler, 2008) considered only classification of sentences, possibly due to the lack of appropriately annotated resources.

The increasing interest in detecting speculative language in scientific text resulted in a number of guidelines. Compared to the most recent previous definition by Medlock and Briscoe (2007), BioScope differs in the following ways:

- BioScope does not annotate anaphoric hedge references.
- BioScope annotates indications of experimentally observed non-universal behaviour.
- BioScope annotates statements of explicitly proposed alternatives.

The first difference is due to the requirement that the scope of the speculation be annotated, which is not possible when it is present in a different sentence. The other two differences follow from the stated purpose which is the detection of sentences containing uncertain information.

In related work, Hyland (1996) associates the use of speculative language in scholarly publications with the purpose for which they are employed by the authors. In particular, he distinguishes content-oriented hedges from reader-oriented ones. The former are used to calibrate the strength of the claims made, while the latter are employed in order to soften anticipated criticism on behalf of the reader. Content-oriented hedges are further distinguished as accuracy-oriented ones, used to express uncertain claims more accurately, and writer-oriented ones, used to limit the commitment of the author(s) to the claims. While the boundaries between these distinctions are not clear-cut and instances of hedging can serve more than one of these purposes simultaneously, it is worth bearing them in mind while approaching the task. With respect to the shared task, taking into account that hedging is used to express statements more accurately can help resolve the ambiguity when annotating certain statements about uncertainty. Such statements, which involve words such as "estimate", "possible", "predict", occur frequently in full articles.

Wilson (2008) analyzes speculation detection inside a general framework for sentiment analysis centered around the notion of private states (emotions, thoughts, intentions, etc.) that are not open to objective observation or verification. Speculation is annotated with a *spec-span/spec-target* scheme by answering the questions *what the speculation is* and *what the speculation is about*. With respect to the BioScope guidelines, *spec-span* is similar to what *scope* attempts to capture. *spec-span* and *spec-target* do not need to be present at the same time, which could help annotating anaphoric cues.

9 Conclusions

This paper describes our approach to the CoNLL-2010 shared task on speculative language detection using logistic regression and syntactic dependencies. We achieved competitive performance on sentence level uncertainty classification (Task1), but not on scope identification (Task2). Motivated by our error analysis we suggest refinements to the task definition that could improve annotation.

Our approach to detecting speculation cues successfully employed syntax as a proxy for the semantic content of words. In addition, we demonstrated that performance gains can be obtained by choosing an appropriate prior for feature weights in logistic regression. Finally, our performance in scope detection was improved substantially by the simplification scheme used to reduce the sparsity of the dependency paths. It was devised using human judgment, but as information extraction systems become increasingly reliant on syntax and each task is likely to need a different scheme, future work should investigate how this could be achieved using machine learning.

Acknowledgements

We would like to thank the organizers for providing the infrastructure and the data for the shared task, Laura Rimell for her help with the C&C parser and Marina Terkourafi for useful discussions. The authors were funded by NIH/NLM grant R01/LM07050.

References

Jinying Chen and Martha Palmer. 2009. Improving English Verb Sense Disambiguation Performance with Linguistically Motivated Features and Clear Sense Distinction Boundaries. *Language Resources and Evaluation*, 43(2):143–172.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *CrossParser '08: Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, pages 1–8.

Richárd Farkas, Veronika Vincze, György Móra, János Csirik, and György Szarvas. 2010. The CoNLL-2010 Shared Task: Learning to Detect Hedges and their Scope in Natural Language Text. In *Proceedings of CoNLL-2010 Shared Task*, pages 1–12.

Alexander Genkin, David D. Lewis, and David Madigan. 2006. Large-scale Bayesian Logistic Regression for Text Classification. *Technometrics*, 49(3):291–304.

Ken Hyland. 1996. Writing Without Conviction? Hedging in Science Research Articles. *Applied Linguistics*, 17(4):433–454.

Halil Kilicoglu and Sabine Bergler. 2008. Recognizing speculative language in biomedical research articles: a linguistically motivated perspective. In *BioNLP '08: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 46–53.

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

Marc Light, Xin Ying Qiu, and Padmini Srinivasan. 2004. The Language of Bioscience: Facts, Speculations, and Statements In Between. In *HLT-NAACL 2004 Workshop: BioLINK 2004, Linking Biological Literature, Ontologies and Databases*, pages 17–24.

Ben Medlock and Ted Briscoe. 2007. Weakly Supervised Learning for Hedge Classification in Scientific Literature. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 992–999.

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

Laura Rimell and Stephen Clark. 2009. Porting a lexicalized-grammar parser to the biomedical domain. *Journal of Biomedical Informatics*, 42(5):852–865.

György Szarvas, Veronika Vincze, Richárd Farkas, and János Csirik. 2008. The BioScope corpus: annotation for negation, uncertainty and their scope in biomedical texts. In *BioNLP '08: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 38–45.

Theresa Ann Wilson. 2008. *Fine-grained Subjectivity and Sentiment Analysis: Recognizing the Intensity, Polarity, and Attitudes of Private States*. Ph.D. thesis, University of Pittsburgh.