Generalizing Dependency Features for Opinion Mining

Mahesh Joshi[1] and **Carolyn Penstein-Rosé**[1,2]
[1]Language Technologies Institute
[2]Human-Computer Interaction Institute
Carnegie Mellon University, Pittsburgh, PA, USA
{maheshj,cprose}@cs.cmu.edu

Abstract

We explore how features based on syntactic dependency relations can be utilized to improve performance on opinion mining. Using a transformation of dependency relation triples, we convert them into "composite back-off features" that generalize better than the regular lexicalized dependency relation features. Experiments comparing our approach with several other approaches that generalize dependency features or ngrams demonstrate the utility of composite back-off features.

1 Introduction

Online product reviews are a crucial source of opinions about a product, coming from the people who have experienced it first-hand. However, the task of a potential buyer is complicated by the sheer number of reviews posted online for a product of his/her interest. Opinion mining, or sentiment analysis (Pang and Lee, 2008) in product reviews, in part, aims at automatically processing a large number of such product reviews to identify opinionated statements, and to classify them into having either a positive or negative polarity.

One of the most popular techniques used for opinion mining is that of supervised machine learning, for which, many different lexical, syntactic and knowledge-based feature representations have been explored in the literature (Dave et al., 2003; Gamon, 2004; Matsumoto et al., 2005; Ng et al., 2006). However, the use of syntactic features for opinion mining has achieved varied results. In our work, we show that by altering syntactic dependency relation triples in a particular way (namely, "backing off" only the head word in a dependency relation to its part-of-speech tag), they generalize better and yield a significant improvement on the task of identifying opinions from product reviews. In effect, this work demonstrates a better way to utilize syntactic dependency relations for opinion mining.

In the remainder of the paper, we first discuss related work. We then motivate our approach and describe the composite back-off features, followed by experimental results, discussion and future directions for our work.

2 Related Work

The use of syntactic or deep linguistic features for opinion mining has yielded mixed results in the literature so far. On the positive side, Gamon (2004) found that the use of deep linguistic features extracted from phrase structure trees (which include syntactic dependency relations) yield significant improvements on the task of predicting satisfaction ratings in customer feedback data. Matsumoto et al. (2005) show that when using frequently occurring sub-trees obtained from dependency relation parse trees as features for machine learning, significant improvement in performance is obtained on the task of classifying movie reviews as having positive or negative polarity. Finally, Wilson et al. (2004) use several different features extracted from dependency parse trees to improve performance on the task of predicting the strength of opinion phrases.

On the flip side, Dave et al. (2003) found that for the task of polarity prediction, adding *adjective-noun* dependency relationships as features does not provide any benefit over a simple bag-of-words based feature space. Ng et al. (2006) proposed that rather than focusing on just *adjective-noun* relationships, the *subject-verb* and *verb-object* relationships should also be considered for polarity classification. However, they observed that the addition of these dependency relationships does not improve performance over a feature space that includes unigrams, bigrams and trigrams.

One difference that seems to separate the successes from the failures is that of using the entire set of dependency relations obtained from a dependency parser and allowing the learning algorithm to generalize, rather than picking a small subset of dependency relations manually. However, in such a situation, one critical issue might be the sparseness of the very specific linguistic features, which may cause the classifier learned from such features to not generalize. Features based on dependency relations provide a nice way to enable generalization to the right extent through utilization of their structural aspect. In the next section, we motivate this idea in the context of our task, from a linguistic as well as machine learning perspective.

3 Identifying Opinionated Sentences

We focus on the problem of automatically identifying whether a sentence in a product review contains an opinion about the product or one of its features. We use the definition of this task as formulated by Hu and Liu (2004) on Amazon.com and CNet.com product reviews for five different products. Their definition of an opinion sentence is reproduced here verbatim: *"If a sentence contains one or more product features and one or more opinion words, then the sentence is called an opinion sentence."* Any other sentence in a review that does not fit the above definition of an opinion sentence is considered as a non-opinion sentence. In general, these can be expected to be verifiable statements or facts such as product specifications and so on.

Before motivating the use of dependency relations as features for our task, a brief overview about dependency relations follows.

3.1 Dependency Relations

The dependency parse for a given sentence is essentially a set of triplets or triples, each of which is composed of a grammatical relation and the pair of words from the sentence among which the grammatical relation holds ($\{rel_i, w_j, w_k\}$, where rel_i is the dependency relation among words w_j and w_k). The set of dependency relations is specific to a given parser – we use the Stanford parser[1] for computing dependency relations. The word w_j is usually referred to as the *head word* in the dependency triple, and the word w_k is usually referred to as the *modifier word*.

One straightforward way to use dependency relations as features for machine learning is to generate features of the form `RELATION_HEAD_MODIFIER` and use them in a standard bag-of-words type binary or frequency-based representation. The indices of the head and modifier words are dropped for the obvious reason that one does not expect them to generalize across sentences. We refer to such features as *lexicalized dependency relation features*.

3.2 Motivation for our Approach

Consider the following examples (these are made-up examples for the purpose of keeping the discussion succinct, but still capture the essence of our approach):

(i) *This is a great camera!*

(ii) *Despite its few negligible flaws, this really great mp3 player won my vote.*

Both of these sentences have an adjectival modifier (`amod`) relationship, the first one having `amod_camera_great)` and the second one having `amod_player_great)`. Although both of these features are good indicators of opinion sentences and are closely related, any machine learning algorithm that treats these features independently will not be able to generalize their relationship to the opinion class. Also, any new test sentence that contains a noun different from either "camera" or "player" (for instance in the review of a different electronic product), but is participating in a similar relationship, will not receive any importance in favor of the opinion class – the machine learning algorithm may not have even seen it in the training data.

Now consider the case where we "back off" the head word in each of the above features to its part-of-speech tag. This leads to a single feature: `amod_NN_great`. This has two advantages: first, the learning algorithm can now learn a weight for a more general feature that has stronger evidence of association with the opinion class, and second, any new test sentence that contains an unseen noun in a similar relationship with the adjective "great" will receive some weight in favor of the opinion class. This "back off" operation is a generalization of the regular lexicalized dependency relations mentioned above. In the next section we describe all such generalizations that we experimented with.

[1] `http://nlp.stanford.edu/software/lex-parser.shtml`

4 Methodology

Composite Back-off Features: The idea behind our composite back-off features is to create more generalizable, but not overly general back-off features by backing off to the part-of-speech (POS) tag of either the head word or the modifier word (but not both at once, as in Gamon (2004) and Wilson et al. (2004)) – hence the description "composite," as there is a lexical part to the feature, coming from one word, and a POS tag coming from the other word, along with the dependency relation itself.

The two types of composite back-off features that we create from lexicalized dependency triples are as follows:

(i) **h-bo**: Here we use features of the form $\{rel_i, POS_j, w_k\}$ where the head word is replaced by its POS tag, but the modifier word is retained.

(ii) **m-bo**: Here we use features of the form $\{rel_i, w_j, POS_k\}$, where the modifier word is replaced by its POS tag, but the head word is retained.

Our hypothesis is that the **h-bo** features will perform better than purely lexicalized dependency relations for reasons mentioned in Section 3.2 above. Although **m-bo** features also generalize the lexicalized dependency features, in a relation such as an adjectival modifier (discussed in Section 3.2 above), the head noun is a better candidate to back-off for enabling generalization across different products, rather than the modifier adjective. For this reason, we do not expect their performance to be comparable to **h-bo** features.

We compare our composite back-off features with other similar ways of generalizing dependency relations and lexical ngrams that have been tried in previous work. We describe these below.

Full Back-off Features: Both Gamon (2004) and Wilson et al. (2004) utilize features based on the following version of dependency relationships: $\{rel_i, POS_j, POS_k\}$, where they "back off" both the head word and the modifier word to their respective POS tags (POS_j and POS_k). We refer to this as **hm-bo**.

NGram Back-off Features: Similar to McDonald et al. (2007), we utilize backed-off versions of lexical bigrams and trigrams, where all possible combinations of the words in the ngram are replaced by their POS tags, creating features such as $w_j\_POS_k$, $POS_j\_w_k$, $POS_j\_POS_k$ for each lexical bigram and similarly for trigrams. We refer to these as **bi-bo** and **tri-bo** features respectively.

In addition to these back-off approaches, we also use regular lexical bigrams (**bi**), lexical trigrams (**tri**), POS bigrams (**POS-bi**), POS trigrams (**POS-tri**) and lexicalized dependency relations (**lexdep**) as features. While testing all of our feature sets, we evaluate each of them individually by adding them to the basic set of unigram (**uni**) features.

5 Experiments and Results

Details of our experiments and results follow.

5.1 Dataset

We use the extended version of the Amazon.com / CNet.com product reviews dataset released by Hu and Liu (2004), available from their web page[2]. We use a randomly chosen subset consisting of 2,200 review sentences (200 sentences each for 11 different products)[3]. The distribution is 1,053 (47.86%) opinion sentences and 1,147 (52.14%) non-opinion sentences.

5.2 Machine Learning Parameters

We have used the Support Vector Machine (SVM) learner (Shawe-Taylor and Cristianini, 2000) from the MinorThird Toolkit (Cohen, 2004), along with the χ-squared feature selection procedure, where we reject features if their χ-squared score is not significant at the 0.05 level. For SVM, we use the default linear kernel with all other parameters also set to defaults. We perform 11-fold cross-validation, where each test fold contains all the sentences for one of the 11 products, and the sentences for the remaining ten products are in the corresponding training fold. Our results are reported in terms of average accuracy and Cohen's kappa values across the 11 folds.

5.3 Results

Table 1 shows the full set of results from our experiments. Our results are comparable to those reported by Hu and Liu (2004) on the same task; as well as those by Arora et al. (2009) on a similar task of identifying qualified vs. bald claims in product reviews. On the accuracy metric, the composite features with the head word backed off

[2]`http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html`

[3]`http://www.cs.cmu.edu/~maheshj/datasets/acl09short.html`

Features	**Accuracy**	**Kappa**
uni	.652 ($\pm$.048)	.295 ($\pm$.049)
uni+bi	.657 ($\pm$.066)	.304 ($\pm$.089)
uni+bi-bo	.650 ($\pm$.056)	.299 ($\pm$.079)
uni+tri	.655 ($\pm$.062)	.306 ($\pm$.077)
uni+tri-bo	.647 ($\pm$.051)	.287 ($\pm$.075)
uni+POS-bi	.676 ($\pm$.057)	**.349 ($\pm$.083)**
uni+POS-tri	.661 ($\pm$.050)	.317 ($\pm$.064)
uni+lexdep	.639 ($\pm$.055)	.268 ($\pm$.079)
uni+hm-bo	.670 ($\pm$.046)	.336 ($\pm$.065)
uni+h-bo	**.679 ($\pm$.063)**	**.351 ($\pm$.097)**
uni+m-bo	.657 ($\pm$.056)	.308 ($\pm$.063)

Table 1: Shown are the average accuracy and Cohen's kappa across 11 folds. Bold indicates statistically significant improvements ($p < 0.05$, two-tailed pairwise T-test) over the (**uni**) baseline.

are the only ones that achieve a statistically significant improvement over the **uni** baseline. On the kappa metric, using POS bigrams also achieves a statistically significant improvement, as do the composite **h-bo** features. None of the other back-off strategies achieve a statistically significant improvement over **uni**, although numerically **hm-bo** comes quite close to **h-bo**. Evaluation of these two types of features by themselves (without unigrams) shows that **h-bo** are significantly better than **hm-bo** at $p < 0.10$ level. Regular lexicalized dependency relation features perform worse than unigrams alone. These results thus demonstrate that composite back-off features based on dependency relations, where only the head word is backed off to its POS tag present a useful alternative to encoding dependency relations as features for opinion mining.

6 Conclusions and Future Directions

We have shown that for opinion mining in product review data, a feature representation based on a simple transformation ("backing off" the head word in a dependency relation to its POS tag) of syntactic dependency relations captures more generalizable and useful patterns in data than purely lexicalized dependency relations, yielding a statistically significant improvement.

The next steps that we are currently working on include applying this approach to polarity classification. Also, the aspect of generalizing features across different products is closely related to fully supervised domain adaptation (Daumé III, 2007), and we plan to combine our approach with the idea from Daumé III (2007) to gain insights into whether the composite back-off features exhibit different behavior in domain-general versus domain-specific feature sub-spaces.

Acknowledgments

This research is supported by National Science Foundation grant IIS-0803482.

References

Shilpa Arora, Mahesh Joshi, and Carolyn Rosé. 2009. Identifying Types of Claims in Online Customer Reviews. In *Proceedings of NAACL 2009*.

William Cohen. 2004. Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data.

Hal Daumé III. 2007. Frustratingly Easy Domain Adaptation. In *Proceedings of ACL 2007*.

Kushal Dave, Steve Lawrence, and David Pennock. 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In *Proceedings of WWW 2003*.

Michael Gamon. 2004. Sentiment Classification on Customer Feedback Data: Noisy Data, Large Feature Vectors, and the Role of Linguistic Analysis. In *Proceedings of COLING 2004*.

Minqing Hu and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *Proceedings of ACM SIGKDD 2004*.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment Classification Using Word Sub-sequences and Dependency Sub-trees. In *Proceedings of the 9th PAKDD*.

Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. 2007. Structured Models for Fine-to-Coarse Sentiment Analysis. In *Proceedings of ACL 2007*.

Vincent Ng, Sajib Dasgupta, and S. M. Niaz Arifin. 2006. Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews. In *Proceedings of the COLING/ACL 2006*.

Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 2(1–2).

John Shawe-Taylor and Nello Cristianini. 2000. *Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. 2004. Just How Mad Are You? Finding Strong and Weak Opinion Clauses. In *Proceedings of AAAI 2004*.