Learning to Negate Adjectives with Bilinear Models

First Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Second Author
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
email@domain

Abstract

TODO

1 Introduction

Learning the meaning of closed-class function words, such as negation, coordination, and quantification, is important for compositional distributional semantics. Unlike most open-class lexical items, function words have an operator semantics, transforming the meaning of words and phrases over which they take scope. In this paper we focus on negation, and the task of learning a function that maps a word to an approximate "opposite". This is a challenging task because related words of opposite polarity are close to each other in most of the commonly-used word embedding models.

We distinguish the task of learning a negation mapping from the task of retrofitting vectors for lexical contrast (**?**; **?**), or adding lexical contrast into word embeddings as they are learned (Pham et al., 2015; **?**). Instead, we assume that we are given a set of word embeddings with no particular characteristics to aid in identifying opposites.

We focus on the relatively simple case of adjectives, where negation can be approximated by antonymy. For example, given the expression *not talkative* and the word vector for *talkative*, we would like the function to return a word from the set *quiet, taciturn*, etc. This is only an approximation: it's well-known that e.g. *not hot* doesn't necessarily mean *cold*, but could be *cool, tepid, warm* etc. depending on the context. [TODO: Grefenstette, Baroni, maybe a linguistic cite] However, for adjectives it is a reasonable approximation and this is where we begin. This puts our task with the general area of antonym detection.

The intuition behind our approach is to exploit the semantic neighbourhood of each adjective. The features that need adjusting to negate a temperature adjective may differ from those that need adjusting to negate an emotion adjective. [TODO: Santus and Baroni cites] By using a bilinear relational neural network architecture used to identify transformations in computer vision, we are able to learn the relevant function. Our model outperforms other variants on a GRE multiple choice taks, and more importantly learns to produce a one-best antonym with high precision.

2 Relational Encoders

2.1 Relational Autoencoders

In computer vision, relational autoencoders (RAE), also known as gated autoencoders (GAE), are used to identify images that correspond modulo a relation such as rotation or translation (Memisevic). In an RAE, the original image is reconstructed with regard to a related image. The hidden layer learns to detect features of the relation. The "gating" of one image vector by another allows the feature detectors to concentrate on the correspondences between the related images, rather than being distracted by the differences between untransformed images.The key is bilinear connections; there is a weight for every pair of units in the input vector and "gate" vector.

We note that gates in an RAE are NOT the same as gates in LSTMs and other gated recurrent networks. We will use "gate" in quotation marks to make this clear.

Figure 1(a) shows an example for vision. As standard for autoencoders, encoder and decoder weights are usually tied, although untied weights can be used. [TODO: cite Alain and Olivier 2013]

[TODO: RAE equations here]

Rudy and Taylor 2015 introduce a class-

[TODO: Figure]

Figure 1: Neural network architectures for RAE, CCRAE, CCCURE.

conditional autoencoder in which the "gate" is a one-hot class label, essentially training one autoencoder per class but with weight sharing across classes.

2.2 Continuous Class-Conditional Untied Relational Autoencoders

We introduce continuous class-conditional untied relational encoders (CCCURE). The architecture is essentially an RAE with untied encoder and decoder weights. However, this is not an autoencoder since it is not trained to reproduce the input, but to transform it to its opposite. Additionally, instead of one-hot class representations as appropriate for problems with a small number of classes, our "gates" are continuous representations of the semantic neighbourhood of the input word. We call this "gate" a neutral context vector. Figure 1 shows our architecture in comparison with Memisevic and R&T.

[TODO: CCCURE equations here]

3 Experiments

3.1 Models

We compare the CCCURE with (1) a simple feed-forward network that uses no neutral context "gate". (2) a feed-forward network where the input consists of the word vector concatenated with the "gate" vector. The network therefore has information about the semantic network and can relate the word and the "gate" by way of the connections to the hidden layer, but has no bilinear connections. (3) auto-encoders with and without the concatenated "gate" as input.

Baselines are: cosine similarity, linear map.

4 Experimental Settings

We use pretrained 300-dimensional w2v SGNS.

Antonym training data was obtained from WordNet. We started with all adjectives having antonyms in WN. We expanded the list by taking WN synonyms of the antonyms in these pairs. This resulted in an order of magnitude more training data, approximately 20K pairs (depending on setting), without introducing too much noise into the training data. Any input word in the test set (see below) was excluded from the INPUT side of the training data. We did not exclude it as a target in the training data. We also did not exclude cohyponyms of the test input words, since the theory is that the network needs these examples to learn the gates / contexts. That is, if 'cold' is in the test set, the model learns how to negate it by learning how to negate other temperature words.

Neutral context "gates" were obtained under three conditions. First, we took all WordNet cohyponyms of the adjective[1] sense of input word w, where cohyponyms include: other lemmas in the synset, children of attribute, synonyms, antonyms, synonyms of the antonyms, similar-tos. If there were fewer than 10 cohyponyms in WordNet, we increased the number to 10 with non-overlapping nearest neighbours from the original vector space. The "gate" is the centroid of these vectors.

The second condition is **unsupervised** gates. Here we do not use WN to find the semantic neighbourhood. We use the ten nearest neighbours from the original vector space. However, the target antonym is still supervised.

In the third condition, which we call **restricted**, we remove all cohyponys of test input words from the training data. This is to test how important it is to have training examples with similar gates. For example, if *cold* is a test word, we remove *hot, cold, tepid, cool* etc. from the training data.

We used MSE loss. Hyperparameters were

[1] includes adjective satellite

tuned on the GRE development set. The feed forward and CCCURE networks have hidden layers of 600 units, while the DAE has a hidden layer of 150, and 300 for DAE-Concat. Minibatch size was 48 for bilinear networks and 16 for the others. Number of epochs 400 for feed-forward, 200 for bilinear and DAE, 100 for DAE-Concat. Optimization adadelta with $\rho = 0.95$.

5 Evaluation

We evaluted our models with two experiments. Experiment 1 uses the GRE questions of Mohammad. The task is, given an input word, to pick the best antonym out of five options. An example is [TODO: GRE EXAMPLE]. We restrict the questions to those in which both input and target words are adjectives.

Experiment 2 evaluates the precision of the models. Remember that the task is to negate an adjective given its word vector. The most natural criterion for success is whether the model returns a good antonym at rank one, or a number of good antonyms at lower ranks. Following (Gorman and Curran, 2005), we expand our gold standard for evaluation beyond WN, because we found that WN did not have enough coverage. We obtained additional gold standard antonyms from the online version of Roget's 21st Century Thesaurus, Third Edition.[2] We report precision at ranks 1, 5, and 10. For this evaluation we use two test datasets: the input words from the GRE test set, and the test set of Santus [TODO: cite].

6 Results and Discussion

Table 6 gives the results of Experiment 1. The cosine baseline is already fairly strong at 0.50, suggesting that in general about two out of the five choices are closely related to the input word.

The neutral context "gate" clearly provides useful information for this task, because FF-Concat beats FF, and DAE-Concat beats DAE. Under all conditions, the CCCURE achieves the highest accuracy, reaching an impressive 0.69 in the supervised condition, which suggests that the bilinear connections are useful for antonym detection.

The DAE does fairly poorly, suggesting that DAEs are not the right architecture for this task. Putting the input word through a bottleneck may be enough to extract crucial semantic features for reconstructing the original word, but not enough to negate it, where you need to detect where the input word is more extreme than its neutral context.

[2] http://thesaurus.com

Method	Accuracy
Random	0.20
Cosine	0.50
Linear	
FF	0.58
FF-Concat	0.65
DAE	0.52
DAE-Concat	0.59
CCCURE	**0.69**
FF Unsup	0.54
FF-Concat Unsup	0.56
DAE Unsup	
DAE-Concat Unsup	
CCCURE Unsup	**0.60**
FF Restricted	0.51
FF-Concat Restricted	0.63
DAE Restricted	
DAE-Concat Restricted	
CCCURE Restricted	**0.65**

Table 1: Accuracy on the 367 multiple-choice adjective questions in the GRE test set.

In the unsupervised setting, in which the neutral context "gates" are the average of ten nearest neighbour vectors rather than including supervised cohyponyms from WordNet, there is a notable loss of accuracy. This indicates that the makeup of the gates matters and the neighbours are insufficient. For one thing, the centroid of the neighbours may be too close to the vector itself. Even in this setting, however, CCCURE achieves a respectable 0.60 accuracy.

In the restricted setting, FF performs at baseline, suggesting that it is overfitting when it can use the cohyponyms of the test words to learn antonyms. Surprisingly, the CCCURE still performs quite well.

Overall, this task does not mirror what the CCCURE was designed to do. Inspection of the output shows a number of cases where it fails to predict the specific antonym chosen in the GRE, but the one-best antonym is still very good. [TODO: example]. This is addressed in Experiment 2.

[TODO: table and discussion of Expt 2]

[TODO: we expect the network will have the hardest time negating adjs that don't have a good gate, i.e. a well-defined semantic neighbourhood. Look for some examples.]

7 Related Work

[TODO: Widdows. O Seaghdha. Pham. Nguyen. 3x Santus. Grefenstette and Hermann. Baroni. Mohammad. Maybe de Marneffe on adjective scales.]

[TODO: Vision citations: Memisevic et al.]

8 Conclusion

We have shown that a bilinear model is effective at learning to negate adjectives, in the form of predicting their antonyms, which provides a good first approximation of negation. This improves over a linear model, a feed-forward network that concatenates a word vector and its context gate, and a standard denoising autoencoder. The results suggest that knowledge of the alternative set is important for negation, and gives us a good idea about what kinds of features may be important for learning a negation mapping. In the broader context of function words, it is perhaps not too farfetched to imagine that a specialised network with multiplicative connections might be necessary to model these kinds of words.

[TODO Future work: more efficient relational encoders that need fewer parameters. Disambiguation (some errors due to polysemy). Less supervision, larger/noiser training sets. Nouns and verbs, which involves alternatives rather than antonyms, with a probability distribution over alternatives.]

References

James Gorman and James R. Curran. 2005. Approximate searching for distributional similarity. In *Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition*, pages 97–104, Ann Arbor.

N. Pham, A. Lazaridou, and M. Baroni. 2015. A multitask objective to inject lexical contrast into distributional semantics. In *Proceedings of ACL*, pages 21–26.