Detecting Hedge Cues and their Scopes with Average Perceptron

Feng Ji, Xipeng Qiu, Xuanjing Huang
Fudan University
{fengji,xpqiu,xjhuang}@fudan.edu.cn

Abstract

In this paper, we proposed a hedge detection method with average perceptron, which was used in the closed challenge in CoNLL-2010 Shared Task. There are two subtasks: (1) detecting uncertain sentences and (2) identifying the in-sentence scopes of hedge cues. We use the unified learning algorithm for both subtasks since that the hedge score of sentence can be decomposed into scores of the words, especially the hedge words. On the biomedical corpus, our methods achieved F-measure with 77.86% in detecting in-domain uncertain sentences, 77.44% in recognizing hedge cues, and 19.27% in identifying the scopes.

1 Introduction

Detecting hedged information in biomedical literatures has received considerable interest in the biomedical natural language processing (NLP) community recently. Hedge information indicates that authors do not or cannot back up their opinions or statements with facts (Szarvas et al., 2008), which exists in many natural language texts, such as webpages or blogs, as well as biomedical literatures.

For many NLP applications, such as question answering and information extraction, the information extracted from hedge sentences would be harmful to their final performances. Therefore, the hedge or speculative information should be detected in advance, and dealt with different approaches or discarded directly.

In CoNLL-2010 Shared Task (Farkas et al., 2010), there are two different level subtasks: detecting sentences containing uncertainty and identifying the in-sentence scopes of hedge cues.

For example, in the following sentence:

> These results **suggest** that the IRE motif in the ALAS mRNA is functional and **imply** that translation of the mRNA is controlled by cellular iron availability during erythropoiesis.

The words **suggest** and **imply** indicate that the statements are not supported with facts.

In the first subtask, the sentence is considered as uncertainty.

In the second subtask, **suggest** and **imply** are identified as hedge cues, while the consecutive blocks *suggest that the IRE motif in the ALAS mRNA is functional* and *imply that translation of the mRNA is controlled by cellular iron availability during erythropoiesis* are recognized as their corresponding scopes.

In this paper, we proposed a hedge detection method with average perceptron (Collins, 2002), which was used in the closed challenges in CoNLL-2010 Shared Task (Farkas et al., 2010). Our motivation is to use a unified model to detect two level hedge information (word-level and sentence-level) and the model is easily expanded to joint learning of two subtasks. Since that the hedge score of sentence can be decomposed into scores of the words, especially the hedge words, we chosen linear classifier in our method and used average perceptron as the training algorithm.

The rest of the paper is organized as follows. In Section 2, a brief review of related works is presented. Then, we describe our method in Section 3. Experiments and results are presented in the section 4. Finally, the conclusion will be presented in Section 5.

2 Related works

Although the concept of hedge information has been introduced in linguistic community for a long time, researches on automatic hedge detection emerged from machine learning or compu-

Proceedings of the Fourteenth Conference on Computational Natural Language Learning: Shared Task, pages 32–39,
Uppsala, Sweden, 15-16 July 2010.

tational linguistic perspective in recent years. In this section, we give a brief review on the related works.

For speculative sentences detection, Medlock and Briscoe (2007) report their approach based on weakly supervised learning. In their method, a statistical model is initially derived from a seed corpus, and then iteratively modified by augmenting the training dataset with unlabeled samples according the posterior probability. They only employ bag-of-words features. On the public biomedical dataset[1], their experiments achieve the performance of 0.76 in BEP (break even point). Although they also introduced more linguistic features, such as part-of-speech (POS), lemma and bigram (Medlock, 2008), there are no significant improvements.

In Ganter and Strube (2009), the same task on Wikipedia is presented. In their system, score of a sentence is defined as a normalized tangent value of the sum of scores over all words in the sentence. Shallow linguistic features are introduced in their experiments.

Morante and Daelemans (2009) present their research on identifying hedge cues and their scopes. Their system consists of several classifiers and works in two phases, first identifying the hedge cues in a sentence and secondly finding the full scope for each hedge cue. In the first phase, they use IGTREE algorithm to train a classifier with 3 categories. In the second phase, three different classifiers are trained to find the first token and last token of in-sentence scope and finally combined into a meta classifier. The experiments shown that their system achieves an F1 of nearly 0.85 of identifying hedge cues in the abstracts sub corpus, while nearly 0.79 of finding the scopes with predicted hedge cues. More experiments could be found in their paper (Morante and Daelemans, 2009). They also provide a detail statistics on hedge cues in BioScope corpus[2].

3 Hedge detection with average perceptron

3.1 Detecting uncertain sentences

The first subtask is to identify sentences containing uncertainty information. In particular, this subtask is a binary classification problem at sentence-level.

We define the score of sentence as the confidence that the sentence contains uncertainty information.

The score can be decomposed as the sum of the scores of all words in the sentence,

$$S(\mathbf{x}, y) = \sum_{x_i \in \mathbf{x}} s(x_i, y) = \sum_{x_i \in \mathbf{x}} \mathbf{w}^T \phi(x_i, y)$$

where, $\mathbf{x}$ denotes a sentence and x_i is the i-th word in the sentence $\mathbf{x}$, $\phi(x_i, y)$ is a sparse high-dimensional binary feature vector of word x_i. $y \in \{\textbf{uncertain}, \textbf{certain}\}$ is the category of the sentence. For instance, in the example sentence, if current word is **suggest** while the category of this sentence is uncertain, the following feature is hired,

$$\phi_n(x_i, y) = \begin{cases} 1, & \texttt{if } \substack{x_i=\texttt{``suggest''} \\ y=\texttt{``uncertain''}}, \\ 0, & otherwise \end{cases}$$

where n is feature index.

This representation is commonly used in structured learning algorithms. We can combine the features into a sparse feature vector $\Phi(\mathbf{x}, y) = \sum_i \phi(x_i, y)$.

$$S(\mathbf{x}, y) = \mathbf{w}^T \Phi(x, y) = \sum_{x_i \in \mathbf{x}} \mathbf{w}^T \phi(x_i, y)$$

In the predicting phase, we assign $\mathbf{x}$ to the category with the highest score,

$$y^* = \arg\max_y \mathbf{w}^T \Phi(x, y)$$

We learn the parameters $\mathbf{w}$ with online learning framework. The most common online learner is the perceptron (Duda et al., 2001). It adjusts parameters $\mathbf{w}$ when a misclassification occurs. Although this framework is very simple, it has been shown that the algorithm converges in a finite number of iterations if the data is linearly separable. Moreover, much less training time is required in practice than the batch learning methods, such as support vector machine (SVM) or conditional maximum entropy (CME).

Here we employ a variant perceptron algorithm to train the model, which is commonly named average perceptron since it averages parameters $\mathbf{w}$ across iterations. This algorithm is first proposed in Collins (2002). Many experiments of

[1] http://www.benmedlock.co.uk/hedgeclassif.html

[2] http://www.inf.u-szeged.hu/rgai/bioscope

NLP problems demonstrate better generalization performance than non averaged parameters. More theoretical proofs can be found in Collins (2002). Different from the standard average perceptron algorithm, we slightly modify the average strategy. The reason to this modification is that the original algorithm is slow since parameters accumulate across all iterations. In order to keep fast training speed and avoid overfitting at the same time, we make a slight change of the parameters accumulation strategy, which occurs only after each iteration over the training data finished. Our training algorithm is shown in Algorithm 1.

input : training data set:
$(x_n, y_n), n = 1, \cdots, N$,
parameters: average number: K,
maximum iteration number: T.
output: average weight: $\mathbf{cw}$

Initialize: $\mathbf{cw} \leftarrow 0$,;
for $k = 0 \cdots K-1$ **do**
  $\mathbf{w}_0 \leftarrow 0$;
  for $t = 0 \cdots T-1$ **do**
    receive an example $(\mathbf{x}_t, y_t)$;
    predict: $\hat{y}_t = \arg\max_y \ \mathbf{w}_t^T \Phi(\mathbf{x}_t, y)$;
    if $\hat{y}_t \neq y_t$ **then**
      $\mathbf{w}_{t+1} = \mathbf{w}_t + \Phi(\mathbf{x}_t, y_t) - \Phi(\mathbf{x}_t, \hat{y}_t)$
    end
  end
  $\mathbf{cw} = \mathbf{cw} + \mathbf{w}_T$;
end
$\mathbf{cw} = \mathbf{cw}/K$;

Algorithm 1: Average Perceptron algorithm

Binary context features are extracted from 6 predefined patterns, which are shown in Figure 1. By using these patterns, we can easily obtain the complicate features. As in the previous example, if the current word is *suggest*, then a new compound feature could be extracted in the form of w_{-1} =*results*//w_0 =*suggest* by employing the pattern $w_{-1}w_0$. // is the separate symbol.

3.2 Identifying hedge cues and their scopes

Our approach for the second subtask consists of two phases: (1) identifying hedge cues in a sentence, then (2) recognizing their corresponding scopes.

3.2.1 Identifying hedge cues

Hedge cues are the most important clues for determining whether a sentence contains uncertain information. Therefore in this phase, we treat the problem of identifying hedge cues as a classification problem. Each word in a sentence would be predicted a category indicating whether this word is a hedge cue word or not. In the previous example, there are two different hedge cues in the sentence (show in bold manner). Words **suggest** and **imply** are assigned with the category *CUE* denoting hedge cue word, while other words are assigned with label *O* denoting non hedge cue word.

In our system, this module is much similar to the module of detecting uncertain sentences. The only difference is that this phase is word level. So that each training sample in this phase is a word, while in detecting speculative sentences training sample is a sentence. The training algorithm is the same as the algorithm shown in Algorithm 1. 12 predefined patterns of context features are shown in Figure 2.

- **unigram**: w_0,p_0
- **bigram**: w_0w_1, w_0p_0, p_0p_1
- **trigram**: $w_{-1}w_0w_1$

Figure 1: Patterns employed in the sentence-level hedge detection. Here w denotes single word, p is part of speech, and the subscript denotes the relative offset compared with current position.

- **unigram**: w_{-2}, w_{-1}, w_0, w_1, w_2, p_0
- **bigram**: $w_{-1}w_0$, w_0w_1, w_0p_0, $p_{-1}p_0$, p_0p_1
- **trigram**: $w_{-1}w_0w_1$

Figure 2: Patterns employed in the word-level hedge detection.

3.2.2 Recognizing in-sentence scopes

After identifying the hedge cues in the first phase, we need to recognize their corresponding in-sentence scopes, which means the boundary of scope should be found within the same sentence.

We consider this problem as a word-cue pair classification problem, where word is any word in a sentence and cue is the identified hedge cue word. Similar to the previous phase, a word-level linear classifier is trained to predict whether each

word-cue pair in a sentence is in the scope of the hedge cue.

Besides base context features used in the previous phase, we introduce additional syntactic dependency features. These features are generated by a first-order projective dependency parser (McDonald et al., 2005), and listed in Figure 3.

The scopes of hedge cues are always covering a consecutive block of words including the hedge cue itself. The ideal method should recognize only one consecutive block for each hedge cue. However, our classifier cannot work so well. Therefore, we apply a simple strategy to process the output of the classifier. The simple strategy is to find a maximum consecutive sequence which covers the hedge cue. If a sentence is considered to contain several hedge cues, we simply combine the consecutive sequences, which have at least one common word, to a large block and assign it to the relative hedge cues.

4 Experiments

In this section, we report our experiments on datasets of CoNLL-2010 shared tasks, including the official results and our experimental results when developing the system.

Our system architecture is shown in Figure 4, which consists of the following modules.

1. corpus preprocess module, which employs a tokenizer to normalize the corpus;

2. sentence detection module, which uses a binary sentence-level classifier to determine whether a sentence contains uncertainty information;

3. hedge cues detection module, which identifies which words in a sentence are the hedge cues, we train a binary word-level classifier;

4. cue scope recognition module, which recognizes the corresponding scope for each hedge cue by another word-level classifier.

Our experimental results are obtained on the training datasets by 10-fold cross validation. The maximum iteration number for training the average perceptron is set to 20. Our system is implemented with Java[3].

[3] http://code.google.com/p/fudannlp/

	biomedical	**Wikipedia**
#sentences	14541	11111
#words	382274	247328
#hedge sentences	2620	2484
%hedge sentences	0.18	0.22
#hedge cues	3378	3133
average number	1.29	1.26
average cue length	1.14	2.45
av. scope length	15.42	-

Table 1: Statistical information on annotated corpus.

4.1 Datasets

In CoNLL-2010 Shared Task, two different datasets are provided to develop the system: (1) biological abstracts and full articles from the BioScope corpus, (2) paragraphs from Wikipedia. Besides manually annotated datasets, three corresponding unlabeled datasets are also allowed for the closed challenges. But we have not employed any unlabeled datasets in our system.

A preliminary statistics can be found in Table 1. We make no distinction between sentences from abstracts or full articles in biomedical dataset. From Table 1, most sentences are certainty while about 18% sentences in biomedical dataset and 22% in Wikipedia dataset are speculative. On the average, there exists nearly 1.29 hedge cues per sentence in biomedical dataset and 1.26 in Wikipedia. The average length of hedge cues varies in these two corpus. In biomedical dataset, hedge cues are nearly one word, but more than two words in Wikipedia. On average, the scope of hedge cue covers 15.42 words.

4.2 Corpus preprocess

The sentence are processed with a maximum-entropy part-of-speech tagger[4] (Toutanova et al., 2003), in which a rule-based tokenzier is used to separate punctuations or other symbols from regular words. Moreover, we train a first-order projective dependency parser with MSTParser[5] (McDonald et al., 2005) on the standard WSJ training corpus, which is converted from constituent trees to dependency trees by several heuristic rules[6].

[4] http://nlp.stanford.edu/software/tagger.shtml

[5] http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html

[6] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html

- **word-cue pair**: current word and the hedge cue word pair,
- **word-cue POS pair**: POS pair of current word and the hedge cue word,
- **path of POS**: path of POS from current word to the hedge cue word along dependency tree,
- **path of dependency**: relation path of dependency from current word to the hedge cue word along dependency tree,
- **POS of hedge cue word+direction**: POS of hedge cue word with the direction to the current word. Here direction can be "LEFT" if the hedge cue is on the left to the current word, or "RIGHT" on the right,
- **tree depth**: depth of current in the corresponding dependency tree,
- **surface distance**: surface distance between current word and the hedge cue word. The value of this feature is always 10 in the case of surface distance greater than 10,
- **surface distance+tree depth**: combination of surface distance and tree depth
- **number of punctuations**: number of punctuations between current word and the hedge cue word,
- **number of punctuations + tree depth**: combination of number of punctuations and tree depth

Figure 3: Additional features used in recognizing in-sentence scope

4.3 Uncertain sentences detection

In the first subtask, we carry out the experiments within domain and cross domains. As previously mentioned, we do not use the unlabeled datasets and make no distinction between abstracts and full articles in biomedical dataset. This means we train the models only with the official annotated datasets. The model for cross-domain is trained on the combination of annotated biomedical and Wikipedia datasets.

In this subtask, evaluation is carried out on the sentence level and F-measure of uncertainty sentences is employed as the chief metric.

Table 2 shows the results within domain. After 10-fold cross validation over training dataset, we achieve 84.39% of F1-measure on biomedical while 56.06% on Wikipedia.

We analyzed the low performance of our submission result on Wikipedia. The possible reason is our careless work when dealing with the trained model file. Therefore we retrain a model for Wikipedia and the performance is listed on the bottom line (Wikipedia*) in Table 2.

Dataset	**Precision**	**Recall**	**F1**
10-fold cross validation			
biomedical	91.03	78.66	84.39
Wikipedia	66.54	48.43	56.06
official evaluation			
biomedical	79.45	76.33	77.86
Wikipedia	94.23	6.58	1.23
Wikipedia*	82.19	32.86	46.95

Table 2: Results for in-domain uncertain sentences detection

Table 3 shows the results across domains. We split each annotated dataset into 10 folds. Then training dataset is combined by individually drawing 9 folds out from the split datasets and the rests are used as the test data. On biomedical dataset, F1-measure gets to 79.24% while 56.16% on Wikipedia dataset. Compared with the results within domain, over 5% performance decreases from 84.39% to 79.24% on biomedical, but a slightly increase on Wikipedia.

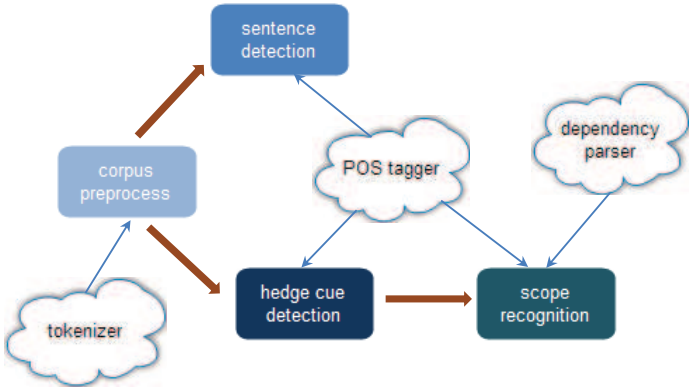

Figure 4: System architecture of our system

Dataset	Precision	Recall	F1
10-fold cross validation			
biomedical	87.86	72.16	79.24
Wikipedia	67.78	47.95	56.16
official evaluation			
biomedical	62.81	79.11	70.03
Wikipedia	62.66	55.28	58.74

Table 3: Results for across-domain uncertain sentences detection

Dataset	Precision	Recall	F1
10-fold cross validation(word-level)			
biomedical	90.15	84.43	87.19
Wikipedia	57.74	39.81	47.13
official evaluation(phrase-level)			
biomedical	78.7	76.22	77.44

Table 6: Results for in-domain hedge cue identification

4.3.1 Results analysis

We investigate the weights of internal features and found that the words, which have no uncertainty information, also play the significant roles to predict the uncertainty of the sentence.

Intuitively, the words without uncertainty information should just have negligible effect and the corresponding features should have low weights. However, this ideal case is difficult to reached by learning algorithm due to the sparsity of data.

In Table 4, we list the top 10 words involved in features with the largest weights for each category. These words are ranked by the accumulative scores of their related features.

In Table 5, we list the top 10 POS involved in features with the largest weight for each category.

4.4 Hedge cue identification

Hedge cues identification is one module for the second subtask, we also analyze the performance on this module.

Since we treat this problem as a binary classification problem, we evaluate F-measure of hedge cue words. The results are listed in Table 6.

We have to point out that our evaluation is based on word while official evaluation is based on phrase. That means our results would seem to be higher than the official results, especially on Wikipedia dataset because average length of hedge cues in Wikipedia dataset is more than 2 words.

4.4.1 Result Analysis

We classify the results into four categories: false negative, false positive, true positive and true negative. We found that most mistakes are made because of polysemy and collocation.

In Table 7, we list top 10 words for each category. For the false results, the words are difficult to distinguish without its context in the corresponding sentence.

4.5 Scopes recognition

For recognizing the in-sentence scopes, F-measure is also used to evaluate the performance of the word-cue pair classifier. The results using gold hedge cues are shown in Table 8. From the results, F-measure achieves respectively 70.44% and 75.94% when individually using the base context features extracted by 12 predefined patterns (see Figure 1) and syntactic dependency features (see Figure 3), while 79.55% when using all features.

The results imply that syntactic dependency

biomedical		Wikipedia		cross domain	
uncertain	certain	uncertain	certain	uncertain	certain
whether	show	probably	the other	suggest	show
may	demonstrate	some	often	whether	used to
suggest	will	many	patients	probably	was
likely	can	one of	another	indicate	CFS
indicate	role	believed	days	appear	demonstrate
possible	found	possibly	CFS	putative	the other
putative	human	considered	are	some	of all
appear	known to	such as	any other	thought	':'
thought	report	several	western	possibly	people
potential	evidence	said to	pop	likely	could not

Table 4: Top 10 significant words in detecting uncertain sentences

biomedical		Wikipedia		cross domain	
uncertain	certain	uncertain	certain	uncertain	certain
MD	SYM	RB	VBZ	JJS	SYM
VBG	PRP	JJS	CD	RBS	':'
VB	NN	RBS	':'	RB	JJR
VBZ	CD	FW	WRB	EX	WDT
IN	WDT	VBP	PRP	CC	CD

Table 5: Top 5 significant POS in detecting uncertain sentences

Dataset	Precision	Recall	F1
base context features			
biomedical	66.04	75.48	70.44
syntactic dependency features			
biomedical	93.77	63.05	75.94
all features			
biomedical	78.72	80.41	79.55

Table 8: Results for scopes recognizing with gold hedge cues (word-level)

features contribute more benefits to recognize scopes than surface context features.

Official results evaluated at block level are also listed in Table 9.

dataset	Precision	Recall	F1
biomedical	21.87	17.23	19.27

Table 9: Official results for scopes recognizing (block level)

From Table 9 and the official result on hedge cue identification in Table 6, we believe that our post-processing strategy would be responsible for the low performance on recognizing scopes. Our strategy is to find a maximum consecutive block covering the corresponding hedge cue. This strategy cannot do well with the complex scope structure. For example, a scope is covered by another scope. Therefore, our system would generate a block covering all hedge cues if there exists more than one hedge cues in a sentence.

5 Conclusion

We present our implemented system for CoNLL-2010 Shared Task in this paper. We introduce syntactic dependency features when recognizing hedge scopes and employ average perceptron algorithm to train the models. On the biomedical corpus, our system achieves F-measure with 77.86% in detecting uncertain sentences, 77.44% in recognizing hedge cues, and 19.27% in identifying the scopes.

Although some results are low and beyond our expectations, we believe that our system can be at least improved within the following fields. Firstly, we would experiment on other kinds of features, such as chunk or named entities in biomedical. Secondly, the unlabeled datasets would be explored in the future.

False Negative	False Positive	True Positive	True Negative
support	considered	suggesting	chemiluminescence
of	potential	may	rhinitis
demonstrate	or	proposal	leukemogenic
a	hope	might	ribosomal
postulate	indicates	indicating	bp
supports	expected	likely	nc82
good	can	appear	intronic/exonic
advocates	should	possible	large
implicated	either	speculate	allele
putative	idea	whether	end

Table 7: Top 10 words with the largest scores for each category in hedge cue identification

Acknowledgments

This work was funded by Chinese NSF (Grant No. 60673038), 973 Program (Grant No. 2010CB327906, and Shanghai Committee of Science and Technology(Grant No. 09511501404).

References

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, pages 1–8. Association for Computational Linguistics.

Richard O. Duda, Peter E. Hart, and David G. Stork. 2001. *Pattern classification*. Wiley, New York.

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. Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the ACL*, pages 91–98. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Ben Medlock. 2008. Exploring hedge identification in biomedical literature. *Journal of Biomedical Informatics*, 41(4):636–654.

Roser Morante and Walter Daelemans. 2009. Learning the scope of hedge cues in biomedical texts. In *Proceedings of the Workshop on BioNLP*, pages 28–36. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics.