Adjusting Word Embeddings with Semantic Intensity Orders

Joo-Kyung Kim[†], **Marie-Catherine de Marneffe**[‡], **Eric Fosler-Lussier**[†]
[†]Department of Computer Science and Engineering,
[‡]Department of Linguistics,
The Ohio State University,
Columbus, Ohio 43210, USA
kimjook@cse.ohio-state.edu, mcdm@ling.ohio-state.edu,
fosler@cse.ohio-state.edu

Abstract

Semantic lexicons such as WordNet and PPDB have been used to improve the vector-based semantic representations of words by adjusting the word vectors. However, such lexicons lack semantic intensity information, inhibiting adjustment of vector spaces to better represent semantic intensity scales. In this work, we adjust word vectors using the semantic intensity information in addition to synonyms and antonyms from WordNet and PPDB, and show improved performance on judging semantic intensity orders of adjective pairs on three different human annotated datasets.

1 Introduction

Word embedding models that represent words as real-valued vectors have been directly used in word-level NLP tasks such as word similarity (Mikolov et al., 2013b), antonym detection (Ono et al., 2015; Pham et al., 2015; Chen et al., 2015), knowledge relations (Toutanova et al., 2015; Socher et al., 2013; Bordes et al., 2013), and semantic scale inference (Kim and de Marneffe, 2013). Word embedding models such as Word2Vec (continuous bag-of-words (CBOW) and skip-gram) (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014), widely used to generate word vectors, are trained following the distributional hypothesis (Harris, 1954) which assumes that the meaning of words can be represented by their context.

However, word embedding models based solely on the distributional hypothesis often place words improperly in vector spaces. For example, in a vector space, a word and its antonym should be sufficiently far apart, but they can be quite close because they can have similar contexts in many cases.

For better semantic representations, different approaches using semantic lexicons as well as lexical knowledge to adjust word vectors have recently been introduced. Faruqui et al. (2015) adjusted each word vector to be in the middle between the initial position and its synonymous words. Mrkšić et al. (2016) used max-margin approaches to adjust each word vector with synonyms and antonyms while keeping the relative similarities to the neighbors. While these two approaches are post-processing models that adjust preexisting word vectors, Ono et al. (2015), Pham et al. (2015), and Liu et al. (2015) jointly train models that augment the skip-gram (Mikolov et al., 2013a) objective function to include knowledge from semantic lexicons. The common goal in these approaches is to make semantically close words closer and semantically distant words farther apart while keeping each word vector not to be too far from the original position. Although the joint training models can even indirectly adjust words that are not listed in the semantic lexicons (Pham et al., 2015), the post-processing models are much more efficient and can be applied to word vectors from any kinds of models, which can eventually perform better than the joint training models (Mrkšić et al., 2016).

Although Faruqui et al. (2015), Mrkšić et al. (2016), Ono et al. (2015), Pham et al. (2015), and Liu et al. (2015)'s adjustment approaches have been shown to represent word semantics better in vector spaces, their coarse modeling of words as synonyms or antonyms may be insufficient for modeling words lying along a semantic intensity scale. For example, assume that "great" is erroneously between "bad" and "good" in a vector space ("bad" should be closer to "good" than "great"). Since semantic lexicons such as Word-

Proceedings of the 1st Workshop on Representation Learning for NLP, pages 62–69,
Berlin, Germany, August 11th, 2016.

Net (Fellbaum, 1998) and the Paraphrase Database (PPDB) (Pavlick et al., 2015) only inform us that "good" and "great" are semantically similar and "good" is semantically opposite to "bad", adjusting word vectors with those semantic lexicons does not permit to retrieve the appropriate semantic intensity ordering: bad < good < great.

Accurate representation of such semantic intensity scales can help correct processing in downstream tasks that require robust textual understanding. For instance, given an assertion such as *the movie is outstanding*, statements that contain a semantically weaker expression (e.g., *the movie is good*, *the movie is okay*) are entailed, whereas *the movie is okay* does not entail that *the movie is outstanding*. Similarly, correct information about semantic scales can also provide accurate inferences: when answers to a yes/no question that contains a gradable adjective does not explicitly contain a *yes* or a *no*, we can derive the intended answer by figuring out whether the answer entails or implicates the question (Horn, 1972; Hirschberg, 1985; de Marneffe et al., 2010). For example, for the question *Was the talk good?*, if the answer is *It was excellent*, the answer entails "yes", but if the answer is *It was okay*, "no" will be implied.

To deal with the representation of semantic intensity scales, we infer semantic intensity orders with de Melo and Bansal (2013)'s approach and then use the intensity orders to adjust the word vectors. Evaluating on three different human annotated datasets, we show that the adjustment with intensity orders in addition to adjustments with synonyms and antonyms performs best in representing semantic intensities.

2 Adjusting word embeddings with semantic lexicons

In this study, we start from one of three different off-the-shelf word vector types as a baseline for our studies: GloVe, CBOW, and Paragram-SL999 (Wieting et al., 2015); we adjust each of these sets of vectors with a variety of contrastive methods. Our first contrastive system is a baseline using synonyms and antonyms ("syn&ant") following Mrkšić et al. (2016)'s approach, which adjusts word vectors so that the sum of the following three max-margin objective functions are minimized.

Adjusting with antonyms We adjust word vectors so that the cosine similarity between each word and its antonyms is zero or lower:

$$AF\left(V\right)=\sum_{(u,w)\in A}\tau\left(\cos\left(v_u,v_w\right)\right), \qquad (1)$$

where $\tau\left(x\right)=\max\left(0,x\right)$, V is the vocabulary matrix, A is the set of antonym pairs, and v_i is the i-th row of V (i-th word vector). The antonym pairs consist of the antonyms from WordNet and *Exclusion* relations from PPDB word pairs.

Adjusting with synonyms We let the cosine similarities between each word and its synonyms be increased:

$$SC\left(V\right)=\sum_{(u,w)\in S}\tau\left(1-\cos\left(v_u,v_w\right)\right), \qquad (2)$$

where S is the set of synonym pairs. The synonym pairs consist of the *Equivalence* relations from PPDB word pairs.

Keeping the similarity to the initial neighboring words We encourage the cosine similarity between the initial vectors of each word and a neighbor word to be equal to or higher than the current cosine similarity between them:

$$KN\left(V,V^0\right)=\sum_{i=1}^{N}\sum_{j\in N(i)}\tau\left(\cos\left(v_i,v_j\right)-\cos\left(v_i^0,v_j^0\right)\right), \qquad (3)$$

where V^0 is the initial vocabulary matrix, N is the vocabulary size, and $N\left(i\right)$ is the set of the initial neighbors of the i-th word. Word pairs with cosine similarities equal to or higher than 0.8 are regarded as neighbors.

The objective function for the word vector adjustment is represented as the sum of the three terms:

$$C\left(V,V^0\right)=AF\left(V\right)+SC\left(V\right)+KN\left(V,V^0\right) \qquad (4)$$

This function is minimized with stochastic gradient descent with learning rate 0.1 for 20 iterations.

3 Adjusting word embeddings with semantic intensity orders

In order to better model semantic intensity ordering, we augment the synonym and antonym adjusted model with semantic intensity information to adjust word vectors. We first cluster semantically related words, infer semantic intensity orders of words in each cluster, and then adjust word vectors based on the intensity orders.

3.1 Clustering words for intensity ordering

de Melo and Bansal (2013) used WordNet dumbbells (Gross and Miller, 1990), each of which consists of an adjective antonym pair and each adjective's synonyms, to define a set of words along a semantic intensity scale. Words in each half of a dumbbell form a cluster. This clustering is effective since synonyms are semantically highly related but their intensities may be different. However, this approach can only cluster words listed in WordNet.

Shivade et al. (2015) clustered word vectors from the CBOW model with k-means++ clustering (Arthur and Vassilvitskii, 2007). This approach depends on the current word vector placement and does not require semantic lexicons. However, a word can only belong to one cluster since k-means++ is a hard clustering, thus causing issues with polysemous words. For example, “hot” is both on the temperature scale (e.g., *It's hot today*) and on the interestingness scale (e.g., *It's a hot topic*). If “hot” is adjusted for the former scale, “hot” may not properly be placed on the latter scale. Another issue of using clustering algorithms is that unrelated or antonymous words can belong to a cluster, which may hinder correct intensity ordering.

We evaluated both clustering approaches and their combination to cluster words for intensity orders. In Table 2, by default, WordNet dumbbells and *Equivalence* relations of PPDB word pairs are used as the intensity clusters. “kmeans only” denotes that only clusters from k-means++ are used, and “+kmeans” means that WordNet, PPDB, and clusters from k-means++ are used altogether. Following Shivade et al. (2015), when clustering with k-means++, we set k to be 5,798, which is the number of all observed adjectives (17,394) divided by 3 so that the average number of adjectives in a cluster is 3.

3.2 Inferring intensity ordering

We follow de Melo and Bansal (2013)'s approach to order the adjectives in each cluster. For every possible pair of adjectives in the cluster, we search for regular expressions like “$\langle * \rangle$ but not $\langle * \rangle$” in Google N-gram (Brants and Franz, 2006). These patterns give us the direction of the ordering between the adjectives. For example, if “good but not great” appears frequently in Google N-gram, we infer that “great" is semantically stronger than “good".[1] Once we have the intensity differences of adjective pairs in a cluster, mixed integer linear programming (MILP) is used for optimal ordering of all the adjectives in the cluster given the pairwise intensity information of the adjective pairs, following de Melo and Bansal (2013).

3.3 Adjusting word vectors based on intensity orders

Now that we have word clusters whose constituent words are ordered according to their semantic intensities, we adjust the word vectors in two ways, as follows.

3.3.1 Adjusting words with the same intensity order to be closer

When intensity orders are assigned to words in a cluster, different words can have the same rank. For example, given a word cluster {“interesting”, “provocative”, “exciting”, “sexy”, “exhilarating”, “thrilling”}, both “exhilarating” and “thrilling” are assigned the highest order, and “exciting” and “sexy” are assigned the second highest order. Since words in a same cluster are considered to be very close in both the meaning and the intensity, it is desirable to let them to be similar in the vector space. Therefore, we formulate a max-margin function:

$$SO\left(V\right) = \sum_{(u,w)\in E} \tau \left(1 - \cos\left(v_u, v_w\right)\right), \qquad (5)$$

where E is the word pairs of the same intensities from the intensity clusters.

3.3.2 Adjusting weaker/stronger word pairs based on antonyms

For two similar words with different intensities (e.g., “good” and “great”), the similarity between the weaker word vector and its antonym vector should be higher than the similarity between the stronger word vector and the antonym vector. Figure 1 shows an example of word vectors which are wrongly ordered.

To reduce wrong orderings, we formulate a

[1] Shivade et al. (2015) used Tregex (Levy and Andrew, 2006) to extract patterns including more words but it is not necessary when we extract patterns from phrases consisting of less or equal to five words.

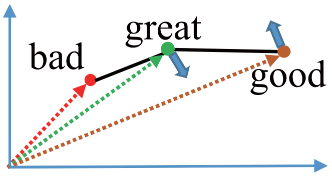

Figure 1: An example of incoherent word vector positions, where “bad” should be closer to “good” than “great” but the similarity between “bad” and “good” is lower than the similarity between “bad” and “great”.

max-margin function:

$$AO(V) = \sum_{(w,a)\in A} \sum_{s\in Str(w)} \tau\left\{\cos(v_s, v_a) - \cos(v_w, v_a)\right\}, \quad (6)$$

where A is the set of antonym pairs and $Str(w)$ is a set of words semantically stronger than w. By minimizing this function, out-of-order vectors are adjusted so that the stronger word vector gets farther from the antonym vector and the weaker word vector gets closer to the antonym vector.

Both equations 5 and 6 can be either solely used or summed to others like equation 4 to serve as a term of the objective function.

4 Evaluation

We evaluate the representation of semantic intensities on the three following human-annotated datasets.

4.1 WordNet synset pairs

We obtained a dataset of 670 synonymous adjective pairs coming from synsets in WordNet from Christopher Potts. Each adjective pair was annotated for intensity order on Mechanical Turk. For each adjective pair <A, B> (e.g., “good” and “great”), ten different Turkers were asked to judge whether A is semantically stronger than B, B is semantically stronger than A, or A is equal to B. For consistency of annotation with the other datasets, we mapped “A is semantically stronger than B" to “no”, “B is semantically stronger than A" to “yes”, and “A is equal to B" to “uncertain”.

For 77.3% of adjective pairs, at least 6 out of the 10 Turkers agreed with each other on the same annotation. Table 1 gives a breakdown of how often Turkers agree with each other. The inter-annotator agreement (Fleiss’ kappa) of this dataset is 0.359. Note that Fleiss’ kappa is a very conservative measure given the partial order in the annotation, which is not taken into account in Fleiss’ kappa.

Max # Turkers agreeing	Coverage (%)
10	17.5
9	17.2
8	13.3
7	14.6
6	14.9
5	16.7
4	6

Table 1: Percentage of adjective pairs and the maximum number of Turkers who agree with each other on the annotation.

4.2 Indirect question-answer pairs (IQAP)

IQAP (de Marneffe et al., 2010) is a corpus consisting of 127 indirect question-answer pairs in which both the question and the answer contain a gradable adjective (*Is Obama qualified? I think he’s young.*). For each pair, 30 Turkers decided whether the answer implies a “yes”, “no” or “uncertain" response to the question. A majority “yes” response implies that the adjective in the question entails the adjective in the answer.

The ordering between the adjectives in the question and in the answer can be used to infer a “yes" or “no" answer: if the adjective in the answer is semantically equivalent or stronger to the adjective in the question, we infer a “yes” answer (*Was the movie good? It was excellent.*); if not, we infer a “no” answer.

4.3 Word intensity orders in clusters

We also use the test set from de Melo and Bansal (2013) consisting of 507 pairs of adjectives in 88 clusters annotated by two native English speakers for intensity ordering. From this set, we generated all the possible adjective pairs from the ordered list in a cluster. For example, for “known” < “famous” < “legendary” in the test set, we generated “known” < “famous”, “known” < “legendary”, and “famous” < “legendary”.

4.4 Evaluation results

In our evaluation of the semantic orderings of adjective pairs, we decide which adjective in a pair <A, B> is semantically stronger following Kim and de Marneffe (2013)’s approach. First, we look

Adjustment methods	WordNet synset pairs			IQAP			de Melo & Bansal (2013)		
	GloVe	CBOW	Pgrm	GloVe	CBOW	Pgrm	GloVe	CBOW	Pgrm
baseline	0.5614	0.5092	0.5224	0.7044	0.7016	0.7591	0.9468	0.9347	0.9803
syn&ant	0.5106	0.5516	0.5572	**0.8143**	**0.8045**	0.8307	0.9632	0.9444	0.9791
same_ord (kmeans only)	**0.5762**	0.5163	0.5196	0.7044	0.7016	0.7473	0.9480	0.9359	0.9791
same_ord, diff_ord	0.5505	0.5331	0.5167	0.7119	0.6889	0.7718	0.9456	0.9371	0.9701
syn&ant,same_ord	0.5364	0.5639	0.5782	0.7922	0.7818	0.8284	0.9632	0.9492	0.9803
syn&ant,diff_ord	0.5300	0.5551	0.5765	**0.8143**	0.7922	0.8307	0.9735	0.9539	**0.9825**
syn&ant,same_ord,diff_ord	0.5467	0.5730	**0.5960**	**0.8143**	0.8033	**0.8395**	**0.9758**	0.9539	**0.9825**
syn&ant,same_ord,diff_ord (kmeans only)	0.5186	0.5516	0.5729	0.8033	**0.8045**	0.8194	0.9609	0.9468	0.9803
syn&ant,same_ord,diff_ord (+kmeans)	0.5512	**0.5828**	**0.5960**	0.8033	0.8033	**0.8395**	0.9735	**0.9609**	0.9814

Table 2: F1 scores for determining semantic intensity ordering on three datasets, across three baseline models (GloVe, CBOW, Paragram), using different compositions of adjustment techniques, including **syn**onyms, **ant**onyms, **same** intensity **ord**ers, and **diff**erent intensity **ord**ers.

Datasets	# pairs	# syn	# ant
WordNet synset pairs	670	79	0
IQAP	127	7	9
de Melo & Bansal	507	54	1

Table 3: The numbers of total adjective pairs, synonymous pairs, and antonymous pairs for each dataset.

for an antonym of A.[2] Then, we check whether the word vector for B is more similar to the vector for A than to the vector for A's antonym, or whether the vector for B is more similar to the vector for A's antonym. We infer a "yes" answer in the former case, and a "no" in the other case. If A has more than one antonym, we select the antonym that is most collinear with the vectors for A and B assuming that the most collinear antonym is most semantically related to A and B.

Table 2 shows the F1 scores of different combinations of the adjustments on the three datasets,[3] whereas Table 3 shows the number of total adjective pairs in each dataset, as well as the number of pairs in which both adjectives are synonyms (*Equivalence* relations from PPDB) and the number of pairs in which both adjectives are antonyms (*Exclusion* relations from PPDB and antonyms from WordNet).

Expanding on the results in Table 2, as the baselines, we used three different 300 dimensional off-the-shelf word vectors: GloVe,[4] CBOW,[5] and Paragram-SL999.[6] Following Mrkšić et al. (2016), for each of the word vector sets, we extracted word vectors corresponding to the 76,427 most frequent words from Open-Subtitles.[7]

Table 4 indicates whether the differences in performance of the adjustment methods in Table 2 are statistically significant (McNemar's χ^2 test with p-value < 0.05). In the table, "merged" columns are the results of the concatenation of all the datasets. For each comparison, '+' denotes that the performance of the latter is significantly higher than that of the former, and '-' denotes the opposite, whereas no value indicates that the difference in performance is not statistically significant. For Paragram vectors, only one case ("baseline" vs "syn&ant,same_ord") is significantly different.

In Table 2, "baseline" shows the performance of the baseline word vectors without any adjustments. Since Paragram-SL999 are optimized to perform best on evaluating SimLex-999 dataset, the baseline performance of Paragram-SL999 on SimLex-999 as well as two of the other datasets are noticeably better than word vectors from GloVe and CBOW.

In "syn&ant", corresponding to the optimization with equation 4, 15,509 words are adjusted with the synonyms and 6,162 words are adjusted with the antonyms. This adjustment significantly

[2] If there are no antonyms of A in WordNet, we obtain antonyms from Roget's thesaurus (Kipfer, 2009).

[3] For simplicity of the evaluation in vector spaces, we calculate F1 scores without "uncertain" cases.

[4] Available from http://nlp.stanford.edu/projects/glove/

[5] Available from https://code.google.com/p/word2vec/

[6] Available from https://drive.google.com/file/d/0B9w48e1rj-MOck1fRGxaZW1LU2M/

[7] Available from invokeit.wordpress.com/frequency-word-lists

Compared adjustment methods	GloVe				CBOW			
	WN	IQAP	dM&B	merged	WN	IQAP	dM&B	merged
baseline v. syn&ant	-	+	+			+		
baseline v. syn&ant,same_ord,diff_ord	-	+	+			+	+	+
syn&ant v. syn&ant,same_ord,diff_ord			+	+				
baseline v. syn&ant,same_ord,diff_ord (+kmeans)		+	+	+		+	+	
syn&ant v. syn&ant,same_ord,diff_ord (+kmeans)	+			+			+	+

Table 4: McNemar's χ^2 test results (p-value < 0.05) for different methods of GloVe/CBOW adjustments across WordNet synset (WN), IQAP, and de Melo & Bansal (dM&B) datasets, as well as concatenating the three datasets (merged). For x v. y, ‘+’ denotes that y's score is significantly higher than that of x, ‘’-’ denotes the opposite, and no value denotes that the difference is not statistically significant.

improves the performance of CBOW vectors and Paragram vectors on the IQAP and de Melo and Bansal (2013)'s datasets. Specifically, for the IQAP dataset, where many of the pairs are either synonyms or antonyms, “syn&ant” showed better performance than including adjustments with semantic intensity orders. However, this adjustment makes GloVe vectors yield significantly worse performance on the WordNet synset pair dataset. This shows that the adjustment with just synonyms and antonyms can worsen the representation of subtle semantics considering intensities. In this case, using just the adjustment with semantic intensity orders can be helpful. “same_ord (kmeans only)”, corresponding to equation 5, adjusts word vectors by just making vectors of words with the same intensity order to be more similar without using synonyms and antonyms. For GloVe vectors, “same_ord (kmeans only)” showed the highest score for the WordNet synset pair dataset. For adjustments with semantic intensity orders, 616 words are adjusted when WordNet dumbbells and *Equivalence* relations from PPDB word pairs are used as the clusters. When clusters from k-means++ are used, several hundreds of words are adjusted, where the adjusted words vary depending on the vector space for each iteration.

For the WordNet synset pair dataset and de Melo and Bansal (2013)'s dataset, where the subtle semantic intensity differences are more critical, using synonyms, antonyms, and semantic intensity orders altogether (“syn&ant,same_ord,diff_ord”) showed significantly higher scores than “syn&ant” in many settings. Here, “diff_ord” corresponds to equation 6.

Table 5 shows the adjective pairs whose intensity judgements were changed by including adjustments with semantic intensity orders. The pairs are from the WordNet synset pairs and GloVe vectors were used as the baseline. “baseline” is compared to “same_ord (kmeans only)” in the first column and “syn&ant” is compared to “syn&ant,same_ord,diff_ord(+kmeans)”. In both cases, we observe that some of the incorrectly judged pairs are corrected when adding the adjustment with semantic intensity orders. In these cases, there were no pairs that were correctly judged by the adjustments without semantic intensity orders but incorrectly judged with semantic intensity orders.

baseline v. same_ord (kmeans only)	syn&ant v. syn&ant, same_ord,diff_ord(+kmeans)
satisfactory < superb	mediocre < severe
unfavorable < poor	troublesome < rocky
crazy < ardent	upfront < blunt
outspoken < expansive	solid < redeeming
sad < tragic	warm < uneasy
deserving < sacred	valuable < sacred

Table 5: Adjective pairs whose incorrect decisions with the former models are corrected by the latter models. For those model comparisons, there were no pairs that were correctly judged with the former models but not with the latter models.

Since the numbers of adjectives pairs in the datasets and the numbers of words that are adjusted with semantic intensity orders are small, not all the cases comparing the adjustments using just synonyms and antonyms to the adjustments including semantic intensity orders were significant for p-value < 0.05, as shown in Table 4. However, since many of them are slightly insignificant (like p-value=0.07) and the scores noticeably increased in many cases, using semantic intensity orders for the adjustments seem promising.

In addition, to show that the adjustments are not harmful for the representation of the general semantics of the words, we also evaluated on SimLex-999 (Hill et al., 2015), where 999 word

	GloVe	CBOW	Pgrm
baseline	0.4453	0.4567	0.6920
syn&ant	0.5969	0.5768	0.7268
same_ord (kmeans only)	0.4420	0.4585	0.6926
same_ord, diff_ord	0.4522	0.4613	0.6872
syn&ant,same_ord	0.5969	0.5768	0.7261
syn&ant,diff_ord	0.5958	0.5767	**0.7274**
syn&ant,same_ord,diff_ord	0.5962	**0.5773**	0.7271
syn&ant,same_ord,diff_ord (kmeans only)	**0.5980**	0.5769	0.7269
syn&ant,same_ord,diff_ord (+kmeans)	0.5956	0.5771	0.7273

Table 6: Spearman's ρ on SimLex-999.

pairs were annotated on Mechanical Turk to score the degree of semantic similarities. This dataset has been widely used to evaluate the quality of semantic representations of words.

Table 6 shows Spearman's ρ scores on the SimLex-999 dataset for the different adjustment methods. Since SimLex-999 dataset is not directly related to semantic intensities compared to the other evaluation datasets, there were no significant gains for the adjustments with semantic intensity orders. However, no significant drops indicate that the adjustments with semantic intensity orders are not harmful for the representation of general word semantics.

5 Discussion and Conclusion

In this work, we adjusted word vectors with inferred semantic intensity orders as well as information from WordNet and PPDB, and showed that adjusting word vectors with semantic intensity orders, synonyms, and antonyms altogether showed the best performance for all the three datasets we evaluated on. Using the semantic intensity orders for adjusting word vectors can help represent semantic intensities of words in vector spaces. In addition, we showed the adjustments including semantic intensity orders are not harmful for the representation of semantics in general by evaluating on SimLex-999.

In future work, we plan to investigate clustering techniques beyond WordNet dumbbells and k-means++ as preprocessing in the semantic ordering. The clusters using WordNet dumbbells depend on a preexisting semantic lexicon that may not cover all the semantically related words. With k-means++, clusters may contain semantically opposite words and a word can belong to only one cluster. As both techniques have limitations, by using another clustering method, the performance could be further improved. In addition, we plan to use larger corpora than Google N-gram so that we can find more intensity orderings within clusters. We can also further improve the performance by using semantic intensity information from other linguistic resources. For example, given a list of base, comparative, and superlative forms of adjectives and adverbs, we can let those adjectives aligned more correctly in vector spaces. We can also use word definitions from dictionaries. For example, from *American Heritage Dictionary*, one of the definitions of "furious" is "extremely angry" and one of that of "excellent" is "exceptionally good". Therefore, by analyzing word definitions, we can obtain word intensity orders.

Acknowledgments

We are very grateful to Christopher Potts for sharing with us the judgements on the WordNet synset pairs that he gathered on Mechanical Turk. We also thank our anonymous reviewers for their comments.

References

David Arthur and Sergei Vassilvitskii. 2007. K-means++: The advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1027–1035.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795.

Thorsten Brants and Alex Franz. 2006. The Google Web 1T 5-gram Version 1.1.

Zhigang Chen, Wei Lin, Qian Chen, Xiaoping Chen, Si Wei, Hui Jiang, and Xiaodan Zhu. 2015. Revisiting word embedding for contrasting meaning. In *Proceedings of ACL*, pages 106–115.

Marie-Catherine de Marneffe, Christopher D. Manning, and Christopher Potts. 2010. Was it good? It was provocative. Learning the meaning of scalar adjectives. In *Proceedings of the 48th Meeting of the Association for Computational Linguistics (ACL)*, pages 167–176.

Gerald de Melo and Mohit Bansal. 2013. Good, Great, Excellent: Global Inference of Semantic Intensities. *Transactions of the Association for Computational Linguistics (TACL)*, 1:279–290.

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*, pages 1606–1615.

Christiane Fellbaum. 1998. *WordNet: An electronic lexical database*. MIT Press.

Derek Gross and Katherine J. Miller. 1990. Adjectives in WordNet. *International Journal of Lexicography*, 3(4):265–277.

Zellig Harris. 1954. Distributional structure. *Word*, 10:146–162.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *Computational Linguistics*, 41(4):665–695.

Julia B. Hirschberg. 1985. *A Theory of Scalar Implicature*. Ph.D. thesis, University of Pennsylvania.

Lawrence Horn. 1972. *On the Semantic Properties of Logical Operators in English*. Bloomington, Indiana: Indianan University Linguistics Club.

Joo-Kyung Kim and Marie-Catherine de Marneffe. 2013. Deriving Adjectival Scales from Continuous Space Word Representations. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, pages 1625–1630.

Barbara Ann Kipfer. 2009. *Rogets 21st Century Thesaurus*. Dell.

Roger Levy and Galen Andrew. 2006. Tregex and Tsurgeon: tools for querying and manipulating tree data structures. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, pages 2231–2234.

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*, pages 1501–1511.

Tomas Mikolov, Greg Corrado, Kai Chen, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations (ICLR) workshop*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 746–751.

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 *NAACL*, pages 142–148.

Masataka Ono, Makoto Miwa, and Yutaka Sasaki. 2015. Word embedding-based antonym detection using thesauri and distributional information. In *Proceedings of NAACL*, pages 984–989.

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 the Association for Computational Linguistics (ACL 2015)*, pages 425–430.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

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

Chaitanya Shivade, Marie-Catherine de Marneffe, Eric Fosler-Lussier, and Albert M. Lai. 2015. Corpus-based discovery of semantic intensity scales. In *Proceedings of NAACL*, pages 483–493.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks For Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26*, pages 926–934.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1509.

John Wieting, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Dan Roth. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the ACL (TACL)*, 3:345–358.