Impart Contextualization to Static Word Embeddings through Semantic Relations

Anonymous ACL submission

Abstract

Dense word embedding is the foundational model for the downstream NLP research. It encodes the meanings of words into low dimensional vector spaces. Recent models with the start-of-the-art performances mostly adopt the contextualized word embeddings, which can distinguish the various meanings of the words by their dynamic context. To impart the information of context to the static word embeddings, we formulate 3 semantic relations: interchangeable, opposite and relative relation to find a sub-set of dimensions for interpreting the specific context. The experiment shows that the relations can be mined from fastText embedding.

1 Introduction

Words are the smallest elements of a language to express a practical meaning. Recently, various unsupervised language models are developed to yield a dense representation to capture the "word meaning", which is commonly referred to as word embeddings. The training procedure is guided by the distributional hypothesis (Harris, 1954) : A word is known by the company it keeps. Existing unsupervised methods can be divided into two types: 1) **Static Word Embeddings (SWE)** map each word into a single vector, such as Word2Vec (Mikolov et al., 2013), Glove (Pennington et al., 2014) and fastText (Bojanowski et al., 2017); 2) **Contextualized Word Embeddings (CWE)** represent the words as vectors varying across extralinguistic contexts, such as ELMo (Peters et al., 2018), BERT(Devlin et al., 2019), XLNet (Yang et al., 2019), etc. At present, nearly all the state-of-the-art models on many NLP tasks use CWE as the word representation, which has yielded significant improvement.

The success of CWE suggests that the *context* is the key to understand the word meaning. However, how to access the meaning if the context is not accessible? For example, the meaning of "apple" can be determined in the context of "Apple is an edible fruit." or "Apple is a technology company." . But if a single word "apple" is given, its meaning is ambiguous.

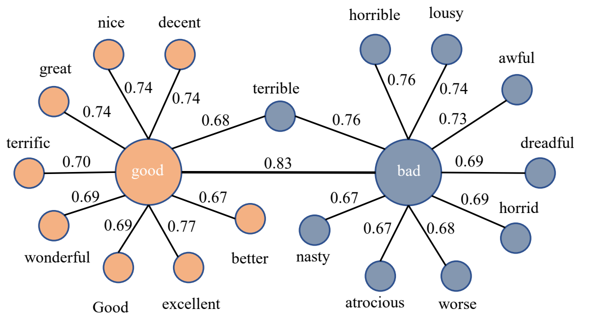


Figure 1: Top 10 most similar words of "good" and "bad" from fastText embeddings. The weight of the edge between two nodes is the cosine similarity.

In the embedding space, the cosine similarity is a common method for approximating how similar two word vectors are. As Fig. 1 shows, the top 10 similar words of "good" and "bad" from fastText embeddings can reflect the obvious sentiment orientation without any training of supervised tasks. However, the most similar word pair (good, bad) confuses the sentiment orientation. It is natural to ask whether we can separate them in the SWE?

Actually, the similarity is a rather general and vague concept. WordSim-353 (Finkelstein et al., 2001) collected 353 word pairs into 6 types of similar relations: synonym, antonym, hypernymy-hyponym, sibling (terms with a common hypernymy), part-of and topically related except the identical relation. The relations can't reflect the context similarity in the vector space of embeddings, so we re-organize them into 3 types of semantic relations:

1. **Interchangeable Relation.** Higher similarity score between two word embeddings essentially indicates that the words are interchangeable under more contexts. The context is their surrounding

words, which is determined by the windows size of language model. Obviously, the words of synonym and antonym are often interchangeable.

2. **Opposite Relation.** In some specific context, the antonym is not interchangeable, which will change the meaning of sentence.

3. **Relative Relation.** The hypernymy and part-of relation can be regarded as the relative relation between a concept and another set of words.

In this paper, we aim to impart the contextualization to static word embeddings. To explore the subset of dimensions, we formulate the above 3 relations in the vector space of dimension, and make the similar words as the context. The experiments show that the relations have insight into the dynamic context of static embeddings.

2 Methods

The whole word embedding matrix is $E \in \mathbb{R}^{n\times d}$, where n is the size of vocabulary, d is the number of dimensions. E_i and E_j represent the vector of ith word and jth dimension respectively .

The methods evaluate the relevance between each dimension and a set of similar words W according to the specific relation. The relevance score can be used to sort the dimensions to gain a dimension subset D ($|D| < d$), where higher similarity under D means the more similar context to W.

2.1 Interchangeable Relation

In the vector space of embedding, the higher similarity means more interchangeable of the words. As Fig. 2 (a) shows, if the words in W are interchangeable, the values of each dimension are required to be close. To make the dimension be sensitive to the given words, the values are also required to be far from the mean value of the dimension. Thus, the interchangeable score of jth dimension is given in Eq. (1).

$$S_I(j) = |\mu_{W^j} - \mu_{E^j}| - \sigma^2_{W^j} \tag{1}$$

The difference between W^j and E^j of the mean value reflects the degree of how the words are distinguished in the jth dimension of the vector space. The lower variance of W^j leads to higher S_O, which reflects the higher aggregation.

2.2 Opposite Relation

The opposite relation models the anatomy and the irrelevant words. In the vector space, the direction of the word vectors should be opposite (cosine similarity is -1) and orthogonal (similarity is 0) as Fig. 2 (b) shows.

Given two word sets with their embedding P and N, the objective of S_O is to increase the similarity within P and N, and decrease the similarity between P and N as Eq. (2), where g is a function to count the number over the mean value as Eq. (3).

$$S_O(j) = \frac{\sum_{i=1}^{|P|} g(P_i^j, j)}{|P|} + 1 - \frac{\sum_{i=1}^{|N|} g(N_i^j, j)}{|N|} \tag{2}$$

$$g(x, j) = \begin{cases} 1 & \text{for } x \geq \mu_{E^j}, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

2.3 Relative Relation

The success of word analogy test in Word2Vec (Mikolov et al., 2013) has shown the word embedding is able to capture the relative relation. In the vector space, the relative relation between a concept and a word set can be illustrated as Fig. 2 (c). It can be viewed as the interchangeable relation within the vector difference from the concept vector to other vectors.

Given the embeddings of a concept C and the relative words R, the relative score of each dimension is given in Eq. (4), where the objective of S_R is to find the representative dimensions in the relative vectors.

$$S_R(j) = S_I(C^j - \mu_{R^j}) \tag{4}$$

3 Experiments

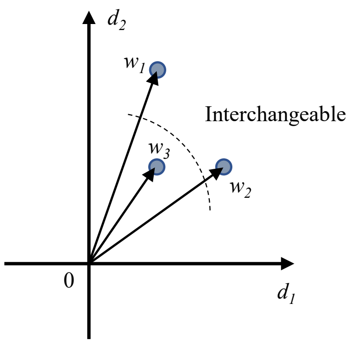
3.1 Dataset

We use the fastText [1] as the static word embedding, which is trained on wiki news and has 300 dimensions. The embeddings are transformed that has mean zero and standard deviation one. FastText is a convenient tool to fine-tune the embeddings on classification tasks (Joulin et al., 2017), which can be regarded as the supervised human concepts against the unsupervised language models.

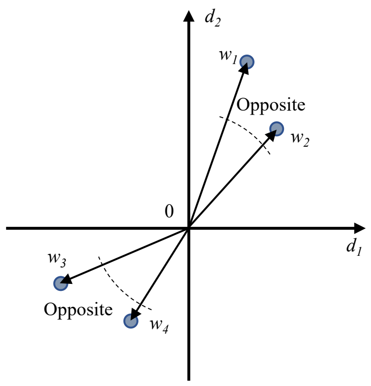
The involved word sets contain:

1. **Positive and Negative words.** We choose the top 100 similar words of "good" and "bad" respectively, and classify them into two types according to their average positive and negative score

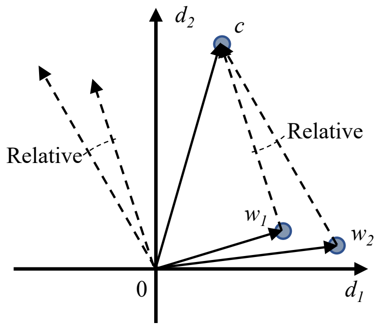
[1] https://dl.fbaipublicfiles.com/fasttext/vectors-english/wiki-news-300d-1M.vec.zip



(a) Interchangeable Relation

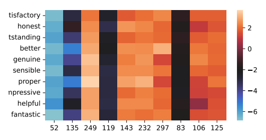


(b) Opposite Relation

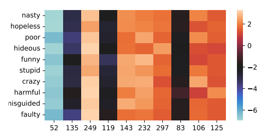


(c) Relative Relation

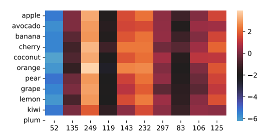
Figure 2: The vectors in the embedding space, where d_1 and d_2 represent dimensions, and w_n is words.



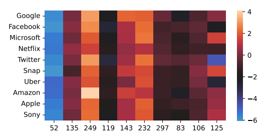
(a) positive



(b) negative



(c) fruits



(d) companies

Figure 3: The heat maps of top 10 dimensions on the 4 interchangeable word sets. We can observe that the similar and slightly different distributions between positive and negative can be captured.

provided by SentiWordNet (Taboada et al., 2011). Their embeddings are represented by P and N.

2. **Fruits, Vegetables and Big Technology Companies.** We collect tens of entities for each category [2]. Their embeddings are represented by F, V and T respectively.

The vocabulary is listed in Appendix A. The complete code with illustrations is available online.

3.2 Interchangeable Relation

We only use P as the interchangeable word sets, and sort dimension by S_I. The top 10 dimensions of P, N, F and T are illustrated in Fig. 3. It shows the similar distribution between P and N, and the difference of F and T are bigger.

3.3 Opposite Relation

The P and N can be deemed as the antonym in sentiment aspect to mine the opposite relation of embeddings. We use 4 models for comparison: 1) **fastText** use S_O scores directly; 2) **fine-tune** the fastText embeddings on IMDB review sentiment classification data set [3]; 3) **re-train** the fastText embeddings from random start on IMDB; 4) **svm** classifier uses the values of 300 dimensions as features to classify the positive and negative words, where its coefficient is the score of dimension. The quality of the top k dimensions D selected by the models is evaluated by the similarity between P^D and N^D as Fig. 4 shows.

The obvious result is that the re-train model makes all the 300 dimensions express the sentiment aspects, where the similarity is always less than -0.2. The curve of fine-tune represents a reduction of similarity compare with the original fastText embeddings. It shows that the sentimental concepts have been injected into the embeddings.

Compare with the curves of fastText and svm, we can find that S_O is effective to the sentimental dimensions, where the similarity of fastText comes to 0 in top 50 against top 10 of svm. But the similarity curve decreases when the k is above 200, it shows the room for improvement of S_O.

SentiWordNet also provides the sentimental polarity for each sense of words, we sum the polarities of all senses as the polarity of a word. A experiment is conducted to analyze the Pearson correlation between the P^K, N^K and their corresponding

[2] https://www.englishclub.com/vocabulary/food-vegetables.php

[3] http://ai.stanford.edu/~amaas/data/sentiment/

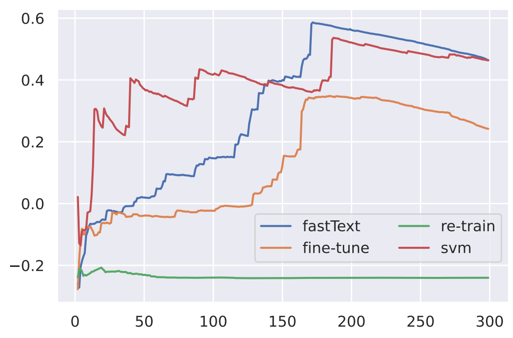


Figure 4: The similarity curves of the 4 models. The x axis means the top k dimensions are involved to calculate the cosine similarity. The y axis is the sum of similarity between positive words and negative words. The smaller values means the model is better.

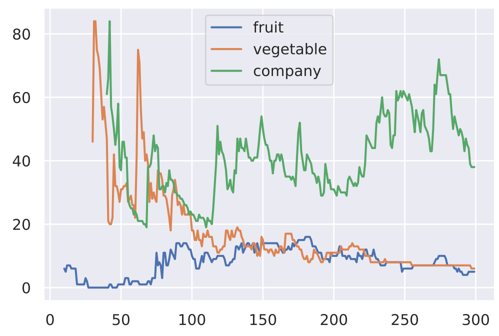


Figure 5: The y axis is recall index of the hypernymy word according to the similarity matrix between full word vectors and corresponding transformed vector of a test word. The vector is $f - \mu_{FK} + w$, where f is the vector of "fruit", w is test word "plum" for fruits, "zucchini" for vegetables, "Sony" for companies.

polarities in Appendix B, the result shows that the S_O score can't strongly reflect the polarity.

3.4 Relative Relation

We construct 3 hypernymy word sets, "fruit" as concept word and fruit words F, "vegetable" and vegetables V, "company" and big technology companies T. For each word set, we left a word to test the performance of S_R score. The curves given in Fig. 5 shows that the effect depends on the hypernymy word. The fruits performs better than the vegetables and companies. The "company" is a less representative hypernymy word to the company names.

3.5 Discussion

For the word embeddings that are trained on large data set, the values of all dimensions correspond to normal distribution. Therefore, the words that have the biggest or smallest value are more distinguished to emerge a certain semantics with less ambiguity. In fastText embeddings, we can find that the distinguished words are dominated by the technical terms, like "Fairchild", "synchrotron", "mammogram". The fields include computer science, biology, chemistry, economics, etc. The terms have less noisy context than the daily words "apple", "bad" with ambiguous sense. According to the observation, it maybe a more effective way to discriminate the interchangeable words.

For the relative relations, a complex situation is the hypernymy path in WordNet. It should be further studied to trace the concept "apple" from "edible fruit", "fruit , "plant", "object", and other intermediate hypernymy to finally "entity".

4 Related Work

To interpret the dense word embeddings, the non-negative matrix factorization is used to learn a sparse and interpretable representation (Murphy et al., 2012; Fyshe et al., 2014) on the co-occurrence matrices. Another route is using sparse coding techniques to project the dense vectors on a sparse higher dimensional vector spaces. The word intrusion test is introduced to evaluate the interpretability of the obtained vector space (Chang et al., 2009; Faruqui et al., 2015). Recently, the semantic structure within the categorized words, like the hypernym in HyperLex (Vulić et al., 2016), are used to interpret the dimensions. And a larger category data set is introduced (Şenel et al., 2018), where the experimental result shows the weak correlation between categories and single dimension. (Chersoni et al., 2021) decodes the brain-based semantic features to interpret the embeddings.

5 Conclusion

We summarizes 3 relations between words, and formulate them to find a sub set of dimensions. The methods rely on a set of similar words instead of contexts in the sentence, which can be viewed as contextualization for the static word embeddings. The experiments show the methods are effective but still have room for improvement. Further, we can extend the methods on the embeddings of subwords, and handle more complex situations.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan Boyd-Graber, and David M. Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems*.

Emmanuele Chersoni, Enrico Santus, Chu-Ren Huang, and Alessandro Lenci. 2021. Decoding Word Embeddings with Brain-Based Semantic Features. *Computational Linguistics*, 47(3):663–698.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. 2015. Sparse overcomplete word vector representations. In *International Joint Conference on Natural Language Processing*.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: the concept revisited. In *The Web Conference*.

Alona Fyshe, Partha Pratim Talukdar, Brian Murphy, and Tom M. Mitchell. 2014. Interpretable semantic vectors from a joint model of brain- and text- based meaning. In *Meeting of the Association for Computational Linguistics*.

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

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*.

Brian Murphy, Partha Pratim Talukdar, and Tom M. Mitchell. 2012. Learning effective and interpretable semantic models using non-negative sparse embedding. In *International Conference on Computational Linguistics*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Lütfi Kerem Şenel, Ihsan Utlu, Veysel Yücesoy, Aykut Koc, and Tolga Cukur. 2018. Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1769–1779.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307.

Ivan Vulić, Daniela Gerz, Douwe Kiela, Felix Hill, and Anna Korhonen. 2016. Hyperlex: A large-scale evaluation of graded lexical entailment. *arXiv: Computation and Language*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

A Vocabulary

Positive: brilliant, solid, interesting, satisfactory, admirable, well, neat, weird, clever, better, perfect, valuable, fantastic, great, excellent, respectable, impressive, suitable, happy, pleasing, easy, decent, proper, alright, honest, genuine, consistent, desirable, fabulous, beneficial, fine, enough, superb, okay, helpful, tasty, nice, straightforward, clear, essential, good, lovely, reasonable, outstanding, pleasant, acceptable, best, worthy, worthwhile, exemplary, marvelous, necessary, cool, amazing, big, wise, foolish, important, sensible, healthy

Negative: ghastly, appalling, strange, fun, unhealthy, careless, dirty, weak, rotten, wrong, hopeless, dreadful, depressing, evil, unhappy, troublesome, dangerous, harmful, silly, worst, bad, strong, plenty, unfortunate, awful, serious, sufficient, undesirable, little, sour, sloppy, disgusting, crazy, nasty, abominable, sad, atrocious, lousy, faulty, disappointing, hideous, miserable, misguided, tough, worse, plentiful, simple, wicked,

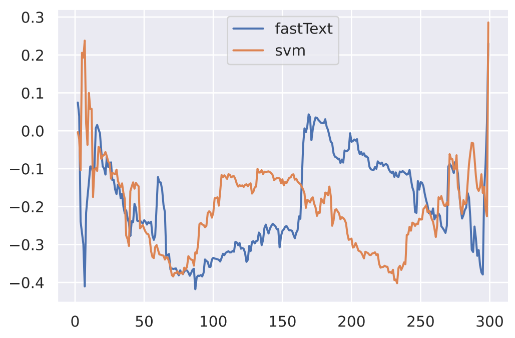


Figure 6: The Pearson correlation coefficient between the 2-norm of top k dimensions vectors and its sentiment polarity.

deplorable, ugly, horrid, stupid, unpleasant, indifferent, funny, mediocre, poor, terrific

Fruits: apple, avocado, banana, cherry, coconut, orange, pear, grape, lemon, kiwi, plum

Vegetables: artichoke, asparagus, beetroot, broccoli, cabbage, carrot, cauliflower, celery, cucumber, eggplant, garlic, lettuce, mushrooms, onion, peas, potato, pumpkin, spinach, turnip, yam, zucchini

Big Technology Companies: Google, Facebook, Microsoft, Netflix, Twitter, Snap, Uber, Amazon, Apple, Sony

B Sentiment Polarity

Fig. 6 shows the Pearson correlation coefficient between the 2-norm of top k dimensions and the polarity value, which is calculated as the square root of the inner product of vector with itself. Both the curves of S_O and svm are less than 0.5, which shows the weak relevance between the embeddings and polarity.