Counter-fitting Word Vectors to Linguistic Constraints

Nikola Mrkšić[1], Diarmuid Ó Séaghdha[2], Blaise Thomson[2], Milica Gašić[1]
Lina Rojas-Barahona[1], Pei-Hao Su[1], David Vandyke[1], Tsung-Hsien Wen[1], Steve Young[1]
[1] Department of Engineering, University of Cambridge, UK
[2] Apple Inc.
{nm480,mg436,phs26,djv27,thw28,sjy}@cam.ac.uk
{doseaghdha, blaisethom}@apple.com

Abstract

In this work, we present a novel *counter-fitting* method which injects antonymy and synonymy constraints into vector space representations in order to improve the vectors' capability for judging semantic similarity. Applying this method to publicly available pre-trained word vectors leads to a new state of the art performance on the SimLex-999 dataset. We also show how the method can be used to tailor the word vector space for the downstream task of dialogue state tracking, resulting in robust improvements across different dialogue domains.

	east	**expensive**	**British**
Before	west	pricey	American
	north	cheaper	Australian
	south	costly	Britain
	southeast	overpriced	European
	northeast	inexpensive	England
After	eastward	costly	Brits
	eastern	pricy	London
	easterly	overpriced	BBC
	-	pricey	UK
	-	afford	Britain

Table 1: Nearest neighbours for target words using GloVe vectors before and after counter-fitting

1 Introduction

Many popular methods that induce representations for words rely on the *distributional hypothesis* – the assumption that semantically similar or related words appear in similar contexts. This hypothesis supports unsupervised learning of meaningful word representations from large corpora (Curran, 2003; Ó Séaghdha and Korhonen, 2014; Mikolov et al., 2013; Pennington et al., 2014). Word vectors trained using these methods have proven useful for many downstream tasks including machine translation (Zou et al., 2013) and dependency parsing (Bansal et al., 2014).

One drawback of learning word embeddings from co-occurrence information in corpora is that it tends to coalesce the notions of *semantic similarity* and *conceptual association* (Hill et al., 2014b). Furthermore, even methods that can distinguish similarity from association (e.g., based on syntactic co-occurrences) will generally fail to tell synonyms from antonyms (Mohammad et al., 2008). For example, words such as *east* and *west* or *expensive* and *inexpensive* appear in near-identical contexts, which means that distributional models produce very similar word vectors for such words. Examples of such anomalies in GloVe vectors can be seen in Table 1, where words such as *cheaper* and *inexpensive* are deemed similar to (their antonym) *expensive*.

A second drawback is that similarity and antonymy can be application- or domain-specific. In our case, we are interested in exploiting distributional knowledge for the *dialogue state tracking* task (DST). The DST component of a dialogue system is responsible for interpreting users' utterances and updating the system's *belief state* – a probability distribution over all possible states of the dialogue. For example, a DST for the restaurant domain needs to detect whether the user wants a *cheap* or *expensive* restaurant. Being able to generalise using distributional information while still distinguishing between semantically different yet conceptually related words

(e.g. *cheaper* and *pricey*) is critical for the performance of dialogue systems. In particular, a dialogue system can be led seriously astray by false synonyms.

We propose a method that addresses these two drawbacks by using synonymy and antonymy relations drawn from either a general lexical resource or an application-specific ontology to fine-tune distributional word vectors. Our method, which we term *counter-fitting*, is a lightweight post-processing procedure in the spirit of *retrofitting* (Faruqui et al., 2015). The second row of Table 1 illustrates the results of counter-fitting: the nearest neighbours capture true similarity much more intuitively than the original GloVe vectors. The procedure improves word vector quality regardless of the initial word vectors provided as input.[1] By applying counter-fitting to the Paragram-SL999 word vectors provided by Wieting et al. (2015), we achieve new state-of-the-art performance on SimLex-999, a dataset designed to measure how well different models judge semantic similarity between words (Hill et al., 2014b). We also show that the counter-fitting method can inject knowledge of dialogue domain ontologies into word vector space representations to facilitate the construction of semantic dictionaries which improve DST performance across two different dialogue domains. Our tool and word vectors are available at `github.com/nmrksic/counter-fitting`.

2 Related Work

Most work on improving word vector representations using lexical resources has focused on bringing words which are known to be semantically related closer together in the vector space. Some methods modify the prior or the regularization of the original training procedure (Yu and Dredze, 2014; Bian et al., 2014; Kiela et al., 2015). Wieting et al. (2015) use the Paraphrase Database (Ganitkevitch et al., 2013) to train word vectors which emphasise word similarity over word relatedness. These word vectors achieve the current state-of-the-art performance on the SimLex-999 dataset and are used as input for counter-fitting in our experiments.

[1]When we write "improve", we refer to improving the vector space for a specific purpose. We do not expect that a vector space fine-tuned for semantic similarity will give better results on semantic relatedness. As Mohammad et al. (2008) observe, antonymous concepts are related but not similar.

Recently, there has been interest in lightweight post-processing procedures that use lexical knowledge to refine off-the-shelf word vectors without requiring large corpora for (re-)training as the aforementioned "heavyweight" procedures do. Faruqui et al.'s (2015) *retrofitting* approach uses similarity constraints from WordNet and other resources to pull similar words closer together.

The complications caused by antonymy for distributional methods are well-known in the semantics community. Most prior work focuses on extracting antonym pairs from text rather than exploiting them (Lin et al., 2003; Mohammad et al., 2008; Turney, 2008; Hashimoto et al., 2012; Mohammad et al., 2013). The most common use of antonymy information is to provide features for systems that detect contradictions or logical entailment (Marcu and Echihabi, 2002; de Marneffe et al., 2008; Zanzotto et al., 2009). As far as we are aware, there is no previous work on exploiting antonymy in dialogue systems. The modelling work closest to ours are Liu et al. (2015), who use antonymy and WordNet hierarchy information to modify the heavyweight Word2Vec training objective; Yih et al. (2012), who use a Siamese neural network to improve the quality of Latent Semantic Analysis vectors; Schwartz et al. (2015), who build a standard distributional model from co-occurrences based on *symmetric patterns*, with specified antonymy patterns counted as negative co-occurrences; and Ono et al. (2015), who use thesauri and distributional data to train word embeddings specialised for capturing antonymy.

3 Counter-fitting Word Vectors to Linguistic Constraints

Our starting point is an indexed set of word vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ with one vector for each word in the vocabulary. We will inject semantic relations into this vector space to produce new word vectors $V' = \{\mathbf{v'}_1, \mathbf{v'}_2, \ldots, \mathbf{v'}_N\}$. For antonymy and synonymy we have a set of constraints A and S, respectively. The elements of each set are pairs of word indices; for example, each pair (i, j) in S is such that the i-th and j-th words in the vocabulary are synonyms. The objective function used to counter-fit the pre-trained word vectors V to the sets of linguistic constraints A and S contains three different terms:

1. Antonym Repel (AR): This term serves to *push* antonymous words' vectors away from each other in the transformed vector space V':

$$\text{AR}(V') = \sum_{(u,w)\in A} \tau\left(\delta - d(\mathbf{v}'_u, \mathbf{v}'_w)\right)$$

where $d(v_i, v_j) = 1 - \cos(v_i, v_j)$ is a distance derived from cosine similarity and $\tau(x) \triangleq max(0, x)$ imposes a margin on the cost. Intuitively, δ is the "ideal" minimum distance between antonymous words; in our experiments we set $\delta = 1.0$ as it corresponds to vector orthogonality.

2. Synonym Attract (SA): The counter-fitting procedure should seek to bring the word vectors of known synonymous word pairs closer together:

$$\text{SA}(V') = \sum_{(u,w)\in S} \tau\left(d(\mathbf{v}'_u, \mathbf{v}'_w) - \gamma\right)$$

where γ is the "ideal" maximum distance between synonymous words; we use $\gamma = 0$.

3. Vector Space Preservation (VSP): the topology of the original vector space describes relationships between words in the vocabulary captured using distributional information from very large textual corpora. The VSP term bends the transformed vector space towards the original one as much as possible in order to preserve the semantic information contained in the original vectors:

$$\text{VSP}(V, V') = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau\left(d(\mathbf{v}'_i, \mathbf{v}'_j) - d(\mathbf{v}_i, \mathbf{v}_j)\right)$$

For computational efficiency, we do not calculate distances for every pair of words in the vocabulary. Instead, we focus on the (pre-computed) neighbourhood $N(i)$, which denotes the set of words within a certain radius ρ around the i-th word's vector in the original vector space V. Our experiments indicate that counter-fitting is relatively insensitive to the choice of ρ, with values between 0.2 and 0.4 showing little difference in quality; here we use $\rho = 0.2$.

The objective function for the training procedure is given by a weighted sum of the three terms:

$$C(V, V') = k_1\text{AR}(V') + k_2\text{SA}(V') + k_3\text{VSP}(V, V')$$

where $k_1, k_2, k_3 \geq 0$ are hyperparameters that control the relative importance of each term. In our experiments we set them to be equal: $k_1 = k_2 = k_3$. To minimise the cost function for a set of starting vectors V and produce counter-fitted vectors V', we run stochastic gradient descent (SGD) for 20 epochs. An end-to-end run of counter-fitting takes less than two minutes on a laptop with four CPUs.

3.1 Injecting Dialogue Domain Ontologies into Vector Space Representations

Dialogue state tracking (DST) models capture users' goals given their utterances. Goals are represented as sets of constraints expressed by *slot-value* pairs such as [food: *Indian*] or [parking: *allowed*]. The set of slots S and the set of values V_s for each slot make up the *ontology* of a dialogue domain.

In this paper we adopt the recurrent neural network (RNN) framework for tracking suggested in (Henderson et al., 2014d; Henderson et al., 2014c; Mrkšić et al., 2015). Rather than using a spoken language understanding (SLU) decoder to convert user utterances into meaning representations, this model operates directly on the n-gram features extracted from the automated speech recognition (ASR) hypotheses. A drawback of this approach is that the RNN model can only perform exact string matching to detect the slot names and values mentioned by the user. It cannot recognise synonymous words such as *pricey* and *expensive*, or even subtle morphological variations such as *moderate* and *moderately*. A simple way to mitigate this problem is to use *semantic dictionaries*: lists of rephrasings for the values in the ontology. Manual construction of dictionaries is highly labour-intensive; however, if one could automatically detect high-quality rephrasings, then this capability would come at no extra cost to the system designer.

To obtain a set of word vectors which can be used for creating a semantic dictionary, we need to *inject* the domain ontology into the vector space. This can be achieved by introducing antonymy constraints between all the possible values of each slot (i.e. *Chinese* and *Indian*, *expensive* and *cheap*, etc.). The remaining linguistic constraints can come from semantic lexicons: the richer the sets of injected synonyms and antonyms are, the better the resulting word representations will become.

Model / Word Vectors	ρ
Neural MT Model (Hill et al., 2014a)	0.52
Symmetric Patterns (Schwartz et al., 2015)	0.56
Non-distributional Vectors (Faruqui and Dyer, 2015)	0.58
GloVe vectors (Pennington et al., 2014)	0.41
GloVe vectors + Retrofitting	0.53
GloVe + Counter-fitting	0.58
Paragram-SL999 (Wieting et al., 2015)	0.69
Paragram-SL999 + Retrofitting	0.68
Paragram-SL999 + Counter-fitting	**0.74**
Inter-annotator agreement	0.67
Annotator/gold standard agreement	0.78

Table 2: Performance on SimLex-999. Retrofitting uses the code and (PPDB) data provided by the authors

4 Experiments

4.1 Word Vectors and Semantic Lexicons

Two different collections of pre-trained word vectors were used as input to the counter-fitting procedure:

1. **Glove Common Crawl** 300-dimensional vectors made available by Pennington et al. (2014).

2. **Paragram-SL999** 300-dimensional vectors made available by Wieting et al. (2015).

The synonymy and antonymy constraints were obtained from two semantic lexicons:

1. **PPDB 2.0 (Pavlick et al., 2015):** the latest release of the Paraphrase Database. A new feature of this version is that it assigns relation types to its word pairs. We identify the *Equivalence* relation with synonymy and *Exclusion* with antonymy. We used the largest available (XXXL) version of the database and only considered single-token terms.

2. **WordNet (Miller, 1995)**: a well known semantic lexicon which contains vast amounts of high quality human-annotated synonym and antonym pairs. Any two words in our vocabulary which had antonymous word senses were considered antonyms; WordNet synonyms were not used.

In total, the lexicons yielded 12,802 antonymy and 31,828 synonymy pairs for our vocabulary, which consisted of 76,427 most frequent words in OpenSubtitles, obtained from `invokeit.wordpress.com/frequency-word-lists/`.

Semantic Resource	Glove	Paragram
Baseline (no linguistic constraints)	0.41	0.69
PPDB− (PPDB antonyms)	0.43	0.69
PPDB+ (PPDB synonyms)	0.46	0.68
WordNet− (WordNet antonyms)	0.52	0.74
PPDB− and PPDB+	0.50	0.69
WordNet− and PPDB−	0.53	0.74
WordNet− and PPDB+	0.58	0.74
WordNet− and PPDB− and PPDB+	0.58	0.74

Table 3: SimLex-999 performance when different sets of linguistic constraints are used for counter-fitting

4.2 Improving Lexical Similarity Predictions

In this section, we show that counter-fitting pre-trained word vectors with linguistic constraints improves their usefulness for judging semantic similarity. We use Spearman's rank correlation coefficient with the SimLex-999 dataset, which contains word pairs ranked by a large number of annotators instructed to consider only semantic similarity.

Table 2 contains a summary of recently reported competitive scores for SimLex-999, as well as the performance of the unaltered, retrofitted and counter-fitted GloVe and Paragram-SL999 word vectors. To the best of our knowledge, the 0.685 figure reported for the latter represents the current high score. This figure is above the average inter-annotator agreement of 0.67, which has been referred to as the ceiling performance in most work up to now.

In our opinion, the average inter-annotator agreement is not the only meaningful measure of ceiling performance. We believe it also makes sense to compare: **a)** the model ranking's correlation with the gold standard ranking to: **b)** the average rank correlation that individual human annotators' rankings achieved with the gold standard ranking. The SimLex-999 authors have informed us that the average annotator agreement with the gold standard is 0.78.[2] As shown in Table 2, the reported performance of all the models and word vectors falls well below this figure.

Retrofitting pre-trained word vectors improves GloVe vectors, but not the already semantically specialised Paragram-SL999 vectors. Counter-fitting substantially improves both sets of vectors, showing that injecting antonymy relations goes a long way

[2]This figure is now reported as a potentially fairer ceiling performance on the SimLex-999 website: `http://www.cl.cam.ac.uk/~fh295/simlex.html`.

False Synonyms	Fixed	False Antonyms	Fixed
sunset, sunrise	✓	dumb, dense	
forget, ignore		adult, guardian	
girl, maid		polite, proper	✓✓
happiness, luck	✓✓	strength, might	
south, north	✓	water, ice	
go, come	✓	violent, angry	✓✓
groom, bride		cat, lion	✓✓
dinner, breakfast		laden, heavy	✓✓
-	-	engage, marry	

Table 4: Highest-error SimLex-999 word pairs using Paragram vectors (before counter-fitting)

towards improving word vectors for the purpose of making semantic similarity judgements.

Table 3 shows the effect of injecting different categories of linguistic constraints. GloVe vectors benefit from all three sets of constraints, whereas the quality of Paragram vectors, already exposed to PPDB, only improves with the injection of WordNet antonyms. Table 4 illustrates how incorrect similarity predictions based on the original (Paragram) vectors can be fixed through counter-fitting. The table presents eight false synonyms and nine false antonyms: word pairs with predicted rank in the top (bottom) 200 word pairs and gold standard rank 500 or more positions lower (higher). Eight of these errors are fixed by counter-fitting: the difference between predicted and gold-standard ranks is now 100 or less. Interestingly, five of the eight corrected word pairs do not appear in the sets of linguistic constraints; these are indicated by double ticks in the table. This shows that secondary (i.e. *indirect*) interactions through the three terms of the cost function do contribute to the semantic content of the transformed vector space.

4.3 Improving Dialogue State Tracking

Table 5 shows the dialogue state tracking datasets used for evaluation. These datasets come from the Dialogue State Tracking Challenges 2 and 3 (Henderson et al., 2014a; Henderson et al., 2014b).

We used four different sets of word vectors to construct semantic dictionaries: the original GloVe and Paragram-SL999 vectors, as well as versions counter-fitted to each domain ontology. The constraints used for counter-fitting were all those from the previous section as well as antonymy constraints among the set of values for each slot. We treated all vocabulary words within some radius t of a slot value as

Dataset	Train	Dev	Test	#Slots
Restaurants	1612	506	1117	4
Tourist Information	1600	439	225	9

Table 5: Number of dialogues in the dataset splits used for the Dialogue State Tracking experiments

Word Vector Space	Restaurants	Tourist Info
Baseline (no dictionary)	68.6	60.5
GloVe	72.5	60.9
GloVe + Counter-fitting	73.4	**62.8**
Paragram-SL999	73.2	61.5
Paragram-SL999 + Counter-fitting	**73.5**	61.9

Table 6: Performance of RNN belief trackers (ensembles of four models) with different semantic dictionaries

rephrasings of that value. The optimal value of t was determined using a grid search: we generated a dictionary and trained a model for each potential t, then evaluated on the development set. Table 6 shows the performance of RNN models which used the constructed dictionaries. The dictionaries induced from the pre-trained vectors substantially improved tracking performance over the baselines (which used no semantic dictionaries). The dictionaries created using the counter-fitted vectors improved performance even further. Contrary to the SimLex-999 experiments, starting from the Paragram vectors did not lead to superior performance, which shows that injecting the application-specific ontology is at least as important as the quality of the initial word vectors.

5 Conclusion

We have presented a novel *counter-fitting* method for injecting linguistic constraints into word vector space representations. The method efficiently post-processes word vectors to improve their usefulness for tasks which involve making semantic similarity judgements. Its focus on separating vector representations of antonymous word pairs lead to substantial improvements on genuine similarity estimation tasks. We have also shown that counter-fitting can tailor word vectors for downstream tasks by using it to inject domain ontologies into word vectors used to construct semantic dictionaries for dialogue systems.

Acknowledgements

We would like to thank Felix Hill for help with the SimLex-999 evaluation. We also thank the anonymous reviewers for their helpful suggestions.

References

[Bansal et al.2014] Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of ACL*.

[Bian et al.2014] Jiang Bian, Bin Gao, and Tie-Yan Liu. 2014. Knowledge-powered deep learning for word embedding. In *Machine Learning and Knowledge Discovery in Databases*.

[Curran2003] James Curran. 2003. *From Distributional to Semantic Similarity*. Ph.D. thesis, School of Informatics, University of Edinburgh.

[de Marneffe et al.2008] Marie-Catherine de Marneffe, Anna N. Rafferty, and Christopher D. Manning. 2008. Finding contradictions in text. In *Proceedings of ACL*.

[Faruqui and Dyer2015] Manaal Faruqui and Chris Dyer. 2015. Non-distributional word vector representations. In *Proceedings of ACL*.

[Faruqui et al.2015] Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of NAACL HLT*.

[Ganitkevitch et al.2013] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of NAACL HLT*.

[Hashimoto et al.2012] Chikara Hashimoto, Kentaro Torisawa, Stijn De Saeger, Jong-Hoon Oh, and Junichi Kazama. 2012. Excitatory or inhibitory: A new semantic orientation extracts contradiction and causality from the Web. In *Proceedings of EMNLP-CoNLL*.

[Henderson et al.2014a] Matthew Henderson, Blaise Thomson, and Jason D. Wiliams. 2014a. The Second Dialog State Tracking Challenge. In *Proceedings of SIGDIAL*.

[Henderson et al.2014b] Matthew Henderson, Blaise Thomson, and Jason D. Wiliams. 2014b. The Third Dialog State Tracking Challenge. In *Proceedings of IEEE SLT*.

[Henderson et al.2014c] Matthew Henderson, Blaise Thomson, and Steve Young. 2014c. Robust Dialog State Tracking using Delexicalised Recurrent Neural Networks and Unsupervised Adaptation. In *Proceedings of IEEE SLT*.

[Henderson et al.2014d] Matthew Henderson, Blaise Thomson, and Steve Young. 2014d. Word-Based Dialog State Tracking with Recurrent Neural Networks. In *Proceedings of SIGDIAL*.

[Hill et al.2014a] Felix Hill, Kyunghyun Cho, Sbastien Jean, Coline Devin, and Yoshua Bengio. 2014a. Embedding word similarity with neural machine translation. *Computing Research Repository*.

[Hill et al.2014b] Felix Hill, Roi Reichart, and Anna Korhonen. 2014b. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *Computing Research Repository*.

[Kiela et al.2015] Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Proceedings of EMNLP*.

[Lin et al.2003] Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of IJCAI*.

[Liu et al.2015] Quan Liu, Hui Jiang, Si Wei, Zhen-Hua Ling, and Yu Hu. 2015. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of ACL*.

[Marcu and Echihabi2002] Daniel Marcu and Abdsemmad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of ACL*.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of NIPS*.

[Miller1995] George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*.

[Mohammad et al.2008] Saif Mohammad, Bonnie Dorr, and Graeme Hirst. 2008. Computing word-pair antonymy. In *Proceedings of EMNLP*.

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

[Mrkšić et al.2015] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Multi-domain Dialog State Tracking using Recurrent Neural Networks. In *Proceedings of ACL*.

[Ono et al.2015] Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word Embedding-based Antonym Detection using Thesauri and Distributional Information. In *Proceedings of NAACL HLT*.

[Ó Séaghdha and Korhonen2014] Diarmuid Ó Séaghdha and Anna Korhonen. 2014. Probabilistic distributional semantics. *Computational Linguistics*, 40(3):587–631.

[Pavlick et al.2015] Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevich, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of ACL*.

[Pennington et al.2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of EMNLP*.

[Schwartz et al.2015] Roy Schwartz, Roi Reichart, and Ari Rappoport. 2015. Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of CoNLL*.

[Turney2008] Peter D. Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *Proceedings of COLING*.

[Wieting et al.2015] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*.

[Yih et al.2012] Wen-Tau Yih, Geoffrey Zweig, and John C. Platt. 2012. Polarity inducing Latent Semantic Analysis. In *Proceedings of ACL*.

[Yu and Dredze2014] Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of ACL*.

[Zanzotto et al.2009] Fabio Massimo Zanzotto, Marco Pennachiotti, and Alessandro Moschitti. 2009. A machine learning approach to textual entailment recognition. *Journal of Natural Language Engineering*, 15(4):551–582.

[Zou et al.2013] Will Y. Zou, Richard Socher, Daniel M. Cer, and Christopher D. Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In *Proceedings of EMNLP*.