UiO$_2$: Sequence-Labeling Negation Using Dependency Features

Emanuele Lapponi **Erik Velldal** **Lilja Øvrelid** **Jonathon Read**
University of Oslo, Department of Informatics
{emanuel,erikve,liljao,jread}@ifi.uio.no

Abstract

This paper describes the second of two systems submitted from the University of Oslo (UiO) to the 2012 *SEM Shared Task on resolving negation. The system combines SVM cue classification with CRF sequence labeling of events and scopes. Models for scopes and events are created using lexical and syntactic features, together with a fine-grained set of labels that capture the scopal behavior of certain tokens. Following labeling, negated tokens are assigned to their respective cues using simple post-processing heuristics. The system was ranked first in the open track and third in the closed track, and was one of the top performers in the scope resolution sub-task overall.

1 Introduction

Negation Resolution (NR) is the task of determining, for a given sentence, which tokens are affected by a negation cue. The data set most prominently used for the development of systems for automatic NR is the BioScope Corpus (Vincze et al., 2008), a collection of clinical reports and papers in the biomedical domain annotated with negation and speculation cues and their scopes. The data sets released in conjunction with the 2012 shared task on NR hosted by The First Joint Conference on Lexical and Computational Semantics (*SEM 2012) are comprised of the following negation annotated stories of Conan Doyle (CD): a training set of 3644 sentences drawn from *The Hound of the Baskervilles* (CDT), a development set of 787 sentences taken from *Wisteria Lodge* (CDD; we will refer to the combination of CDT and CDD as CDTD), and a held-out test set of 1089 sentences from *The Cardboard Box* and *The Red Circle* (CDE). In these sets, the concept of negation scope extends on the one adopted in the BioScope corpus in several aspects: Negation cues are not part of the scope, morphological (affixal) cues are annotated and scopes can be discontinuous. Moreover, in-scope states or events are marked as negated if they are factual and presented as events that did not happen (Morante and Daelemans, 2012). Examples (1) and (2) below are examples of affixal negation and discontinuous scope respectively: The cues are bold, the tokens contained within their scopes are underlined and the negated event is italicized.

(1) Since we have been so **un**_fortunate_ as to miss him [. . .]

(2) If he was in the hospital and yet **not** on the staff he could only have been a house-surgeon or a house-physician: little more than a senior student.

Example (2) has no negated events because the sentence is non-factual.

The *SEM shared task thus comprises three subtasks: cue identification, scope resolution and event detection. It is furthermore divided into two separate tracks: one closed track, where only the data supplied by the organizers (word form, lemma, PoS-tag and syntactic constituent for each token) may be employed, and an open track, where participants may employ any additional tools or resources.

Pragmatically speaking, a token can be either *out of scope* or assigned to one or more of the three remaining classes: *negation cue*, *in scope* and *negated event*. Additionally, *in-scope* tokens and *negated events* are paired to the cues they are negated by.

*First Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 319–327,
Montréal, Canada, June 7-8, 2012. ©2012 Association for Computational Linguistics

Our system achieves this by remodeling the task as a sequence labeling task. With annotations converted to sequences of labels, we train a Conditional Random Field (CRF) classifier with a range of different feature types, including features defined over dependency graphs. This article presents two submissions for the *SEM shared task, differing only with respect to how these dependency graphs were derived. For our open track submission, the dependency representations are produced by a state-of-the-art dependency parser, whereas the closed track submission employs dependencies derived from the constituent analyses supplied with the shared task data sets through a process of constituent-to-dependency conversion. In both systems, labeling of test data is performed in two stages. First, cues are detected using a token classifier,[1] and secondly, scope and event resolution is achieved by post-processing the output of the sequence labeler.

The two systems described in this paper have been developed using CDT for training and CDD for testing, and differ only with regard to the source of syntactic information. All reported scores are generated using an evaluation script provided by the task organizers. In addition to providing a full end-to-end evaluation, the script breaks down results with respect to identification of cues, events, scope tokens, and two variants of scope-level exact match; one requiring exact match also of cues and another only partial cue match. For our system these two scope-level scores are identical and so are not duplicated in our reporting. Additionally we chose not to optimize for the scope tokens measure, and hence this is also not reported as a development result.

Note also that the official evaluation actually includes two different variants of the metrics mentioned above; a set of *primary measures* with precision computed as P=TP/(TP+FP) and a set of *B measures* where precision is rather computed as P=TP/SYS, where SYS is the total number of predictions made by the system. The reason why SYS is not identical with TP+FP is that partial matches are only counted as FNs (and not FPs) in order to avoid double penalties. We do not report the B measures for development testing as they were introduced for the final evaluation and hence were not considered in our system optimization. We note though, that the relative-ranking of participating systems for the primary and B measures is identical, and that the correlation between the paired lists of scores is nearly perfect (r=0.997).

The rest of the paper is structured as follows. First, the cue classifier, its features and results are described in Section 2. Section 3 presents the system for scope and event resolution and details different features, the model-internal representation used for sequence-labeling, as well as the post-processing component. Error analyses for the cue, scope and event components are provided in the respective sections. Section 4 and 5 provide developmental and held-out results, respectively. Finally, we provide conclusions and some reflections regarding future work in Section 6.

2 Cue detection

Identification of negation cues is based on the light-weight classification scheme presented by Velldal et al. (2012). By treating the set of cue words as a closed class, Velldal et al. (2012) showed that one could greatly reduce the number of examples presented to the learner, and correspondingly the number of features, while at the same time improving performance. This means that the classifier only attempts to “disambiguate” known cue words while ignoring any words not observed as cues in the training data.

The classifier applied in the current submission is extended to also handle affixal negation cues, such as the prefix cue in ***im**patience*, the infix in *care**less**ness*, and the suffix of *colour**less***. The types of negation affixes observed in CDTD are; the prefixes *un*, *dis*, *ir*, *im*, and *in*; the infix *less* (we internally treat this as the suffixes *lessly* and *lessness*); and the suffix *less*. Of the total number of 1157 cues in the training and development set, 192 are affixal. There are, however, a total of 1127 tokens matching one of the affix patterns above, and while we maintain the closed class assumption also for the affixes, the classifier will need to consider its status as a cue

[1] Note that the cue classifier applied in the current paper is the same as that used in the other shared task submission from the University of Oslo (Read et al., 2012), and the two system descriptions will therefore have much overlap on this particular point. For all other components the architectures of the two system are completely different, however.

or non-cue when attaching to any such token, like for instance *image*, *recklessness*, and *bless*.

2.1 Features

In the initial formulation of Velldal (2011), an SVM classifier was trained using simple n-gram features over words, both full forms and lemmas, to the left and right of the candidate cues. In addition to these token-level features, the classifier we apply here includes some features specifically targeting morphological or affixal cues. The first such feature records character n-grams from both the beginning and end of the base that an affix attaches to (up to five positions). For a context like ***im**possible* we would record n-grams such $\{$*possi*, *poss*, *pos*, $\ldots\}$ and $\{$*sible*, *ible*, *ble*, $\ldots\}$, and combine this with information about the affix itself (*im*) and the token part-of-speech ("JJ").

For the second feature type targeting affix cues we try to emulate the effect of a lexicon look-up of the remaining substring that an affix attaches to, checking its status as an independent base form and its part-of-speech. In order to take advantage of such information while staying within the confines of the closed track, we automatically generate a lexicon from the training data, counting the instances of each PoS tagged lemma in addition to n-grams of word-initial characters (again recording up to five positions). For a given match of an affix pattern, a feature will then record the counts from this lexicon for the substring it attaches to. The rationale for this feature is that the occurrence of a substring such as *un* in a token such as *underlying* should be considered more unlikely to be a cue given that the first part of the remaining string (e.g., *derly*) would be an unlikely way to begin a word.

Note that, it is also possible for a negation cue to span multiple tokens, such as the (discontinuous) pair *neither / nor* or fixed expressions like *on the contrary*. There are, however, only 16 instances of such multiword cues (MWCs) in the entire CDTD. Rather than letting the classifier be sensitive to these corner cases, we cover such MWC patterns using a small set of simple post-processing heuristics. A small stop-list is used for filtering out the relevant words from the examples presented to the classifier (*on*, *the*, etc.).

Data set	**Model**	**Prec**	**Prec**	**F$_1$**
CDD	Baseline	90.68	84.39	87.42
	Classifier	93.75	95.38	94.56
CDE	Baseline	87.10	92.05	89.51
	Classifier	89.17	93.56	91.31

Table 1: Cue classification results for the final classifier and the majority-usage baseline, showing test scores for the development set (training on CDT) and the final held-out set (training on CDTD).

2.2 Results

Table 1 presents results for the cue classifier. While the classifier configuration was optimized against CDD, the model used for the final held-out testing is trained on the entire CDTD, which (given our closed-class treatment of cues) provides a total of 1162 positive and 1100 negative training examples. As an informed baseline, we also tried classifying each word based on its most frequent use as cue or non-cue in the training data. (Affixal cue occurrences are counted by looking at both the affix-pattern and the base it attaches to, basically treating the entire token as a cue. Tokens that end up being classified as cues are then matched against the affix patterns observed during training in order to correctly delimit the annotation of the cue.) This simple majority-usage approach actually provides a fairly strong baseline, yielding an F$_1$ of 87.42 on CDD (P=90.68, R=84.39). Compare this to the F$_1$ of 94.56 obtained by the classifier on the same data set (P=93.75, R=95.38). However, when applying the models to the held-out set, with models estimated over the entire CDTD, the baseline seems to able to make good use of the additional data and proves to be even more competitive: While our final cue classifier achieves F$_1$=91.31, the baseline achieves F$_1$=89.51, almost two percentage points higher than its score on the development data, and even outperforms four of the ten cue detection systems submitted for the shared task (three of the 12 shared task submissions use the same classifier).

When inspecting the predictions of our final cue classifier on CDD, comprising a total of 173 gold annotated cues, we find that our system mislabels 11 false positives (FPs) and 7 false negatives (FNs).

Of the FPs, we find five so-called *false negation cues* (Morante et al., 2011), including three instances of *none* in the fixed expression *none the less*. The others are affixal cues, of which two are clearly wrong (*__un__derworked*, *__un__iversal*) while others might arguably be due to annotation errors (*__in__superable*, *__un__happily*, *end__less__*, *list__less__ly*). Among the FNs, two are due to MWCs not covered by our heuristics (e.g., *no more*), while the remaining errors concern affixes, including one in an interesting context of double negation; ***not*** *__dis__satisfied*.

3 Scope and event resolution

In this work, we model negation scope resolution as a special instance of the classical IOB (Inside, Outside, Begin) sequence labeling problem, where negation cues are labeled to be sequence starters and scopes and events as two different kinds of chunks. CRFs allow the computation of $p(\mathbf{X}|\mathbf{Y})$, where $\mathbf{X}$ is a sequence of labels and $\mathbf{Y}$ is a sequence of observations, and have already been shown to be efficient in similar, albeit less involved, tasks of negation scope resolution (Morante and Daelemans, 2009; Councill et al., 2010). We employ the CRF implementation in the Wapiti toolkit, using default settings (Lavergne et al., 2010). A number of features were used to create the models. In addition to the information provided for each token in the CD corpus (lemma, part of speech and constituent), we extracted both left and right token distance to the closest negation cue. Features were expanded to include forward and backward bigrams and trigrams on both token and PoS level, as well as lexicalized PoS unigrams and bigrams[2]. Table 2 presents a complete list of features. The more intricate, dependency-based features are presented in Section 3.1, while the labeling of both scopes and events is detailed in Section 3.2.

3.1 Dependency-based features

For the system submitted to the closed track, the syntactic representations were converted to dependency representations using the Stanford dependency converter, which comes with the Stanford parser (de Marneffe et al., 2006).[3] These dependency representations result from a conversion of Penn Treebank-style phrase structure trees, combining 'classic' head finding rules with rules that target specific linguistic constructions, such as passives or attributive adjectives. The so-called *basic* format provides a dependency graph which is a directed tree, see Figure 1 for an example.

General features
Token
Lemma
PoS unigram
Forward token bigram and trigram
Backward token bigram and trigram
Forward PoS trigram
Backward PoS trigram
Lexicalized PoS
Forward Lexicalized PoS bigram
Backward Lexicalized PoS bigram
Constituent
Dependency relation
First order head PoS
Second order head PoS
Lexicalized dependency relation
PoS-disambiguated dependency relation
Cue-dependent features
Token distance
Directed dependency distance
Bidirectional dependency distance
Dependency path
Lexicalized dependency path

Table 2: List of features used to train the CRF models.

For the open track submission we used Maltparser (Nivre et al., 2006) with its pre-trained parse model for English.[4] The parse model has been trained on a conversion of sections 2-21 of the Wall Street Journal section of the Penn Treebank to Stanford dependencies, augmented with data from Question Bank. The parser was applied to the negation data, using the word tokens and supplied parts-of-speech as input to the parser.

The features extracted via the dependency graphs aim at modeling the syntactic relationship between each token and the closest negation cue. Token distance was therefore complemented with two variants of dependency distance from each token to the lexi-

[2] By *lexicalized PoS* we mean an instance of a PoS-Tag in conjunction with the sentence token.

[3] Note that the converter was applied directly to the phrase-structure trees supplied with the negation data sets, and the Stanford parser was not used to parse the data.

[4] The pre-trained model is available from maltparser.org

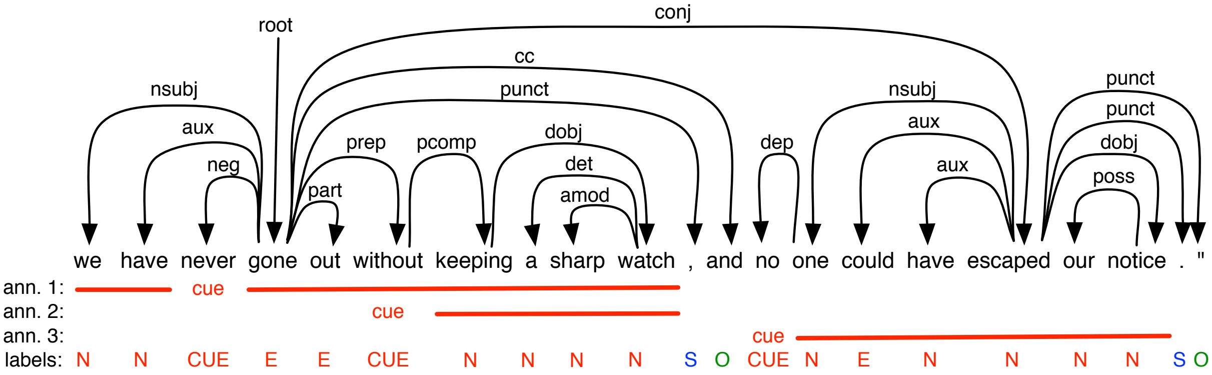
conj
root
cc
nsubj
punct
nsubj
punct
aux
punct
prep
pcomp
dobj
dep
aux
dobj
neg
det
aux
poss
part
amod
we have never gone out without keeping a sharp watch , and no one could have escaped our notice . "
ann. 1:
cue
ann. 2:
cue
ann. 3:
cue
labels: N N CUE E E CUE N N N N S O CUE N E N N N N S O

Figure 1: A sentence from the CD corpus showing a dependency graph and the annotation-to-label conversion.

cally closest cue, *Directed Distance* (DD) and *Bidirectional Distance* (BD). DD is extracted by following the reversed, directed edges from token X to the cue. If there is no such path, the value of the feature is -1. BD uses the Dijkstra shortest path algorithm on an undirected representation of the graph. The latter feature proved to be more effective than the former when not used together; using them in conjunction seemed to confuse the model, thus the final model utilizes only BD. We furthermore use the *Dependency Graph Path* (DGP) as a feature. This feature was inspired by the Parse Tree Path feature presented in Gildea and Jurafsky (2002) in the context of Semantic Role Labeling. It represents the path traversed from each token to the cue, encoding both the dependency relations and the direction of the arc that is traversed: for instance, the relation between *our* and *no* in Figure 1 is described as $\uparrow$ *poss* $\uparrow$ *dobj* $\downarrow$ *nsubj* $\downarrow$ *det*. Like Councill et al. (2010), we also encode the PoS of the first and second order syntactic head of each token. For the token *no* in Figure 1, for instance, we record the PoS of *one* and *escaped*, respectively.

3.2 Model-internal representation

The token-wise annotations in the CD corpus contain multiple layers of information. Tokens may or may not be negation cues and they can be either in or out of scope; in-scope tokens may or may not be negated events, and are associated with each of the cues they are negated by. Moreover, scopes may be (partially) overlapping, as in Figure 1, where the scope of *without* is contained within the scope of *never*. We convert this representation internally by assigning one of six labels to each token: O, CUE, MCUE, N, E and S, for out-of-scope, cue, morphological (affixal) cue, in-scope, event and negation stop respectively. The CUE, O, N and E labels parallel the IOB chunking paradigm and are eventually translated in the final annotations by our post-processing component. MCUE and S extend the label set to account for the specific behavior of the tokens they are associated with. The rationale behind the separation of cues in two classes is the pronounced differences between the PoS frequency distributions of standard versus morphological cues. Table 3 presents the frequency distribution of PoS-tags over the different cue types in CDTD and shows that, unsurprisingly, the majority class for morphological cues is adjectives, which typically generate different scope patterns compared to the majority class for standard cues. The S label, a special instance of an out-of-scope token, is defined as the

PoS	# S	PoS	# MCUE	PoS	# CUE
punctuation	1492	JJ	268	RB	1026
CC	52	RB	28	DT	296
IN + TO	46	NN	16	NN	146
RB	38	NN	4	UH	118
PRP	32	IN	2	IN	64
rest	118	rest	˜	rest	38

Table 3: Frequency distribution of parts of speech over the S, MCUE and CUE labels in CDTD.

first non-cue, out-of-scope token to the right of one labeled with N, and targets mostly punctuation.

After some experimentation with joint labeling of scopes and events, we opted for separation of the two models, hence training separate models for the two tasks of scope resolution and event detection. In the model for scopes, all E labels are switched to N; conversely, Ns become Os in the event model. Given the nature of the annotations, the predictions provided by the model for events serve a double purpose: finding the negated token in a sentence and deciding whether a sentence is factual or not. The outputs of the two classifiers are merged during post-processing.

3.3 Post-processing

A simple, heuristics-based algorithm was applied to the output of the labelers in order to pair each in-scope token to its negation cue(s) and determine overlaps. Our algorithm works by first determining the overlaps among negation cues. Cue A negates cue B if the following conditions are met:

- B is to the right of A.
- There are no tokens labeled with S between A and B.
- Token distance between A and B does not exceed 10.

In the example in Figure 1, the overlapping condition holds for *never* and *without* but not for *without* and *no*, because of the punctuation between them. The token distance threshold of 10 was determined empirically on CDT. In order to assign in-scope tokens to their respective cue, tokens labeled with N are treated as follows:

- Assign each token T to the closest negation cue A with no S-labeled tokens or punctuation separating it from T.
- If A was found to be negated by cue B, assign T to B as well.
- If T is labeled with E by the event classifier, mark it as an event.

		F_1	
	Configuration	**Closed**	**Open**
(A)	O, N, CUE, MCUE, E, S Dependency Features	**64.85**	**66.41**
(B)	O, N, CUE, MCUE, E, S No Dependency Features	59.35	59.35
(C)	O, N, CUE, E Dependency Features	62.69	63.24
(D)	O, N, CUE, E No Dependency Features	56.44	56.44

Table 4: Full negation results on CDD with gold cues.

This algorithm yields the correct annotations for the example in Figure 1; when applied to label sequences originating from the gold scopes in CDD, the reported F_1 is 95%. We note that this loss of information could have been avoided by presenting the CRF with a version of a sentence for each negation cue. Then, when labeling new sentences, the model could be applied repeatedly (based on the number of cues provided by the cue detection system). However, training with multiple instances of the same sentence could result in a dilution of the evidence needed for scope labeling; this remains to be investigated in future work.

4 Development results

To investigate the effects of the augmented set of labels and that of dependency features comparatively, we present four different configurations of our system in Table 4, using F_1 for the stricter score that counts perfect-match negation resolution for each negation cue. Comparing (B) and (D), we observe that explicitly encoding significant tokens with extra labels does improve the performance of the classifier. Comparing (A) to (B) and (C) to (B) shows the effect of the dependency features with and without the augmented set of labels. With (A) being our top performing system and (D) a kind of internal baseline, we observe that the combined effects of the labels and dependency features is beneficial, with a margin of about 8 and 10 percentage points for our closed and open track systems respectively.

Table 5 presents the results for scope resolution on CDD with gold cues. Interestingly, the constituent

	Closed			Open		
	Prec	**Rec**	**F$_1$**	**Prec**	**Rec**	**F$_1$**
Scopes	100.00	70.24	**82.52**	100.00	66.67	80.00
Scope Tokens	94.69	82.16	**87.98**	90.64	81.36	85.75
Negated	82.47	72.07	76.92	83.65	77.68	**80.55**
Full negation	100.00	47.98	64.85	100.00	49.71	**66.41**

Table 5: Results for scope resolution on CDD with gold cues.

trees converted to Stanford dependencies used in the closed track outperform the open system employing Maltparser on scopes, while for negated events the latter is over 5 percentage points better than the former, as shown in Table 5.

4.1 Error analysis

We performed a manual error analysis of the scope resolution on the development data using gold cue information. Since our system does not deal specifically with discontinuous scopes, and seeing that we are employing a sequence classifier with a fairly local window, we are not surprised to find that a substantial portion of the errors are caused by discontinuous scopes. In fact, in our closed track system, these errors amount to 34% of the total number of errors. Discontinuous scopes, as in (3) below, account for 9.3% of all scopes in CDD and the closed task system does not analyze any of these correctly, whereas the open system correctly analyzes one discontinuous scope.

(3) I therefore spent the day at my club and did **not** return to Baker Street until evening.

A similar analysis with respect to event detection on gold scope information indicated that errors are mostly due to either *predicting* an event for a *non-factual* context (false positive) or *not predicting* an event for a *factual* context (false negative), i.e., there are relatively few instances of predicting the wrong token for a factual context (which result in both a false negative and a false positive). This suggests that the CRF has learned what tokens should be labeled as an event for a negation, but has not learned so well how to determine whether the negation is factual or non-factual. In this respect it may be that incorporating information from a separate and dedicated component for factuality detection — as in the system of Read et al. (2012) — could yield improvements for the CRF event model.

5 Held-out evaluation

Final results on held-out data for both closed and open track submissions are reported in Table 6. For the final run, we trained our systems on CDTD. We observe a similar relative performance to our development results, with the open track system outperforming the closed track one, albeit by a smaller margin than what we saw in development. We are also surprised to see that despite not addressing discontinuous scopes directly, our system obtained the best score on scope resolution (according to the metric dubbed "Scopes (cue match)").

6 Conclusions and future work

This paper has provided an overview of our system submissions for the *SEM 2012 shared task on resolving negation. This involves the subtasks of identifying negations cues, identifying the in-sentence scope of these cues, as well as identifying negated (and factual) events. While a simple SVM token classifier is applied for the cue detection task, we apply CRF sequence classifiers for predicting scopes and events. For the CRF models we experimented with a fine-grained set of labels and a wide range of feature types, drawing heavily on information from dependency structures. We have detailed two different system configurations — one submitted for the open track and another for the closed track — and the two configurations only differ with respect to the source used for the dependency parses: For the closed track submission we simply converted the constituent structures provided in the shared task data to Stanford dependencies, while for the open track we apply the Maltparser. For the end-to-end evaluation, our submission was ranked first in the open track and third in the closed track. The system also had the best performance for each individual sub-task in the open track, as well as being among

	Closed			Open		
	Prec	**Rec**	**F_1**	**Prec**	**Rec**	**F_1**
Cues	89.17	93.56	91.31	89.17	93.56	91.31
Scopes	85.71	62.65	72.39	85.71	62.65	72.39
Scope Tokens	86.03	81.55	83.73	82.25	82.16	82.20
Negated	68.18	52.63	59.40	66.90	57.40	61.79
Full negation	78.26	40.91	53.73	78.72	42.05	54.82
Cues B	86.97	93.56	90.14	86.97	93.56	90.14
Scopes B	59.32	62.65	60.94	59.54	62.65	61.06
Negated B	67.16	52.63	59.01	63.82	57.40	60.44
Full negation B	38.03	40.91	39.42	39.08	42.05	40.51

Table 6: End-to-end results on the held-out data.

the top-performers on the scope resolution sub-task across both tracks.

Due to time constraints we were not able to directly address discontinuous scopes in our system. For future work we plan on looking for ways to tackle this problem by taking advantage of syntactic information, both in the classification and in the post-processing steps. We are also interested in developing the CRF-internal label-set to include more informative labels. We also want to test the system design developed for this task on other corpora annotated for negation (or other related phenomena such as speculation), as well as perform extrinsic evaluation of our system as a sub-component to other NLP tasks such as sentiment analysis or opinion mining. Lastly, we would like to try training separate classifiers for affixal and token-level cues, given that largely separate sets of features are effective for the two cases.

Acknowledgements

We thank colleagues at the University of Oslo, and in particular Johan Benum Evensberget and Arne Skjærholt for fruitful discussions and suggestions. We also thank the anonymous reviewers for their helpful feedback.

References

Isaac G. Councill, 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.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistic*, 28(3):245–288.

Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. In *Proceedings the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 504–513. Association for Computational Linguistics.

Roser Morante and Walter Daelemans. 2009. A metalearning approach to processing the scope of negation. In *CoNLL '09: Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 21–29. Association for Computational Linguistics.

Roser Morante and Walter Daelemans. 2012. ConanDoyle-neg: Annotation of negation in Conan Doyle stories. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, Istanbul.

Roser Morante, Sarah Schrauwen, and Walter Daelemans. 2011. Annotation of negation cues and their scope: Guidelines v1.0. Technical report, University of Antwerp. CLIPS: Computational Linguistics & Psycholinguistics technical report series.

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

Jonathon Read, Erik Velldal, Lilja Øvrelid, and Stephan Oepen. 2012. UiO$_1$: Constituent-based discriminative ranking for negation resolution. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, Montreal. Submission under review.

Erik Velldal, Lilja Øvrelid, Jonathon Read, and Stephan Oepen. 2012. Speculation and negation: Rules, rankers and the role of syntax. *Computational Linguistics*, 38(2).

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

Veronika Vincze, 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).