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

Identifying antonyms in a vector space model is a notoriously challenging task, because the vectors for related words of opposite polarity, such as *hot* and *cold*, are close to each other in most commonly-used word embedding models. Retrofitting embeddings to enhance lexical contrast, or learning embeddings with lexical contrast as a secondary objective (Pham et al., 2015; Nguyen et al., 2016; Mrkšić et al., 2016), has been shown to improve the overall embedding quality as well as the ability to distinguish synonyms from antonyms. In this paper we adress a different task: given an arbitrary set of word embeddings, we wish to learn a negation function which returns an antonym for a given word. This task is relevant for composing larger units of meaning, where a phrase such as *not cold* might need to be assigned a vector space representation.

We focus on negating adjectives, and treat negation as prediction of a one-best antonym. For example, given the expression *not talkative* and the vector $\overrightarrow{talkative}$, the negation function should return a word from the set *quiet, taciturn, uncommunicative*, etc.

Semantically, antonyms share a domain—e.g. *temperature*, but differ in their value—e.g. *cold* (Turney, 2012; Hermann et al., 2013). The challenge for negation is to learn which features need to change, and in what direction, to change the value while retaining the semantic domain of the word. Santus et al. (2015) propose that antonyms differ along their less salient shared contexts, using an LMI-based measure for distinguishing antonymy from other binary relations; crucially, however, relation classification on word pairs does not predict an antonym given a word.

In this paper we exploit the semantic neighbourhood of each adjective as a stand-in for the domain. We hypothesize that the relevant features for negating, say, a temperature adjective, differ from those for an emotion adjective. Therefore, we learn a function that uses a vector representing the neutral semantic context of the input word. Moreover, a negation function must be (informally) involuntory, i.e. its own inverse, so it must detect the "direction" in which the input word is more extreme than the neutral vector. This approach is inspired by Kruszewski et al. (2016), who observe that nearest neighbours in a vector space are a good approximation for the alternatives that humans produce for negation.

By using a bilinear relational neural network architecture used to identify transformations in computer vision, we are able to learn a negation 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: Background

Relational autoencoders (RAE), also known as gated autoencoders (GAE), have been used in computer vision to learn representations of transformations between images, such as rotation or translation (Memisevic and Hinton, 2007; Memisevic, 2012, 2013). RAEs are a type of *gated network*, in which multiplicative connections between two related inputs are used for prediction or representation learning. The "gating" of one image vector by another allows the feature detectors

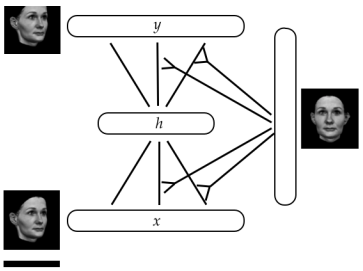

Figure 1: Neural network architectures for RAE, CCRAE, CCCRE.

to concentrate on the correspondences between the related images, rather than being distracted by the differences between untransformed images (Figure 2). The key is bilinear connections; there is a weight for every pair of units in the input vector and gate vector. For an overview of RAEs see Memisevic (2013); Sigaud et al. (2015).

We note that despite the terminology, the gates in an RAE perform a somewhat different function than in an LSTM (Hochreiter and Schmidhuber, 1997). Both network architectures use a nonlinearity to modulate the contents of a product; in an RAE this is an outer (bilinear) product while in an LSTM it is a Hadamard (element-wise) product. However, LSTM gates are memory gates which represent an internal hidden state of the network, while RAE gates are part of the input to the network.

An AE can be defined as follows (we omit bias terms for the sake of simplicity), where $\mathrm{W_e}$ are the encoder weights and $\mathrm{W_d}$ as the decoder weights. In autoencoders it is typical for them to be tied so that $\mathrm{W_d} = \mathrm{W_e}^T$.

$$\begin{aligned} h &= f(x) = \sigma(\mathrm{W_e}x) \\ y &= g(h) = \mathrm{W_d}h \end{aligned} \tag{1}$$

For an RAE, we have two inputs x and z. Instead of a weight matrix W we have a weight tensor $\overline{\mathrm{W}} \in R^{n_H \times n_X \times n_Z}$. The RAE is defined as follows.

$$\begin{aligned} h &= f(x,z) = \sigma((\overline{\mathrm{W_e}}z)x) \\ y &= g(h,z) = \sigma((\overline{\mathrm{W_d}}h)z) \end{aligned} \tag{2}$$

Rudy and Taylor (2015) introduce a class-conditional gated autoencoder in which the gate is a one-hot class label, rather than a transformed version of the input image. For example, in the MNIST task the label represents the digit. This is like training an autoencoder per class but with weight sharing across classes. See Figure 2(b).

2.2 Continuous Class-Conditional Relational Encoders

Our bilinear model is a continuous class-conditional relational encoder (CCCRE). The model architecture is the same as an RAE with untied encoder and decoder weights. However, the training signal differs from a classic RAE in two ways. First, it is not an autoencoder, but simply an encoder, because it is not trained to reproduce the input but rather to transform the input to its antonym. Second, the encoder is class-conditional in the sense of Rudy and Taylor (2015). In a classic RAE the input and gate are related according to a transformation, but in a class-conditional AE the gate represents the class. Unlike the one-hot gates of Rudy and Taylor (2015), our gates, representing the semantic domain of the input vector, are real-valued. See Figure 2(c).

3 Experiments

3.1 Models

We compare the CCCRE with (1) an Untied Encoder with a bottleneck. (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. We did not find denoising to help in any of our models and do not use it.

Baselines are: cosine similarity, linear map.

4 Experimental Settings

We use the publicly-available[1] embeddings obtained using the skip-gram with negative sampling model of Mikolov et al. (2013). These are 300-dimensional and were trained on part of the Google News dataset.

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

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

[2] includes adjective satellite

	Training Condition		
Method	**Stand.**	**Unsup.**	**Restr.**
Random	0.20	—	—
Cosine	0.50	—	—
Linear	0.56		
Linear-Concat	0.66		
UE	0.57		
UE-Concat	0.63		
FF	0.58	0.54	0.51
FF-Concat	0.65	0.56	0.63
CCCRE	**0.69**	**0.60**	**0.65**

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

5 Evaluation

We evaluted our models with two experiments. Experiment 1 uses the Graduate Record Examination (GRE) questions of Mohammad et al. (2013). The task is, given an input word, to pick the best antonym out of five options. An example from the development set is shown in (3), where the input word is *piquant* and the correct answer is *bland*. We restrict the questions to those in which both input and target words are adjectives.

piquant: (a) shocking (b) jovial (c) rigorous (d) merry (e) **bland** (3)

We evaluate by predicting the antonym for the input word. We measure cosine distance from the predicted vector to each of the five terms in the multiple choice question, and predict the closest term. The results are shown as accuracy, i.e. percentage of questions answered correctly.

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,

	GRE						Lenci					
	Stand.		Unsup.		Restr.		Stand.		Unsup.		Restr.	
Method	**P@1**	**P@5**	**P@1**	**P@5**	**P@1**	**P@5**	**P@1**	**P@5**	**P@1**	**P@5**	**P@1**	**P@5**
Cosine	0.05	0.07	—	—	—	—	0.13	0.10	—	—	—	—
Linear	0.39	0.33										
FF	0.37	0.32	0.34	0.30	0.08	0.15	0.30	0.24				
FF-Concat	0.36	0.30	0.46	0.40	0.37	0.34	0.34	0.26				
UE	0.38	0.33					0.28	0.22				
UE-Concat	0.38	0.33					0.33	0.28				
CCCRE	**0.66**	**0.49**	**0.52**	**0.42**	**0.52**	**0.38**	**0.39**	**0.32**				

Table 2: Precision at ranks 1 and 5 on the GRE and Lenci datasets.

Third Edition.[3] We report precision at ranks 1 and 5. For this evaluation we use two test datasets: the input words from the GRE test set, and a set of 99 adjectives and their antonyms which is part of a dataset collected by Lenci and Benotto acccording to the guidelines of Schulte im Walde and Köper (2013).

6 Results and Discussion

Table 4 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 every linear and non-linear model is helped by concatenation of the gate vector to the input. None of the models without the gate score much higher than the cosine similarity baseline.

Under all conditions, the CCCRE achieves the highest accuracy. In fact it is the only model that beats a linear baseline, which suggests that the bilinear connections are useful for antonym detection.

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, CCCRE 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. CCCRE still performs quite well, suggesting that it is able to share weights across the different semantic classes.

Overall, this task does not mirror what the CCCRE 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.

Table 5 shows precision at ranks 1 and 5 on the GRE and Lenci datasets. In the standard training condition with supervised gates, CCCRE shows an impressive P@1 of 0.66, i.e. two thirds of the time it is able to produce an antonym of the input word as the nearest neighbour of the prediction. All of the other models score less than 0.4. In the other trainin conditions CCCRE retains the ability to give a one-best antonym about half the time.

The Lenci dataset is more challenging [TODO: say why]. However, CCCRE still achieves the highest precision.

7 Related Work

Widdows (2003) introduces a binary negation function for vector spaces, e.g. *suit NOT lawsuit*, which is used for word sense disambiguation in Information Retrieval. Turney (2012); Hermann et al. (2013) propose a multi-part vector representation that separates domain and value, but this does not allow negation of a word in an arbitrary vector space. Pham et al. (2015); Nguyen et al. (2016); Mrkšić et al. (2016) use ontologies such as WN or domain-specific ontologies to retrofit embeddings, or learn embeddings with a secondary objective, pushing synonyms closer together and antonyms further apart. This improves the overall vector space and makes it more likely that close words will be synonyms, but does not provide a

[3] http://thesaurus.com

way to find antonyms in the space. Santus et al. (2014a,b, 2015) perform synonym-antonym discrimination using an unsupervised measure based on the relative salience of shared contexts. This type of relation classification does not make it possible to negate an arbitrary word. Mohammad et al. (2013) use a supervised thesaurus-based method augmented with co-occurrence statistics on the GRE task (including nouns and verbs).

[TODO: vision citations for a couple of applications of RAEs]

8 Conclusion

We have shown that a neutral context vector, representing the set of alternatives, improves antonym prediction in linear and non-linear models, and that the multiplicative connections in a bilinear model are effective at learning to negate adjectives. In the broader context of modelling closed-class function words, it is perhaps not too far-fetched to imagine that a specialised network with multiplicative connections might be necessary to model these kinds of words.

Our implementation is a simple one, using a full tensor for the bilinear layer, and takes several hours to train on a GPU. Future work will exploit the many options for reducing the number of parameters to be learned (Alain and Olivier, 2013) and address negation of nouns and verbs, where negation is more likely to involve a probability distribution over alternatives—e.g. *didn't run* might mean *walk* or *amble*—rather than predicting a single best antonym.

References

Droniou Alain and Sigaud Olivier. 2013. Gated autoencoders with tied input weights. In *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, GA.

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.

Karl Moritz Hermann, Edward Grefenstette, and Phil Blunsom. 2013. "not not bad" not "bad": A distributional account of negation. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 74–82, Sofia, Bulgaria.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Germán Kruszewski, Denis Paperno, Raffaella Bernardi, and Marco Baroni. 2016. There is no logical negation here, but there are alternatives: Modeling conversational negation with distributional semantics. *Computational Linguistics*, 42:xxx–xxx.

Roland Memisevic. 2012. On multi-view feature learning. In *Proceedings of ICML*.

Roland Memisevic. 2013. Learning to relate images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1829–1846.

Roland Memisevic and Geoffrey Hinton. 2007. Unsupervised learning of image transformations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Saif M. Mohammad, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2013. Computing lexical contrast. *Computational Linguistics*, 39 (3):555–590.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of NAACL*, pages 142–148, San Diego.

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonymsynonym distinction. In *Proceedings of ACL*, pages 454–459, Berlin.

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.

Jan Rudy and Graham Taylor. 2015. Generative class-conditional denoising autoencoders. In *Proceedings of ICLR Workshop*.

Enrico Santus, Qin Lu, Alessandro Lenci, and Chu-Ren Huang. 2014a. Unsupervised antonym-synonym discrimination in vector space. In *Atti della Conferenza di Linguistica Computazionale Italiana (CLIC-IT 2014)*, Pisa, Italy.

Enrico Santus, Qin Lu, Alessandro Lenci, and Chu-Ren Huang. 2014b. Taking antonymy

mask off in vector space. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation (PACLIC 2014)*, Phuket, Thailand.

Enrico Santus, Alessandro Lenci, Qin Lu, and Chu-Ren Huang. 2015. When similarity becomes opposition: Synonyms and antonyms discrimination in DSMs. *Italian Journal of Computational Linguistics*, 1(1):41–54.

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

Olivier Sigaud, Clément Masson, David Filliat, and Freek Stulp. 2015. Gated networks: an inventory. arXiv:1512.03201 [cs.LG].

Peter D. Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585.

Dominic Widdows. 2003. Orthogonal negation in vector spaces for modelling word-meanings and document retrieval. In *Proceedings of ACL*, pages 136–143, Sapporo, Japan.