Using Mined Coreference Chains as a Resource for a Semantic Task

Heike Adel and Hinrich Schütze
Center for Information and Language Processing
University of Munich
Germany
heike.adel@cis.lmu.de

Abstract

We propose to use coreference chains extracted from a large corpus as a resource for semantic tasks. We extract three million coreference chains and train word embeddings on them. Then, we compare these embeddings to word vectors derived from raw text data and show that coreference-based word embeddings improve F_1 on the task of antonym classification by up to .09.

1 Introduction

After more than a decade of work on coreference resolution, coreference resolution systems have reached a certain level of maturity (e.g., Recasens et al. (2010)). While accuracy is far from perfect and many phenomena such as bridging still pose difficult research problems, the quality of the output of these systems is high enough to be useful for many applications.

In this paper, we propose to run coreference resolution systems on large corpora, to collect the coreference chains found and to use them as a resource for solving semantic tasks. This amounts to using mined coreference chains as an automatically compiled resource similar to the way cooccurrence statistics, dependency pairs and aligned parallel corpora are used in many applications in NLP. Coreference chains have interesting complementary properties compared to these other resources. For example, it is difficult to distinguish true semantic similarity (e.g., "cows" – "cattle") from mere associational relatedness (e.g., "cows" – "milk") based on cooccurrence statistics. In contrast, coreference chains should be able to make that distinction since only "cows" and "cattle" can occur in the same coreference chain, not "cows" and "milk".

As a proof of concept we compile a resource of mined coreference chains from the Gigaword corpus and apply it to the task of identifying antonyms. We induce distributed representations for words based on (i) cooccurrence statistics and (ii) mined coreference chains and show that a combination of both outperforms cooccurrence statistics on antonym identification.

In summary, we make two contributions. First, we propose to use coreference chains mined from large corpora as a resource in NLP and publish the first such resource. Second, in a proof of concept study, we show that they can be used to solve a semantic task – antonym identification – better than is possible with existing resources.

We focus on the task of finding antonyms in this paper since antonyms usually are distributionally similar but semantically dissimilar words. Hence, it is often not possible to distinguish them from synonyms with distributional models only. In contrast, we expect that the coreference-based representations can provide useful complementary information to this task. In general, coreference-based similarity can however be used as an additional feature for any task that distributional similarity is useful for. Thus, our coreference resource can be applied to a variety of NLP tasks, e.g. finding alternative names for entities (in a way similar to Wikipedia anchors) for tasks in the context of knowledge base population.

The remainder of the paper is organized as follows. In Section 2, we describe how we create word embeddings and how our antonym classifier works. The word embeddings are then evaluated qualitatively, quantitatively and for the task of antonym detection (Section 3). Section 4 discusses related work and Section 5 concludes.

2 System description

2.1 Coreference-based embeddings

Standard word embeddings derived from text data may not be able to distinguish between semantic

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1447–1452,
October 25-29, 2014, Doha, Qatar.

	text-based	coref.-based
his	my, their, her, your, our	he, him, himself, zechariah, ancestor
woman	man, girl, believer, pharisee, guy	girl, prostitute, lupita, betsy, lehia

Table 1: Nearest neighbors of "his" / "woman" for text-based & coreference-based embeddings

association and true synonymy. As a result, synonyms and antonyms may be mapped to similar word vectors (Yih et al., 2012). For many NLP tasks, however, information about true synonymy or antonymy may be important.

In this paper, we develop two different word embeddings: embeddings calculated on raw text data and embeddings derived from automatically extracted coreference chains. For the calculation of the vector representations, the word2vec toolkit[1] by Mikolov et al. (2013) is applied. We use the skip-gram model for our experiments because its results for semantic similarity are better according to Mikolov et al. (2013). We train a first model on a subset of English Gigaword data.[2] In the following sections, we call the resulting embeddings *text-based*. To improve the semantic similarities of the vectors, we prepare another training text consisting of coreference chains. We use CoreNLP (Lee et al., 2011) to extract coreference chains from the Gigaword corpus. Then we build a skip-gram model on these coreference chains. The extracted coreference chains are provided as an additional resource to this paper[3]. Although they have been developed using only a publicly available toolkit, we expect this resource to be helpful for other researchers since the process to extract the coreference chains of such a large text corpus takes several weeks on multi-core machines. In total, we extracted 3.1M coreference chains. 2.7M of them consist of at least two different markables. The median (mean) length of the chains is 3 (4.0) and the median (mean) length of a markable is 1 (2.7). To train word embeddings, the markables of each coreference chain are concatenated to one text line. These lines are used as input sentences for word2vec. We refer to the resulting embeddings as *coreference-based*.

2.2 Antonym detection

In the following experiments, we use word embeddings to discriminate antonyms from non-antonyms. We formalize this as a supervised classification task and apply SVMs (Chang and Lin, 2011).

The following features are used to represent a pair of two words w and v:

1. cosine similarity of the text-based embeddings of w and v;
2. inverse rank of v in the nearest text-based neighbors of w;
3. cosine similarity of the coreference-based embeddings of w and v;
4. inverse rank of v in the nearest coreference-based neighbors of w;
5. difference of (1) and (3);
6. difference of (2) and (4).

We experiment with three different subsets of these features: text-based (1 and 2), coreference-based (3 and 4) and all features.

3 Experiments and results

3.1 Qualitative analysis of word vectors

Table 1 lists the five nearest neighbors based on cosine similarity of text-based and coreference-based word vectors for "his" and "woman".

We see that the two types of embeddings capture different notions of similarity. Unlike the text-based neighbors, the coreference-based neighbors have the same gender. The text-based neighbors are mutually substitutable words, but substitution seems to change the meaning more than for the coreference-based neighbors.

In Figure 1, we illustrate the vectors for some antonyms (connected by lines).

For reducing the dimensionality of the vector space to 2D, we applied the t-SNE toolkit[4]. It uses stochastic neighbor embedding with a Student's t-distribution to map high dimensional vectors into a lower dimensional space (Van der Maaten and Hinton, 2008). The Figure shows that the coreference-based word embeddings are able to

[1] https://code.google.com/p/word2vec

[2] LDC2012T21, Agence France-Presse 2010

[3] https://code.google.com/p/cistern

[4] http://homepage.tudelft.nl/19j49/t-SNE.html

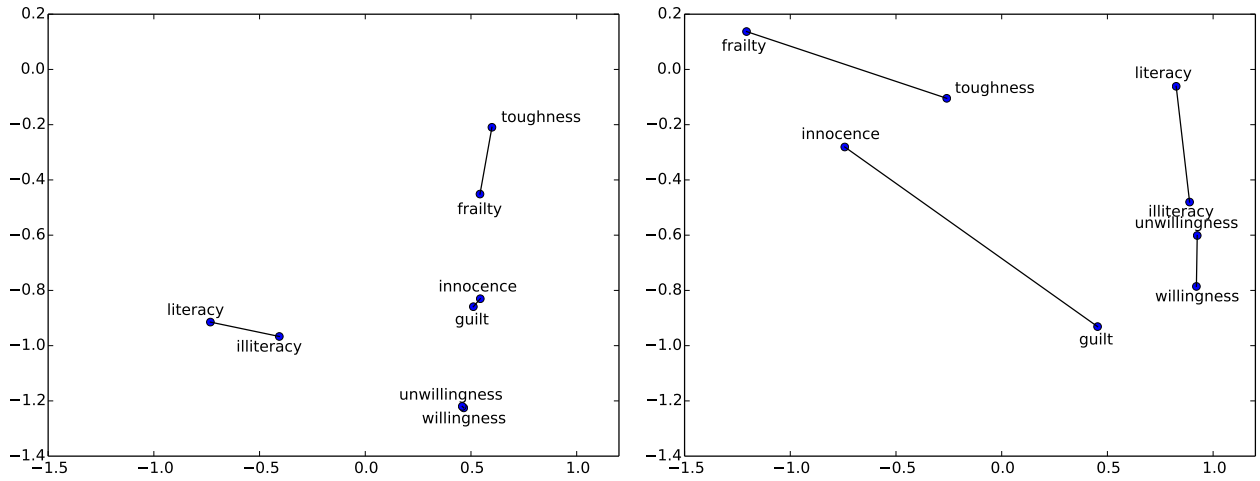

Figure 1: 2D-positions of words in the text-based (top) and coreference-based embeddings (bottom)

enlarge the distance between antonyms (especially for *guilt* vs. *innocence* and *toughness* vs. *frailty*) compared to text-based word vectors.

3.2 Quantitative analysis of word vectors

To verify that coreference-based embeddings better represent semantic components relevant to coreference, we split our coreference resource into two parts (about 85% and 15% of the data), trained embeddings on the first part and computed the cosine similarity – both text-based and coreference-based – for each pair of words occurring in the same coreference chain in the second part. The statistics in Table 2 confirm that coreference-based vectors have higher similarity within chains than text-based vectors.

3.3 Experimental setup

We formalize antonym detection as a binary classification task. Given a target word w and one of its nearest neighbors v, the classifier decides whether v is an antonym of w. Our data set is a set of pairs, each consisting of a target word w and a candidate v. For all word types of our vocabulary, we search for antonyms using the online dictionary Merriam Webster.[5] The resulting list is provided as an additional resource[6]. It contains 6225 words with antonyms. Positive training examples are collected by checking if the 500 nearest text-based neighbors of w contain one of the antonyms listed by Webster. Negative training examples are created by replacing the antonym with a random word from the 500 nearest neighbors that is not listed as an antonym. By selecting both the positive and the negative examples from the nearest neighbors of the word vectors, we intend to develop a task which is hard to solve: The classifier has to find the small portion of semantically dissimilar words (i.e., antonyms) among distributionally very similar words. The total number of positive and negative examples is 2337 each. The data are split into training (80%), development (10%) and test (10%) sets.

In initial experiments, we found only a small difference in antonym classification performance between text-based and coreference-based features. When analyzing the errors, we realized that our rationale for using coreference-based embeddings only applies to nouns, not to other parts of speech. This will be discussed in detail below. We therefore run our experiments in two modes: *all word classification* (all pairs are considered) and *noun classification* (only pairs are considered for which the target word is a noun). We use the Stanford part-of-speech tagger (Toutanova et al., 2003) to determine whether a word is a noun or not.

Our classifier is a radial basis function (rbf) support vector machine (SVM). The rbf kernel performed better than a linear kernel in initial experiments. The SVM parameters C and γ are optimized on the development set. The representation of target-candidate pairs consists of the features described in Section 2.

3.4 Experimental results and discussion

We perform the experiments with the three different feature sets described in Section 2: text-based, coreference-based and all features. Table 3 shows

[5] http://www.merriam-webster.com

[6] https://code.google.com/p/cistern

	all word classification						noun classification					
	development set			test set			development set			test set		
feature set	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
text-based	**.83**	**.66**	**.74**	.74	.55	.63	**.91**	.61	.73	.74	.51	.60
coreference-based	.67	.42	.51	.65	.43	.52	.86	.47	.61	.77	.45	.57
text+coref	.79	.65	.72	**.75**	**.58**	**.66**	.88	**.70**	**.78**	**.79**	**.61**	**.69**

Table 3: Results for different feature sets. Best result in each column in bold.

	minimum	maximum	median
text-based vectors	-0.350	0.998	0.156
coref.-based vectors	-0.318	0.999	0.161

Table 2: Cosine similarity of words in the same coreference chain

results for development and test sets.

For all word classification, coreference-based features do not improve performance on the development set (e.g., F_1 is .74 for text-based vs .72 for text+coref). On the test set, however, the combination of all features (text+coref) has better performance than text-based alone: .66 vs .63.

For noun classification, using coreference-based features in addition to text-based features improves results on development set (F_1 is .78 vs .73) and test set (.69 vs .60).

These results show that mined coreference chains are a useful resource and provide information that is complementary to other methods. Even though adding coreference-based embeddings improves performance on antonym classification, the experiments also show that using only coreference-based embeddings is almost always worse than using only text-based embeddings. This is not surprising given that the amount of training data for the word embeddings is different in the two cases. Coreference chains provide only a small subset of the word-word relations that are given to the word2vec skip-gram model when applied to raw text. If the sizes of the training data sets were similar in the two cases, we would expect performance to be comparable.

In the beginning, our hypothesis was that coreference information should be helpful for antonym classification in general. When we performed an error analysis for our initial results, we realized that this hypothesis only holds for nouns. Other types of words cooccurring in coreference chains are not more likely to be synonyms than words cooccurring in text windows. Two contexts that illustrate this point are "*bright sides*, but also *difficult and dark ones*" and "a series of *black and white shots*" (elements of coreference chains in italics). Thus, adjectives with opposite meanings can cooccur in coreference chains just as they can cooccur in window-based contexts. For nouns, it is much less likely that the same coreference chain will contain both a noun and its antonym since – by definition – markables in a coreference chain refer to the same identical entity.

4 Related work

Traditionally, words have been represented by vectors of the size of the vocabulary with a one at the word index and zeros otherwise (one-hot vectors). However, this approach cannot handle unknown words (Turian et al., 2010) and similarities among words cannot be represented (Mikolov et al., 2013). Therefore, distributed word representations (embeddings) become more and more popular. They are low-dimensional, real-valued vectors. Mikolov et al. (2013) have published word2vec, a toolkit that provides different possibilities to estimate word embeddings (cbow model and skip-gram model). They show that the resulting word vectors capture semantic and syntactic relationships of words. Baroni et al. (2014) show that word embeddings are able to outperform count based word vectors on a variety of NLP tasks. Recently, Levy and Goldberg (2014) have generalized the skip-gram model to include not only linear but arbitrary contexts like contexts derived from dependency parse trees. Andreas and Klein (2014) investigate the amount of additional information continuous word embeddings could add to a constituency parser and find that most of their information is redundant to what can be learned from labeled parse trees. In (Yih et al., 2012), the vector space representation of words is modified so that high positive similarities are assigned to synonyms and high negative similarities to antonyms. For this, latent semantic analysis is applied to a matrix of thesaurus entries. The val-

ues representing antonyms are negated.

There has been a great deal of work on applying the vector space model and cosine similarity to find synonyms or antonyms. Hagiwara et al. (2006) represent each word as a vector with cooccurrence frequencies of words and contexts as elements, normalized by the inverse document frequency. The authors investigate three types of contextual information (dependency, sentence cooccurrence and proximity) and find that a combination of them leads to the most stable results. Schulte im Walde and Köper (2013) build a vector space model on lexico-syntactic patterns and apply a Rocchio classifier to distinguish synonyms from antonyms, among other tasks. Van der Plas and Tiedemann (2006) use automatically aligned translations of the same text in different languages to build context vectors. Based on these vectors, they detect synonyms.

In contrast, there are also studies using linguistic knowledge from external resources: Senellart and Blondel (2008) propose a method for synonym detection based on graph similarity in a graph generated using the definitions of a monolingual dictionary. Harabagiu et al. (2006) recognize antonymy by generating antonymy chains based on WordNet relations. Mohammad et al. (2008) look for the word with the highest degree of antonymy to a given target word among five candidates. For this task, they use thesaurus information and the similarity of the contexts of two contrasting words. Lin et al. (2003) use Hearst patterns to distiguish synonyms from antonyms. Work by Turney (2008) is similar except that the patterns are learned.

Except for the publicly available coreference resolution system, our approach does not need external resources such as dictionaries or bilingual corpora and no human labor is required. Thus, it can be easily applied to any corpus in any language as long as there exists a coreference resolution system in this language. The pattern-based approach (Lin et al., 2003; Turney, 2008) discussed above also needs few resources. In contrast to our work, it relies on patterns and might therefore restrict the number of recognizable synonyms and antonyms to those appearing in the context of the pre-defined patterns. On the other hand, patterns could explicitely distinguish contexts typical for synonyms from contexts for antonyms. Hence, we plan to combine our coreference-based method with pattern-based methods in the future.

5 Conclusion

In this paper, we showed that mined coreference chains can be used for creating word embeddings that capture a type of semantic similarity that is different from the one captured by standard text-based embeddings. We showed that coreference-based embeddings improve performance of antonym classification by .09 F_1 compared to using only text-based embeddings. We achieved precision values of up to .79, recall values of up to .61 and F_1 scores of up to .69.

Acknowledgments

This work was supported by DFG (grant SCHU 2246/4-2).

References

Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax? In *ACL*, pages 822–827.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *ACL*, pages 238–247.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27.

Masato Hagiwara, Yasuhiro Ogawa, and Katsuhiko Toyama. 2006. Selection of effective contextual information for automatic synonym acquisition. In *COLING/ACL*, pages 353–360.

Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. 2006. Negation, contrast and contradiction in text processing. In *AAAI*, volume 6, pages 755–762.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *CoNLL: Shared Task*, pages 28–34.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *ACL*, pages 302–308.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *IJCAI*, pages 1492–1493.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Workshop at ICLR*.

Saif Mohammad, Bonnie Dorr, and Graeme Hirst. 2008. Computing word-pair antonymy. In *EMNLP*, pages 982–991.

Marta Recasens, Lluís Màrquez, Emili Sapena, M Antònia Martí, Mariona Taulé, Véronique Hoste, Massimo Poesio, and Yannick Versley. 2010. Semeval-2010 task 1: Coreference resolution in multiple languages. In *5th International Workshop on Semantic Evaluation*, pages 1–8.

Sabine Schulte im Walde and Maximilian Köper. 2013. Pattern-based distinction of paradigmatic relations for German nouns, verbs, adjectives. In *Language Processing and Knowledge in the Web*, pages 184–198. Springer.

Pierre Senellart and Vincent D Blondel. 2008. Automatic discovery of similar words. In *Survey of Text Mining II*, pages 25–44. Springer.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL-HLT*, pages 252–259.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*, pages 384–394.

Peter D. Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *COLING*, pages 905–912.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(11):2579–2605.

Lonneke Van der Plas and Jörg Tiedemann. 2006. Finding synonyms using automatic word alignment and measures of distributional similarity. In *COLING/ACL*, pages 866–873.

Wen-tau Yih, Geoffrey Zweig, and John C Platt. 2012. Polarity inducing latent semantic analysis. In *EMNLP/CoNLL*, pages 1212–1222.