Exploiting CCG Structures with Tree Kernels for Speculation Detection

Liliana Mamani Sánchez, Baoli Li, Carl Vogel
Computational Linguistics Group
Trinity College Dublin
Dublin 2, Ireland
{mamanisl,baoli.li,vogel}@tcd.ie

Abstract

Our CoNLL-2010 speculative sentence detector disambiguates putative keywords based on the following considerations: a speculative keyword may be composed of one or more word tokens; a speculative sentence may have one or more speculative keywords; and if a sentence contains at least one real speculative keyword, it is deemed speculative. A tree kernel classifier is used to assess whether a potential speculative keyword conveys speculation. We exploit information implicit in tree structures. For prediction efficiency, only a segment of the whole tree around a speculation keyword is considered, along with morphological features inside the segment and information about the containing document. A maximum entropy classifier is used for sentences not covered by the tree kernel classifier. Experiments on the Wikipedia data set show that our system achieves 0.55 F-measure (in-domain).

1 Introduction

Speculation and its impact on argumentation has been studied by linguists and logicians since at least as far back as Aristotle (trans 1991, 1407a, 1407b), and under the category of linguistic "hedges" since Lakoff (1973). Practical application of this research has emerged due to the efforts to create a biomedical database of sentences tagged with speculation information: BioScope (Szarvas et al., 2008) and because of the association of some kinds of Wikipedia data with the speculation phenomenon (Ganter and Strube, 2009). It is clear that specific words can be considered as clues that can qualify a sentence as speculative. However, the presence of a speculative keyword not always conveys a speculation assertion which makes the speculation detection a tough problem. For instance, the sentences below contain the speculative keyword "*may*", but only the sentence (a) is speculative.

(a) *These effects* **may** *be reversible.*

(b) *Members of an alliance* **may** *not attack each other.*

The CoNLL-2010 Shared Task (Farkas et al., 2010), "Learning to detect hedges and their scope in natural language text" proposed two tasks related to speculation research. Task 1 aims to detect sentences containing uncertainty and Task 2 aims to resolve the intra-sentential scope of hedge cues. We engaged in the first task in the biomedical and Wikipedia domains as proposed by the organizers, but eventually we got to submit only Wikipedia domain results. However, in this paper we include results in the biomedical domain as well.

The BioScope corpus is a linguistically hand annotated corpus of negation and speculation phenomena for medical free texts, biomedical article abstracts and full biomedical articles. The aforesaid phenomena have been annotated at sentence level with keyword tags and linguistic scope tags. Some previous research on speculation detection and boundary determination over biomedical data has been done by Medlock & Briscoe (2007) and Özgür & Radev (2009) from a computational view using machine learning methods.

The Wikipedia speculation dataset was generated by exploiting a weasel word marking. As weasel words convey vagueness and ambiguity by providing an unsupported opinion, they are discouraged by Wikipedia editors. Ganter & Strube (2009) proposed a system to detect hedges based on frequency measures and shallow information, achieving a F-score of 0.69[1].

We formulate the speculation detection problem as a word disambiguation problem and developed a system as a pipelined set of natural

[1] They used different Wikipedia data.

Proceedings of the Fourteenth Conference on Computational Natural Language Learning: Shared Task, pages 126–131,
Uppsala, Sweden, 15-16 July 2010.

language processing tools and procedures to preprocess the datasets. A Combinatory Categorial Grammar parsing (CCG) (Steedman, 2000) tool and a Tree Kernel (TK) classifier constitute the core of the system.

The Section 2 of this paper describes the overall architecture of our system. Section 3 depicts the dataset pre-processing. Section 4 shows how we built the speculation detection module, outlines the procedure of examples generation and the use of the Tree-kernel classifier. Section 5 presents the experiments and results, we show that sentence CCG derivation information helps to differentiate between apparent and real speculative words for speculation detection. Finally Section 6 gives our conclusions.

2 Speculation detection system

Our system for speculation detection is a machine learning (ML) based system (Figure 1). In the preprocessing module a dataset of speculative/non-speculative sentences goes through a process of information extraction of three kinds: speculative word or keyword extraction,[2] sentence extraction and document feature extraction (i.e document section). Later the extracted keywords are used to tag potential speculative sentences in the training/evaluation datasets and used as features by the classifiers. The sentences are submitted to the tokenization and parsing modules in order to provide a richer set of features necessary for creating the training/evaluation datasets, including the document features as well.

In the ML module two types of dataset are built: one used by a TK classifier and other one by a bag-of-features based maximum entropy classifier. As the first one processes only those sentences that contain speculative words, we use the second classifier, which is able to process samples of all the sentences.

The models built by these classifiers are combined in order to provide a better performance and coverage for the speculation problem in the classification module which finally outputs sentences labeled as speculative or non-speculative. Used tools are the GeniaTagger (Tsuruoka et al., 2005) for tokenization and lemmatization, and the C&C Parser (Clark and Curran, 2004). The next sections explain in detail the main system components.

3 Dataset pre-processing for rich feature extraction

The pre-processing module extracts keywords, sentences and document information.

All sentences are processed by the tokenizer/lemmatizer and at the same time specific information about the keywords is extracted.

Speculative keywords

Speculative sentences are evidenced by the presence of speculation keywords. We have the following observations:

- A hedge cue or speculative keyword [3] may be composed of one or more word tokens.
- In terms of major linguistic categories, the word tokens are heterogeneous: they may be verbs, adjectives, nouns, determiners, etc. A stop-word removing strategy was dismissed, since no linguistic category can be eliminated.
- A keyword may be covered by another longer one. For instance, the keyword *most* can be seen in keywords like *most of all the heroes* or *the most common*.

Considering these characteristics for each sentence, in the training stage, the keyword extraction module retrieves the speculative/non-speculative property of each sentence, the keyword occurrences, number of keywords in a sentence, the initial word token position and the number of word tokens in the keyword. We build a keyword lexicon with all the extracted keywords and their frequency in the training dataset, this speculative keyword lexicon is used to tag keyword occurrences in non-speculative training sentences and in all the evaluation dataset sentences.

The overlapping problem when tagging keywords is solved by maximal matching strategy. It is curious that speculation phrases come in degrees of specificity; the approach adopted here favors “specific” multi-word phrases over single-word expressions.

Sentence processing

Often, speculation keywords convey certain information that can not be successfully expressed by morphology or syntactic relations provided by phrase structure grammar parsers. On the other

[2] Extraction of keywords for the training stage.

[3] Or just “keyword” for sake of simplicity.

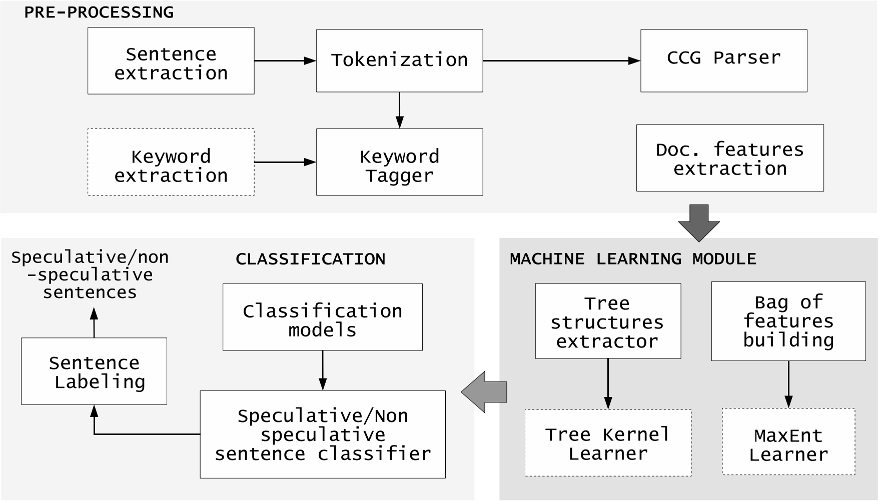

Figure 1: Block diagram for the speculation detection system.

hand, CCG derivations or dependencies provide deeper information, in form of predicate-argument relations. Previous works on semantic role labeling (Gildea and Hockenmaier, 2003; Boxwell et al., 2009) have used features derived from CCG parsings and obtained better results.

C&C parser provides CCG predicate-argument dependencies and Briscoe and Carroll (2006) style grammatical relations. We parsed the tokenized sentences to obtain CCG derivations which are binary trees as shown in the Figure 2. The CCG derivation trees contain function category and part-of-speech labels; this information is contained in the tree structures to be used in building a subtree dataset for the TK classifier.

4 Speculative sentence classifier

4.1 Tree Kernel classification

The subtree dataset is processed by a Tree Kernel classifier (Moschitti, 2006) based on Support Vector Machines. TK uses a kernel function between two trees, allowing a comparison between their substructures, which can be subtrees (ST) or subset trees (SST). We chose the comparison between subset trees since it expands the kernel calculation to those substructures with constituents that are not in the leaves. Our intuition is that real speculative sentences have deep semantic structures that are particularly different from those ones in apparent speculative sentences, and consequently the comparison between the structures of well identified and potential speculative sentences may enhance the identification of real speculative keywords.

4.2 Extracting tree structures

The depth of a CCG derivation tree is proportional to the number of word tokens in the sentence. Therefore, the processing of a whole derivation tree by the classifier is highly demanding and many subtrees are not relevant for the classification of speculative/non-speculative sentences, in particular when the scope of the speculation is a small proportion of a sentence.

In order to tackle this problem, a fragment of the CCG derivation tree is extracted. This fragment or subtree spans the keyword together with neighbors terms in a fixed-size window of n word tokens, (i.e. n word tokens to the left and n word tokens to the right of the keyword) and has as root the lower upper bound node of the first and last tokens of this span. After applying the subtree extraction, the subtree can contain more word tokens in addition to those contained in the n-span, which are replaced by a common symbol.

Potential speculative sentences are turned into training examples. However, as described in Section 3, a speculative sentence can contain one or more speculative keywords. This can produce an overlapping between their respective n-spans of individual keywords during the subtree extraction, producing subtrees with identical roots for both keywords. For instance, in the following sentence(c), the spans for the keywords *suggests* and *thought* will overlap if $n = 3$.

(c) *This* ***suggests*** *that diverse agents* ***thought*** *to activate NF-kappa B ...*

The overlapping interacts with the windows size and potential extraction of dependency relations

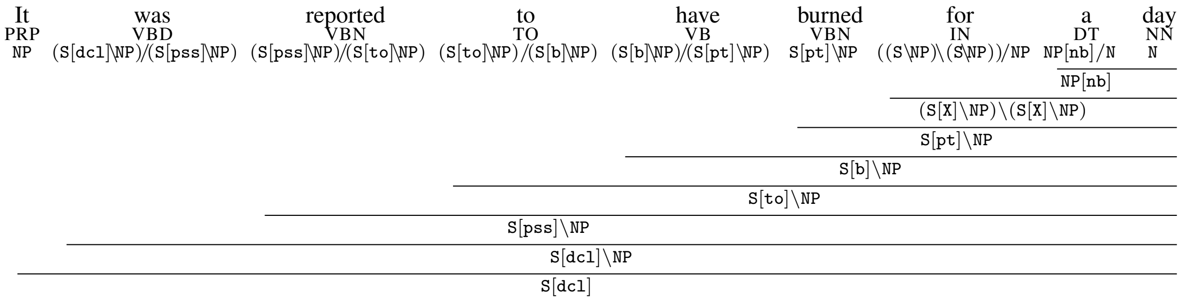
It was reported to have burned for a day
PRP VBD VBN TO VB VBN IN DT NN
NP (S[dcl]\NP)/(S[pss]\NP) (S[pss]\NP)/(S[to]\NP) (S[to]\NP)/(S[b]\NP) (S[b]\NP)/(S[pt]\NP) S[pt]\NP ((S\NP)\(S\NP))/NP NP[nb]/N N
NP[nb]
(S[X]\NP)\(S[X]\NP)
S[pt]\NP
S[b]\NP
S[to]\NP
S[pss]\NP
S[dcl]\NP
S[dcl]

Figure 2: CCG derivations tree for *It was reported to have burned for a day.*

shared by terms belonging to the two different spans. We deal with this issue by extracting one training example if two spans have a common root and two different examples otherwise.

4.3 Bag of features model

By default, our system classifies the sentences not covered by the TK model using a baseline classifier that labels a sentence as speculative if this has at least one keyword. Alternatively, a bag of features classifier is used to complement the tree kernel, aimed to provide a more precise method that might detect even speculative sentences with new keywords in the evaluation dataset. The set of features used to build this model includes:

a) Word unigrams;
b) Lemma unigrams;
c) Word+POS unigrams;
d) Lemma+POS unigrams;
e) Word+Supertag unigrams;
f) Lemma+Supertag unigrams;
g) POS+Supertag unigrams;
h) Lemma bigrams;
i) POS bigrams;
j) Supertag bigrams;
k) Lemma+POS bigrams;
l) Lemma+Supertag bigrams;
m) POS+Supertag bigrams;
n) Lemma trigrams;
o) POS trigrams;
p) Supertag trigrams;
q) Lemma+POS trigrams;
r) Lemma+Supertag trigrams;
s) POS+Supertag trigrams;
t) Number of tokens;
u) Type of section in the document (Title, Text, Section);
v) Name of section in the document;
w) Position of the sentence in a section starting from beginning;
x) Position of the sentence in a section starting from end.

Position of the sentence information, composed by the last four features, represents the information about the sentence relative to a whole document. The bag of features model is generated using a Maximum Entropy algorithm (Zhang, 2004).

Dataset	Dev.	Train.	Eval.
Biomedical	39	14541	5003
Wikipedia	124	11111	9634

Table 1: Datasets sizes.

5 Experiments and results

5.1 Datasets

In the CoNLL-2010 Task 1, biomedical and Wikipedia datasets were provided for development, training and evaluation in the BioScope XML format. Development and training datasets are tagged with cue labels and a certainty feature.[4] The number of sentences for each dataset [5] is detailed in Table 1.

After manual revision of sentences not parsed by C&C parser, we found that they contain equations, numbering elements (e.g. (i), (ii).. 1), 2)), or long n-grams of named-entities, for instance: *...mannose-capped lipoarabinomannan (ManLAM) of Mycobacterium tuberculosis (M. tuberculosis)...* that out of a biomedical domain appear to be ungrammatical. Similarly, in the Wikipedia datasets, some sentences have many named entities. This suggests the need of a specific pre-processor or a parser for this kind of sentences like a named entity tagger.

In Table 2, we present the number of parsed sentences, processed sentences by the TK model and examples obtained in the tree structure extraction.

[4] certainty="uncertain" and certainty="certain".

[5] The biomedical abstracts and biomedical articles training datasets are processed as a single dataset.

Dataset	Parsed	Process.	Samples
Biomedical train.	14442	10852	23511
Biomedical eval.	4903	3395	7826
Wikipedia train.	10972	7793	13461
Wikipedia eval.	9559	4666	8467

Table 2: Count of processed sentences.

5.2 Experimental results

The CoNLL-2010 organizers proposed in-domain and cross-domain evaluations. In cross-domain experiments, test datasets of one domain can be used with classifiers trained on the other or on the union of both domains. We report here our results for the Wikipedia and biomedical datasets.

So far, we mentioned two settings for our classifier: a TK classifier complemented by a baseline classifier (BL) and TK classifier complemented by a bag of features classifier (TK+BF). Table 3 shows the scores of our submitted system (in-domain Task 1) on the Wikipedia dataset, whereas Table 4 gives the scores of the baseline system.

	TP	FP	FN	Precision	Recall	F
Our system	1033	480	1201	0.6828	0.4624	0.5514
Max.	1154	448	1080	0.7204	0.5166	0.6017
Min.	147	9	2087	0.9423	0.0658	0.123

Table 3: Comparative scores for our system with CoNLL official maximum and minimum scores in Task 1, Wikipedia dataset in-domain.

	TP	FP	FN	Precision	Recall	F
Biomedical	786	2690	4	0.2261	0.9949	0.3685
Wikipedia	1980	2747	254	0.4189	0.8863	0.5689

Table 4: Baseline results.

Additionally, we consider a bag of features classifier (BF) and a classifier that combines the baseline applied to the sentences that have at least one keyword plus the BF classifier for the remaining sentences (BL+BF). In Tables 5 to 10, results for the four classifiers (TK, TK+BF, BF, BL+BF) with evaluations in-domain and cross-domain are presented[6].

	TP	FP	FN	Precision	Recall	F
BL	1980	2747	254	0.4189	0.8863	0.5689
TK	1033	480	1201	0.6828	0.4624	0.5514
TK+BF	1059	516	1175	0.6729	0.4740	0.5560
BF	772	264	1462	0.7452	0.3456	0.4722
BL+BF	2028	2810	206	0.4192	0.9078	0.5735

Table 5: Results for Wikipedia dataset in-domain.

	TP	FP	FN	Precision	Recall	F
BL	1980	2747	254	0.4189	0.8863	0.5689
TK	1776	2192	458	0.4476	0.7950	**0.5727**
TK+BF	1763	2194	471	0.4455	0.7892	0.5695
BF	403	323	1831	0.5551	0.1804	0.2723
BL+BF	1988	2772	246	0.4176	0.8899	0.5685

Table 6: Wikipedia data classified with biomedical model scores (cross-domain).

	TP	FP	FN	Precision	Recall	F
BL	1980	2747	254	0.4189	0.8863	0.5689
TK	1081	624	1153	0.6340	0.4839	0.5489
TK+BF	1099	636	1135	0.6334	0.4919	0.5538
BF	770	271	1464	0.7397	0.3447	0.4702
BL+BF	2017	2786	217	0.4199	0.9029	0.5733

Table 7: Wikipedia data classified with biomedical + Wikipedia model scores (cross-domain).

	TP	FP	FN	Precision	Recall	F
BL	786	2690	4	0.2261	0.9949	0.3685
TK	759	777	31	0.4941	0.9606	0.6526
TK+BF	751	724	39	0.5092	0.9506	0.6631
BF	542	101	248	0.8429	0.6861	**0.7565**
BL+BF	786	2695	4	0.2258	0.9949	0.3681

Table 8: Biomedical data scores (in-domain).

	TP	FP	FN	Precision	Recall	F
BL	786	2690	4	0.2261	0.9949	0.3685
TK	786	2690	4	0.2261	0.9949	0.**3685**
TK+BF	771	2667	19	0.2243	0.9759	0.3647
BF	174	199	616	0.4665	0.2206	0.2992
BL+BF	787	2723	3	0.2242	0.9962	0.3660

Table 9: Biomedical data classified with Wikipedia model scores (cross-domain).

	TP	FP	FN	Precision	Recall	F
BL	786	2690	4	0.2261	0.9949	0.3685
TK	697	357	93	0.6613	0.8823	0.7560
TK+BF	685	305	105	0.6919	0.8671	**0.7697**
BF	494	136	296	0.7841	0.6253	0.6958
BL+BF	786	2696	4	0.2257	0.9949	0.3679

Table 10: Biomedical data classified with biomedical + Wikipedia model scores (cross-domain).

The baseline scores confirm that relying on just the keywords is not enough to identify speculative sentences. In the biomedical domain, the classifiers give high recall but too low precision resulting in low F-scores. Still, the TK, TK+BF and BF (in-domain configurations) gives much better results than BL and BL+BF which indicates that the information from CCG improves the performance of the classifiers when compared to the baseline classifier.

Even though in the Wikipedia domain the TK+BF score is less than the baseline score, still the performance of the classifiers do not fall much in any of the in-domain and cross-domain experiments. On the other hand, BF does not have a good performance in 5 of 6 the experiments. To make a more precise comparison between TK and BF, the TK and BL+BF scores show that BL+BF performs better than TK in only 2 of the 6 experiments but the better performances achieved by BL+BF are very small. This suggests that

[6] It is worth to note that the keyword lexicons have been not used in cross-domain way, so the TK and TK+BF models have not been tested in regards to keywords.

the complex processing made by tree kernels is more useful when disambiguating speculative keywords than BF. Nonetheless, the bag-of-features approach is also of importance for the task at hand when combined with TK. We observe that the TK classifer and BF classifier perform well making us believe that the CCG derivations provide relevant information for speculation detection. The use of tree kernels needs further investigations in order to evaluate the suitability of this approach.

6 Concluding remarks

Speculation detection is found to be a tough task given the high ambiguity of speculative keywords. We think these results can be improved by studying the influences of context on speculation assertions.

This paper presents a new approach for disambiguating apparent speculative keywords by using CCG information in the form of supertags and CCG derivations. We introduce the use of the tree kernel approach for CCG derivations trees. The inclusion of other features like grammatical relations provided by the parser needs to be studied before incorporating this information into the current classifier and possibly to resolve the boundary speculation detection problem.

Acknowledgments

This research is supported by the Trinity College Research Scholarship Program and the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Trinity College of Dublin.

References

Aristotle. trans. 1991. *The Art of Rhetoric*. Penguin Classics, London. Translated with an Introduction and Notes by H.C. Lawson-Tancred.

Stephen Boxwell, Dennis Mehay, and Chris Brew. 2009. Brutus: A semantic role labeling system incorporating CCG, CFG, and dependency features. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 37–45, Suntec, Singapore.

Ted Briscoe and John Carroll. 2006. Evaluating the accuracy of an unlexicalized statistical parser on the PARC DepBank. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 41–48, Morristown, NJ, USA.

Stephen Clark and James R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 103, Morristown, NJ, USA.

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 the Fourteenth Conference on Computational Natural Language Learning (CoNLL-2010): Shared Task*, pages 1–12, Uppsala, Sweden, July. Association for Computational Linguistics.

Viola Ganter and Michael Strube. 2009. Finding hedges by chasing weasels: Hedge detection using Wikipedia tags and shallow linguistic features. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 173–176, Suntec, Singapore.

Daniel Gildea and Julia Hockenmaier. 2003. Identifying semantic roles using combinatory categorial grammar. In *Proceedings of 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sapporo, Japan.

George Lakoff. 1973. Hedges: A study in meaning criteria and the logic of fuzzy concepts. *Journal of Philosophical Logic*, 2(4):458–508.

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, Prague, Czech Republic.

Alessandro Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*.

Arzucan Özgür and Dragomir R. Radev. 2009. Detecting speculations and their scopes in scientific text. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1398–1407, Singapore.

Mark Steedman. 2000. *The syntactic process*. MIT Press, Cambridge, MA, USA.

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

Yoshimasa Tsuruoka, 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 *Advances in Informatics*, pages 382–392.

Le Zhang. 2004. Maximum entropy modeling toolkit for Python and C++ (version 20041229). In *Natural Language Processing Lab, Northeastern*.