Retrofitting Word Vectors to Semantic Lexicons

Manaal Faruqui **Jesse Dodge** **Sujay K. Jauhar**
Chris Dyer **Eduard Hovy** **Noah A. Smith**
Language Technologies Institute
Carnegie Mellon University
Pittspurgh, PA, 15213, USA
{mfaruqui,jessed,sjauhar,cdyer,ehovy,nasmith}@cs.cmu.edu

Abstract

Vector space word representations are typically learned using only co-occurrence statistics from text corpora. Although such statistics are informative, they disregard easily accessible (and often carefully curated) information archived in semantic lexicons such as WordNet, FrameNet, and the Paraphrase Database. This paper proposes a technique to leverage both distributional and lexicon-derived evidence to obtain better representations. We run belief propagation on a word type graph constructed from word similarity information from lexicons to encourage connected (related) words to have similar representations, and also to be close to the unsupervised vectors. Evaluated on a battery of standard lexical semantic evaluation tasks in several languages, using several different underlying word vector models, we obtain substantially improved vectors and consistently outperform existing approaches of incorporating semantic knowledge in word vectors.

1 Introduction

Data-driven learning of word vectors that capture lexico-semantic properties is a technique of central importance in natural language processing. These word vectors can in turn be used for identifying semantically related word pairs [1, 2] or as features in downstream text processing applications [3]. A variety of approaches for constructing vector space embeddings of vocabularies are in use, notably including taking low rank approximations of cooccurrence statistics [4] and using internal representations from neural network models of word sequences [5].

Because of their value as lexical semantic representations, there has been much research on improving the quality of vectors. For example, cooccurrence statistics have been expanded to incorporate multilingual context [6, 7, 8] or define context in terms of dependency links [9]. However, a notable absence in vector construction techniques is the use of information in *semantic lexicons*, which provide type-level information about the semantics of words, typically by identifying *synonymy*, *hypernymy*, *hyponymy*, and *paraphrase* relations between word types (or, in some cases, between sense-distinguished word types). Examples of such resources include WordNet [10], FrameNet [11] and the Paraphrase database [12].

The contribution of this paper is a pair of graph-based learning techniques for leveraging such resources to obtain higher quality semantic vectors. The first, which we call "retrofitting", is applied as a post-processing step and uses belief propagation on a graph constructed from lexicon-derived relational information to update word vectors (§2). Intuitively, this method encourages the new vectors to be (i) similar to the vectors of related word types and (ii) similar to their starting representations. Retrofitting is extremely fast, takes about 5 seconds for a graph of 102,000 words and its run-time is independent of the original word vector model. The second approach formulates the relationship between words found in the semantic lexicon as a structured regulariser on the original training objective (§3).

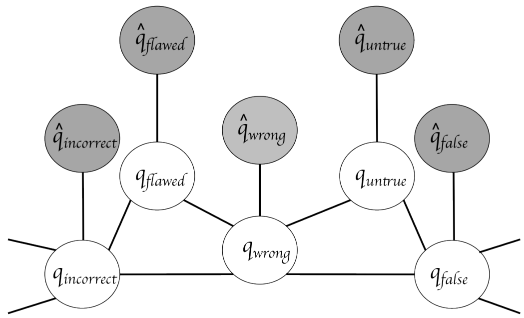

Figure 1: Word graph with edges between related words showing the observed and the inferred word vector representations.

Experimentally, we show that our methods work well with different state-of-the-art word vector models (§4) while using different kinds of semantic lexicons (§5) and gives substantial improvements on a variety of vector evaluations tasks (§6) in multiple languages (§7.3). Furthermore, retrofitting—which is applied as a post-processing step to vectors of any kind—performs at least as well or better than using lexical knowledge during training.

2 Retrofitting with Semantic Lexicons

Let $V = \{w_1, \ldots, w_n\}$ be a **vocabulary**, i.e. the set of word types, and Ω be an **ontology** that encodes semantic relations between words in V. We represent Ω as an undirected graph (V, E) with one vertex for each word type and edges $(w_i, w_j) \in E \subseteq V \times V$ indicating a semantic relationship of interest. These relations differ for different semantic lexicons and are described later (§5).

The matrix $\hat{Q}$ will be the collection of vector representations $\hat{q}_i \in \mathbb{R}^n$, for each $w_i \in V$, learned using a standard data-driven technique. Our objective is to learn the matrix $Q = (q_1, \ldots, q_n)$ such that the columns are both close (under a distance metric) to their counterparts in $\hat{Q}$ *and* to adjacent vertices in Ω. Figure 1 shows a small word graph with such edge connections; white nodes are labeled with the Q vectors to be retrofitted (and correspond to V_Ω); filled nodes are labeled with the corresponding vectors in $\hat{Q}$, which are observed. The graph can be interpreted as a Markov random field [13].

The distances between a pair of vectors is defined to be the Euclidean distance. Since we want the inferred word vector to be close to the observed value $\hat{q}_i$ and close to its neighbors $q_j, \forall j$ such that $(i, j) \in E$, the objective to be minimized becomes:

$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

where α and β values control the relative strengths of associations (§7.1).

In this case, we first train the word vectors independent of the information in the semantic lexicons and then retrofit them. Ψ is convex in Q and its solution can be found by solving a system of linear equations. To do so, we use an efficient iterative updating method [14, 15, 16, 17]. We take the first derivative of Ψ with respect to one q_i vector, and by equating it to zero arrive at the following online update:

$$q_i = \frac{\sum_{j:(i,j)\in E} \beta_{ij} q_j + \alpha \hat{q}_i}{\sum_{j:(i,j)\in E} \beta_{ij} + \alpha}. \quad (1)$$

In practice running this procedure for 10 iterations leads to convergence i.e, euclidean distance between $\forall i, q_i$ of consecutive iterations $< 10^{-3}$. Any kind of word vector representation can be retrofitted with this approach. We initialize the vectors in Q to be equal to the vectors in $\hat{Q}$.

3 Semantic Lexicons during Learning

Many word vector learning methods are cast as maximum likelihood estimation problems that rely on gradient-based algorithms to update word representations to iteratively improve the log-likelihood [18, 19, 20, *inter alia*]. In these cases, instead of "retrofitting" vectors, we can alter the learning objective with a prior that encourages semantically related vectors (in Ω) to be close together. In this setting, semantic lexicons can play the role of a prior on Q which we define as follows:

$$p(Q) \propto \exp\left(-\gamma \sum_{i=1}^{n} \sum_{j:(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2\right), \tag{2}$$

where γ is a hyperparameter that controls the strength of the prior. This prior on the word vector parameters forces words connected in the lexicon to have close vector representations as did $\Psi(Q)$ (with the role of $\hat{Q}$ being played by cross entropy to the empirical distribution).

Incorporating this prior during estimation of Q equates to maximum *a posteriori* (MAP) estimation. We consider two techniques. In the first, we apply adaptive gradient descent (AdaGrad [21]), using the sum of gradients of the log-likelihood (given by the extant vector learning model) and the log-prior (from Eq. 2), with respect to Q. Since computing gradient of Eq. 2 is linear in the vocabulary size n, we use lazy updates [22] every k words during training.[1] We call this the **lazy** method of Bayesian retrofitting during training. The second technique applies stochastic gradient descent to the log-likelihood, and after every k words applies the update in Eq. 1. We call this the **periodic** method of retrofitting during training.

4 Word Vector Representations

We test our retrofitting model on several different models of word vector representations described below. In some cases we use pre-trained vectors; in others we train models on our own data.

Latent Semantic Analysis (LSA) We perform latent semantic analysis [4] on a word-word co-occurrence matrix. We construct a word co-occurrence frequency matrix for a given training corpus where each row corresponds to a word type in the corpus and each column corresponds to a context feature. In our case, every column/context is a word which might occur in a window of 5 positions around the target word; for scalability, we include words with frequency ≥ 10 and exclude the 100 most frequent words. We then replace every entry in the sparse frequency matrix by its pointwise mutual information (PMI) [23, 24] resulting in $\mathbf{X}$. We factorize the matrix $\mathbf{X} = \mathbf{U\Sigma V}^\top$ using singular value decomposition (SVD) [25]. Finally, we obtain a reduced dimensional representation of words from size $O(n)$ to k by selecting the first k columns of $\mathbf{U}$.

Skip-Gram Vectors (SG) The `word2vec` tool [20] is fast and currently in wide use.[2] In this model, each word's Huffman code is used as an input to a log-linear classifier with a continuous projection layer; words within 5 positions are predicted.

Global Context Vectors (GC) These vectors are learned using a recursive neural network that incorporates both local and global (document-level) context features [26]. (We use pre-trained vectors.[3])

Multilingual Vectors (Multi) Faruqui and Dyer [8] learned vectors by first performing SVD on text in different languages, then applying canonical correlation analysis on pairs of vectors for words that align in parallel corpora. (We use pre-trained vectors.[4])

---

[1] i.e., if the gradient update for one training instance is g, the lazy gradient would be kg and would be computed after every k training instances.

[2] `https://code.google.com/p/word2vec`

[3] `http://nlp.stanford.edu/~socherr/ACL2012_wordVectorsTextFile.zip`

[4] `http://www.wordvectors.org/web-eacl14-vectors/de-projected-en-512.txt.gz`

Log-bilinear Vectors (LBL) The log bilinear language model [19] predicts a word token w's vector given the set of words in its context (h), also represented as vectors:

$$p(w \mid h; Q) \propto \exp\left(\sum_{i \in h} q_i^\top q_j + b_j\right) \tag{3}$$

Since it is costly to renormalize over the whole vocabulary, we use *noise constrastive estimation* (NCE) to estimate the parameters of the model [19] using AdaGrad [21] with a learning rate of 0.05.

5 Semantic Lexicons

We use three different semantic lexicons to evaluate their utility in improving the word vectors.

Paraphrase Database. The paraphrase database (PPDB; Ganitkevitch et al., 2013) is a semantic lexicon containing more than 220 million paraphrase pairs of English.[5] Of these, 8 million are lexical (single word to single word) paraphrases. The key intuition behind the acquisition of its lexical paraphrases is that two words in one language that align, in parallel text, to the same word in a different language, should be synonymous. For example, if the words *jailed* and *imprisoned* map to the same word in another language, it may be reasonable to assume they have the same meaning.

In our experiments, we instantiate an edge in E for each lexical paraphrase in PPDB. The lexical paraphrase dataset comes in different sizes ranging from S to XXXL, in decreasing order of paraphrasing confidence and increasing order of size. We chose XL size for our experiments; it produces a graph of 103,000 nodes and 230,000 edges. Since PPDB is an automatically created ontology, it includes a confidence score for each paraphrase. We experimented with (1) setting each (directed) edge weight β_{ij} to be $\mathrm{degree}(i)^{-1}$; and (2) setting β_{ij} to be the average of the two directed confidence scores provided in PPDB, interpreted as $p(w_i \mid w_j)$ and $p(w_j \mid w_i)$. In both settings, all α_i are fixed to 1. We refer to these graphs, respectively, as PPDB and PPDB Weighted.

WordNet WordNet [10] is a large human-constructed semantic lexicon of English words. It groups English words into sets of synonyms called synsets, provides short, general definitions, and records the various semantic relations between synsets. This database is structured in a graph particularly suitable for our task because it explicitly relates concepts with semantically aligned relations such as hypernyms and hyponyms. For example, the word *dog* has a synonym *canine*, a hypernym *puppy* and a hyponym *animal*. We perform two different experiments with WordNet: (1) connecting a word only to synonyms, and (2) including edges for synonym, hypernym, and hyponym relations. Respectively, these lead to graphs of $148,000$ nodes and $560,000$ edges. We refer to these two graphs as WN and WN++, respectively. In both settings, all α_i are set to 1 and β_{ij} to be $\mathrm{degree}(i)^{-1}$.

FrameNet FrameNet [11, 27] is a rich linguistic resource containing information about lexical and predicate-argument semantics in English. Grounded in the theory of frame semantics, it suggests a semantic representation that blends word-sense disambiguation and semantic role labeling. Its expert-constructed taxonomy is organized around general-purpose frames (abstract predicates), each associated with a set of lemmas that can evoke the frame. For example, the frame Cause_change_of_position_on_a_scale is associated with *push*, *raise*, and *growth* (among many others). In our use of FrameNet, two words that group together with any frame are given an edge in E. We get a graph of 11,000 nodes and 210,000 edges. We refer to this graph as FN. All α_i are set to 1 and β_{ij} to be $\mathrm{degree}(i)^{-1}$.

6 Evaluation Benchmarks

We evaluate the quality of our word vector representations on tasks that test how well they capture both semantic and syntactic aspects of the representations along with an extrinsic sentiment analysis task.

[5] http://www.cis.upenn.edu/~ccb/ppdb

Word Similarity We evaluate our word representations on a variety of different benchmarks that have been widely used to measure word similarity. The first one is the **WS-353** dataset [28] containing 353 pairs of English words that have been assigned similarity ratings by humans. The second benchmark is the **RG-65** [29] dataset that contain 65 pairs of nouns. Since the commonly used word similarity datasets contain a small number of word pairs we also use the **MEN** dataset [30] that contains 3,000 word pairs which have been sampled from words that occur at least 700 times in a large web corpus.

We calculate cosine similarity between the vectors of two words forming a test item, and report Spearman's rank correlation coefficient [31] between the rankings produced by our model against the human rankings.

Syntactic Relations (SYN-REL) Mikolov et. al [32] present a syntactic relation dataset composed of analogous word pairs. It contains pairs of tuples of word relations that follow a common syntatic relation. For example, given *walking* and *walked*, the words are differently inflected forms of the same verb. There are nine such different kinds of relations: adjective-adverb, opposites, comaparative, superlative, present-participle, nation-nationality, past tense, plural nouns, and plural verbs. Overall there are 10,675 such syntactic pairs of word tuples.

The task is to find a word *d* that best fits the following relationship: "*a* is to *b* as *c* is to *d*," given *a*, *b*, and *c*. We use the vector offset method [20], computing $q = q_a - q_b + q_c$ and returning the vector from Q which has the highest cosine similarity to q.

Synonym Selection (TOEFL) The synonym selection task is to select the semantically closest word to a target from a list of candidates. The dataset we use on this task is the TOEFL dataset [33] which consists of a list of target words that appear with 4 candidate lexical substitutes each. The dataset contains 80 such questions. An example is "*rug* $\rightarrow$ {*sofa*, *ottoman*, *carpet*, *hallway*}", with *carpet* being the most synonym-like candidate to the target.

Sentiment Analysis (SA) Socher et. al, [34] created a treebank containing sentences annotated with fine-grained sentiment labels on phrases and sentences from movie review excerpts. The coarse-grained treebank of positive and negative classes has been split into training, development and test datasets containing 6,920, 872, and 1,821 sentences, respectively. We train an ℓ_2-regularized logistic regression classifier on the average of the word vectors of a given sentence to predict the coarse-grained sentiment tag at the sentence level, and report the test-set accuracy of the classifier.

7 Experiments & Results

We now turn to an expiration of whether the static retrofitting approach is effective, or if incorporating lexical knowledge during training is more effective. Using a corpus of English news from WMT-2011,[6] we trained word vectors of length 80 using the LSA, LBL, and SG methods. After normalization the corpus contained 360 million word tokens and 180,000 word types. As noted in §4, we also considered pre-trained, publicly released vectors trained using GC and Multi. GC vectors are trained on 990 million tokens from Wikipedia and are 50-dimensional. Multi-lingual vectors are also trained on the WMT-2011 data and are 512-dimensional.

7.1 Retrofitting

We use Equation 1 to retrofit each of the five sets of word vectors (§4) using each of the five graphs derived from semantic lexicons (§5).

Results Figure 2 shows the absolute changes in performance (Spearman's correlation ratio or accuracy, as appropriate) on different tasks (as columns) with different semantic lexicons (as rows). Colored bars correspond to the five different sets of input vectors. All of the lexicons offer improvements on the word similarity tasks (the first three columns). FrameNet's performance is weaker, in some cases leading to worse performance (e.g., with SG and Multi vectors). On the TOEFL task,

[6] `http://www.statmt.org/wmt11`

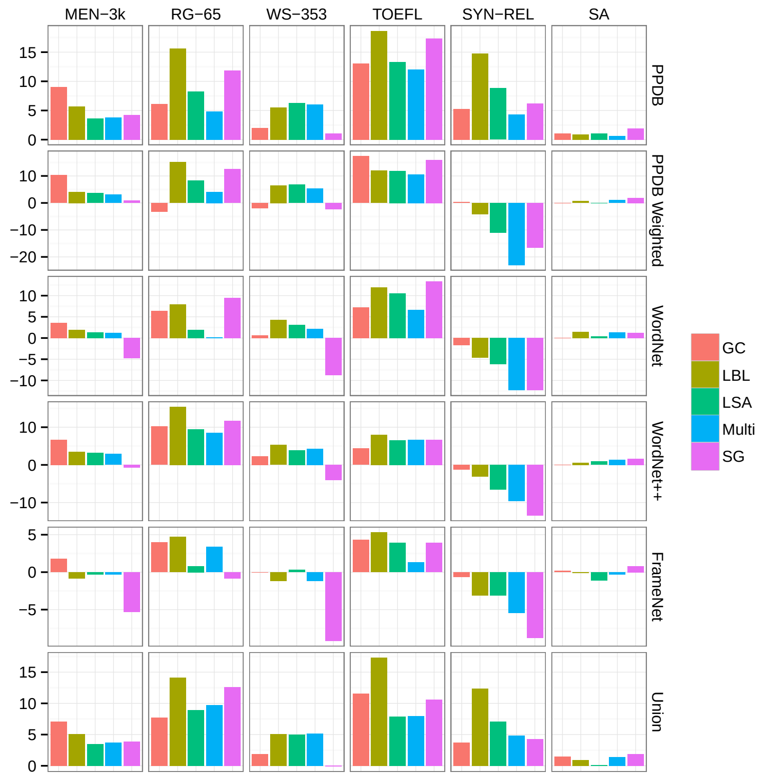

Figure 2: Absolute percentage changes with retrofitting. Each row is a semantic lexicon; each column is a task (the first three are measured in Spearman's correlation ratio, the last three in accuracy; in both cases, higher is better). The five colored bars correspond to different vector learning methods.

we get huge improvements of the order of 10 absolute points in accuracy for all ontologies except for FrameNet.

For the extrinsic sentiment analysis task, we get improvements using all the ontologies and achieve the highest improvement of absolute 1.4 points in accuracy for the multilingual vectors over the baseline. This increase is statistically significant ($p < 0.01$) according to a McNemar's test.

The weighted variant of PPDB offers no benefit over the simpler PPDB graph. We believe that FrameNet does not perform as well as the other lexicons is that its frames group words based on abstract concepts; often words with seemingly distantly related meanings (e.g., *push* and *growth*) can evoke the same frame. In summary, we find overall that PPDB gives the best graph among those tested (across tasks and vectors), LBL vectors tend to improve more than others and the TOEFL task tends to receive the greatest benefit from retrofitting.

Lexicon Ensemble We also considered an ensemble, in which the graph is the union of the WordNet and PPDB lexicons. The result is shown in the bottom row of Figure 2. On average, it performs slightly worse than the best component.

7.2 Lexicon Information during Training

We next consider the approach to training word vectors with the lexical knowledge incorporated as a prior (§3). Here we consider length-80 LBL vectors—for which we have our own training implementation to modify, and which performed reasonably well in the retrofitting experiments (§7.1)—and the PPDB graph. We consider the **lazy** and **periodic** algorithms described in §3. For the **lazy** method we update the prior every k = 100,000 words[7] and test for different values of prior

[7]Experiments with $k \in [10000, 50000]$ yielded almost similar results.

Method	k/γ	MEN-3k	RG-65	WS-353	TOEFL	SYN-REL	SA
Baseline	$k=\infty, \gamma=0$	58.0	42.7	53.6	66.7	31.5	72.5
	$\gamma=1$	57.6	46.9	54.2	66.6	32.1	73.7
Lazy	$\gamma=0.1$	58.7	50.8	54.0	65.3	32.2	73.3
	$\gamma=0.01$	58.7	52.2	55.3	69.3	33.4	72.9
	$k=100$M	61.8	61.1	57.2	78.7	36.3	73.8
Periodic	$k=50$M	61.4	**62.2**	58.0	85.3	32.1	**74.4**
	$k=25$M	58.5	60.8	56.3	**88.0**	27.8	73.3
Retrofitting	–	**63.7**	58.3	**59.1**	85.3	**46.2**	73.4

Table 1: Performance of variants of including PPDB information in LBL vectors. Spearman's correlation (3 left columns) and accuracy (3 right columns) on different tasks. Bold indicates best result across all vector types.

strength $\gamma \in \{1, 0.1, 0.01\}$. For the **periodic** method, we update the word vectors using Eq. 1 every $k \in \{25, 50, 100\}$ million words.

Results See Table 1. For **lazy**, $\gamma = 0.01$ performs best, but the method is in most cases not highly sensitive to γ's value. For **periodic**, which overall leads to greater improvements over the baseline than **lazy**, $k = 50$M performs best, although all other values of k also outperform the the baseline. The last row shows the results obtained using retrofitting for comparison and it can be seen that retrofitting performs either better or competitively to using semantic knowledge during training—this is an important result since retrofitting is more flexible as it is applicable to any word vector model.

7.3 Multilingual Evaluation

Language, Task	Original	Retrofitted
DE, RG-65	33.1	52.2
FR, WS-353	46.7	60.6
ES, MC-30	45.9	53.3

Table 2: Spearman's correlation for word similarity evaluation using the original and retrofitted German, French and Spanish word vectors.

We tested our method on three additional languages: German, French, and Spanish. We used the Universal WordNet [35], an automatically constructed multilingual lexical knowledge base based on WordNet.[8] It contains words connected via different lexical relations to other words both in and across languages. We construct separate graphs for different languages (i.e., only linking words to other words in the same language) and apply retrofitting to each.

Since not many word similarity evaluation benchmarks are available for other languages we tested our baseline and improved vectors on one benchmark per language. We used RG-65 [36], WS-353 [37] and MC-30 [38] for German, French and Spanish respectively. These benchmarks were created by translating the corresponding English benchmarks word by word manually. We trained SG vectors for each language of length 80. We again used the WMT-2011 monolingual news corpus for each language to train its word vectors and evaluate word similarity on these tasks before and after retrofitting. Table 2 shows the results that strongly indicate that our method generalizes across languages. We get high improvements in the Spearman's correlation coefficient on the word similarity tasks for the three languages for three different kinds of vectors.

8 Conclusion

We have proposed a simple, effective and fast method named **retrofitting** to improve word vectors using word relation knowledge found in semantic lexicons constructed either automatically or by humans. Retrofitting is used as a post-processing step to improve vector quality and is simpler to use than other approaches that use semantic information while training. It can be used for improving vectors obtained from any word vector training model and performs better than current state-of-the-art approaches to semantic enrichment of word vectors. We validated the applicability of our method across tasks, semantic lexicons, and languages.

[8] `http://www.mpi-inf.mpg.de/yago-naga/uwn`

Acknowledgements

This research was supported in part by the National Science Foundation under grants IIS-1143703, IIS-1147810, and IIS-1251131; by IARPA via Department of Interior National Business Center (DoI/NBC) contract number D12PC00337; and by DARPA under grant FA87501220342. Part of the computational work was carried out on resources provided by the Pittsburgh Supercomputing Center. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, DARPA, or the U.S. Government.

References

[1] Peter D. Turney. Similarity of semantic relations. *Comput. Linguist.*, 32(3):379–416, September 2006.

[2] Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of NAACL*, 2009.

[3] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proc. of ACL*, 2010.

[4] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 1990.

[5] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of ICML*, 2008.

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

[7] Karl Moritz Hermann and Phil Blunsom. Multilingual models for compositional distributed semantics. *arXiv preprint arXiv:1404.4641*, 2014.

[8] Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *Proceedings of EACL*, 2014.

[9] Sebastian Padó and Mirella Lapata. Dependency-based construction of semantic space models. *Comput. Linguist.*, 33(2):161–199, June 2007.

[10] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 1995.

[11] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. ACL '98, 1998.

[12] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. PPDB: The paraphrase database. In *Proceedings of NAACL*, 2013.

[13] Ross Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. AMS, 1980.

[14] Yoshua Bengio, Olivier Delalleau, and Nicolas Le Roux. Label propagation and quadratic criterion. In *Semi-Supervised Learning*. 2006.

[15] Amarnag Subramanya, Slav Petrov, and Fernando Pereira. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of EMNLP*, 2010.

[16] Dipanjan Das and Slav Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proc. of ACL*, 2011.

[17] Dipanjan Das and Noah A. Smith. Semi-supervised frame-semantic parsing for unknown predicates. In *Proc. of ACL*, 2011.

[18] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *JMLR*, 2003.

[19] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of ICML*, 2012.

[20] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[21] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. Technical Report UCB/EECS-2010-24, Mar 2010.

[22] Bob Carpenter. Lazy sparse stochastic gradient descent for regularized multinomial logistic regression., 2008.

[23] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information, and lexicography. *Comput. Linguist.*, 16(1):22–29, March 1990.

[24] Peter D. Turney. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *ECML*, 2001.

[25] Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[26] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*, 2012.

[27] Charles Fillmore, Christopher Johnson, and Miriam Petruck. *International Journal of Lexicography*, 2003.

[28] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. In *WWW*, New York, NY, USA, 2001.

[29] Herbert Rubenstein and John B. Goodenough. Contextual correlates of synonymy. *Commun. ACM*, 8(10):627–633, October 1965.

[30] Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. Distributional semantics in technicolor. In *Proceedings of ACL*, 2012.

[31] Jerome L. Myers and Arnold D. Well. *Research Design & Statistical Analysis*. Routledge, 1995.

[32] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL*, 2013.

[33] Thomas K Landauer and Susan T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 1997.

[34] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, 2013.

[35] Gerard de Melo and Gerhard Weikum. Towards a universal wordnet by learning from combined evidence. In *Proceedings of CIKM*, 2009.

[36] Iryna Gurevych. Using the structure of a conceptual network in computing semantic relatedness. In *Proceedings of IJCNLP*, 2005.

[37] Colette Joubarne and Diana Inkpen. Comparison of semantic similarity for different languages using the google n-gram corpus and second- order co-occurrence measures. In *Proceedings of CAAI*, 2011.

[38] Samer Hassan and Rada Mihalcea. Cross-lingual semantic relatedness using encyclopedic knowledge. In *Proc. of EMNLP*, 2009.