Towards Useful Word Embeddings

Evaluation on Information Retrieval, Text Classification, and Language Modeling

Vít Novotný, Michal Štefánik, Dávid Lupták, and Petr Sojka

Faculty of Informatics, Masaryk University
https://mir.fi.muni.cz/

Abstract. Since the seminal work of Mikolov et al. (2013), word vectors of log-bilinear models have found their way into many NLP applications and were extended with the positional model. Although the positional model improves accuracy on the intrinsic English word analogy task, prior work has neglected its evaluation on extrinsic end tasks, which correspond to real-world NLP applications. In this paper, we describe our first steps in evaluating positional weighting on the information retrieval, text classification, and language modeling extrinsic end tasks.

Keywords: Evaluation, word vectors, word2vec, fastText, information retrieval, text classification, language modeling

1 Introduction

In the beginning, there was a word. Word representations produced by log-bilinear models have found their way into many real-world NLP applications such as word similarity, word analogy, and language modeling [2] as well as dependency parsing [8, Section 5], word sense disambiguation [3], text classification [8], semantic text similarity [2], and information retrieval [19, Section 4].

The log-bilinear language of Mikolov et al. (2013) [10] and Bojanowski et al. (2018) [1] have been evaluated on both the intrinsic word analogy tasks and the extrinsic tasks of information retrieval, text classification, and language modeling. By contrast, the positional model of Mikolov et al. (2018) [11] has only been shown to reach SOTA performance on the intrinsic English word analogy task, but its usefulness to real-world NLP applications has been neglected. Our work describes the first steps in evaluating positional weighting on the information retrieval, text classification, and language modeling extrinsic end tasks.

Our work is structured as follows: In Section 2, we will describe the baseline model and the positional model. In Section 3, we will describe the initialization, the parameters, and the datasets for training the models. In Section 4, we will describe real-world applications of word vectors, and the datasets that we used in our experiments. In Section 5, we will show and discuss the results of our experiments. We conclude in Section 6 and suggest directions for future work.

A. Horák, P. Rychlý, A. Rambousek (eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2020, pp. 37–46, 2020.

2 Word embedding models

In this section, we will describe the baseline model and the positional model, which we used in our experiments.

2.1 Baseline model

Continuous bag of words In their seminal work, Mikolov et al. (2013) [10, Section 3.1] have introduced the continuous bag of words (CBOW) model, which is trained to predict a masked word in a context window P from the average $\mathbf{v}_C$ of the context word vectors $\mathbf{d}_p$:

$$\mathbf{v}_C = \frac{1}{|P|} \sum_{p \in P} \mathbf{d}_p. \tag{1}$$

In their later work, Mikolov et al. (2013) [12, Section 2.2] have introduced a faster alternative to the full/hierarchical softmax objectives by using a simplified variant of noise contrastive approximation (NCA) [6] called negative sampling.

Subword model The CBOW model only produces vectors for words that are present in the training corpus. Additionally, no weight sharing is used for different inflectional forms of a word, which slows down training convergence. To address both these points, Bojanowski et al. (2018) [2, Section 3.2] have extended CBOW by modeling subwords instead of words, making the vector $\mathbf{d}_p$ for a context word w_p an average of the vectors $\mathbf{d}_g$ for its subwords $g \in G_{w_p}$.

2.2 Positional model

In many sentences, the position of the context words is important for predicting the masked word. Consider the following two sentences, which produce an identical context vector $\mathbf{v}_C$ despite the different masked words:

1. Unlike dogs, cats are ⟨*mask*⟩.
2. Unlike cats, dogs are ⟨*mask*⟩.

If the context window P is large, distant context words will also be unimportant for predicting the masked word.

To better model these scenarios, Mikolov et al. (2018) [11, Section 2.2] adopted the positional weighting of Mnih and Kavukcuoglu (2013) [13, Section 3]. Positional weighting makes the average $\mathbf{v}_C$ into a weighted average $\mathbf{w}_C$, where the weight of a context word at a position p is the positional vector $\mathbf{u}_p$, and the weighting is carried out using the pointwise vector product $\odot$:

$$\mathbf{w}_C = \frac{1}{|P|} \sum_{p \in P} \mathbf{d}_p \odot \mathbf{u}_p. \tag{2}$$

Mikolov et al. reached SOTA performance on the intrinsic English word analogy task with the positional models. In our work, we will investigate whether positional weighting also improves the performance of the CBOW model on extrinsic end tasks with real-world NLP applications.

3 Experimental setup

In this section, we will describe the initialization, the parameters, and the datasets for training the models, which we used in our word vector experiments.

3.1 Initialization

For our baseline CBOW model, we follow the experimental setup of Bojanowski et al. (2017) [2] and initialize the subword vectors $\mathbf{d}_g$ from the continuous uniform distribution $\mathcal{U}(\pm\frac{1}{D})$, where D is the dimensionality of the subword vectors.

For the positional model, we initialize the subword vectors $\mathbf{d}_g$ to $\mathcal{U}(\pm\frac{1}{D})$ and the positional vectors $\mathbf{u}_p$ to $\mathbf{1}$. This corresponds to the *identity positional vectors* initialization described by Novotný in a separate article from these proceedings.

3.2 Model parameters

For our baseline CBOW model, we follow the experimental setup of Mikolov et al. (2018) [11, Section 4]: $|P| = 5, D = 300$, hash table bucket size $2 \cdot 10^6$, negative sampling loss with 10 negative samples, initial learning rate 0.05 with a linear decay to zero, sampling threshold 10^{-5}, subword sizes $\{3, 4, 5, 6\}$, minimum word count 5, and window size 5.

For the positional model, we follow the experimental setup of Mikolov et al. (2018) [11, Section 4] and increase the context window size to $|P| = 15$. These parameters are the SOTA on the English word analogy task.

3.3 Datasets

We trained both the baseline CBOW model and the positional model on the 2017 English Wikipedia[1] dataset over one epoch. The size of our dataset is only 4% of the Common Crawl dataset used by Mikolov et al. (2018) [11] to reach SOTA performance on the English word analogy task.

4 Applications

In this section, we will describe real-world applications of word vectors, and the datasets that we either used in our experiments or that we plan to use in future.

4.1 Information retrieval

Ad-hoc information retrieval is a standard retrieval task with applications in full-text search engines. In information retrieval, word vectors can be used as a source of word relatedness in semantic text distance or similarity measures such as the word mover's distance (WMD) [8] or the soft cosine measure (SCM) [17].

[1] https://github.com/RaRe-Technologies/gensim-data (release wiki-english-20171001)

We have not yet evaluated word vectors on information retrieval and this section will therefore discuss the Text Retrieval Conferences (TREC) datasets, which have been used for the evaluation of information retrieval systems at TREC conferences, and which we have preprocessed for future experiments.

Introduction The Natural Language Processing Centre at the Faculty of Informatics, Masaryk University (NLP Centre), obtained TREC disks 4 and 5 for research purposes free of charge from the National Institute of Standards and Technology (NIST) in 2019. Together with these test collections, test questions (topics) and relevance judgments are publicly available to evaluate information retrieval systems with the underlying TREC disks 4 and 5 documents. [16]

The Text Research Collection Volume 4 (Disk 4) from May 1996 and the Text Research Collection Volume 5 (Disk 5) from April 1997 were used together in the information retrieval tracks at TREC-6 through 8 in the years 1997–1999. [15] The collections include material from the Financial Times Limited (years 1991–1994), the Federal Register (year 1994), the Foreign Broadcast Information Service (year 1996), and the Los Angeles Times (years 1989 and 1990). [14] All collections consist of English texts, ranging from approximately 55k to 210k documents per collection and with a median of 316 to 588 words per document. [20] For each TREC conference, the organizers provided a set of 50 natural language topic statements [20,21,22] from which participants produced a set of queries that would be run against the collection documents.

The format of both documents and topics is the Standard Generalized Markup Language (SGML), with Document Type Definition (DTD) grammars provided for the documents. These validation capabilities ensure that we have superbly valid documents at our hands, but due to the overall complexity and lenient tagging of SGML, we lack a simple machine-readable format such as the Extensible Markup Language (XML) or the JavaScript Object Notation (JSON). In order to enhance machine-readability, we converted the documents to XML.

Preprocessing As listed above, Disks 4 and 5 consist of four different document collections. Hence, they contain four different DTD files because the dataset creators tended to keep the data as close to the original sources as possible. One of the DTD grammars contains the SGML declaration with parser instructions, which differs from the default Reference Concrete Syntax, mainly in increased limits to provide large document instances, like the quantities, capacities, or lengths of attribute values. None of these limits are restricted in XML. [4] Other changes include e.g. switching the SGML markup minimization features, which are also irrelevant for XML markup. Therefore, our main focus in the DTD files are the declarations of elements, attributes, and entities. These declarations are supported in both SGML DTDS and XML DTDS, which allowed us convert the SGML DTDS to XML DTDS with minimum effort just by following general rules. [9,23]

The inspection of the DTD grammars showed that the structure of the provided data in the SGML format is very close to the generally stricter syntax of XML. This is mainly the case of the syntax for omitting tags, which is present only for the top-level elements that enclose all documents in the collection. One possible

reason is that the data collection itself consists of multiple smaller files for easier manipulation. Otherwise, no markup minimization is present in the data.

To convert element type declarations, we excluded `OMITTAG` specifications since tag minimization is not allowed in XML. Another change was in mixed content models, i.e. elements that contain other elements or text (`#PCDATA`), that must have `#PCDATA` as the first element and must be optional (the `*` operator) in XML. For instance, the `TABLE` element type declaration in SGML DTDs

```
<!ELEMENT TABLE - - (FRFILING* | SIGNER* | #PCDATA)+ >
```

becomes the following in XML DTDs:

```
<!ELEMENT TABLE     (#PCDATA | FRFILING | SIGNER)* >
```

In SGML and XML documents, internal entities specify names, usually for special characters, that we can use in texts for markup, and the entity declaration contains the respective Unicode character reference to which the entity should expand. For most of the entity declarations in the SGML DTDs, we needed to supply numerical character references in decimal or hexadecimal format.

Another significant difference between the SGML and XML syntax is the requirement to put quotes around attribute values in XML, which, on the other hand, can be omitted in SGML. We satisfied this requirement with `osx`, a converter from SGML to XML from the OpenSP library, at the level of document collections.

The final preprocessing steps consisted of joining all chunks of documents into a single file, appending the updated external DTD file, and enclosing the whole data in the top-level element according to the given document collection and DTD grammar. We can validate the data with the `xmllint` XML parser using the following command.

```
$ xmllint --dtdvalid <DTD> <DATA> --noout
```

Conclusion We converted the TREC datasets for information retrieval purposes to a machine-readable format that we can use for our upcoming experiments. The datasets themselves are available[2] to all NLP Centre members who sign the individual agreement for research purposes. Even though the datasets are available under restrictive license conditions, we believe the described preprocessing steps will be useful for broader audiences outside the NLP Centre.

4.2 Text classification

Text classification is an NLP task with applications in automatic categorization and clustering of documents in interactive databases, e-shops, and digital libraries. In text classification, word vectors can be used as a source of word relatedness in semantic text distance or similarity measures such as the word mover's distance (WMD) [8] or the soft cosine measure (SCM) [17], respectively. To enable reproducibility, we publish our experimental code online.[3]

[2] https://nlp.fi.muni.cz/projekty/information-retrieval/
[3] http://drive.google.com/drive/folders/1TtGM0r4etmB8wU7_jtap3zokXd_O2mC2

Experimental setup In our evaluation, we used the WMD [8] and the orthogonalized SCM [17,18] semantic text distance and similarity measures with the k-nearest neighbors (kNN) classifier, following the setup of Kusner et al. (2015) [8]. Unlike Kusner et al., we did not optimize k and we hand-picked the value $k = 11$. For the SCM, we used the default parameters $o = 2, t = 0, C = 100$, Sym = `True`, and Dom = `False` from the implementation[4] of Novotný (2018) [17]. Whereas Kusner et al.'s WMD used the `nnn` BOW scheme for term weighting, our WMD uses the `bnn` binary BOW scheme and the SCM uses the `nfn` TF-IDF scheme.

Datasets As our datasets, we used BBCSPORT, TWITTER, RECIPE, OHSUMED, CLASSIC, REUTERS, and AMAZON from Kusner et al (2015) [8]. In TWITTER, the class labels indicate sentiment, whereas the other datasets are labeled by topic. Due to the high time complexity of the WMD, [18, Section 3] we only evaluated the WMD on the three hardest datasets in terms of the test error reported by Kusner et al. (TWITTER, RECIPE, and OHSUMED). Additionally, whereas Kusner et al. averaged their results on the TWITTER and RECIPE datasets across five train-test splits, we evaluate the WMD using only the first train-test split. To show that this should not significantly affect the accuracy of measurement, we report 95% confidence intervals for the test error of the SCM on the TWITTER and RECIPE datasets. For all datasets, measures, and models, we report test error. For the SCM, we also compute 95% significance on datasets with five splits (all except OHSUMED and REUTERS).

4.3 Language modeling

Language modeling is an NLP task with applications in predictive text, speech recognition, and optical character recognition. In language modeling, word vectors can be used to initialize the lookup table of a recurrent neural network. To enable reproducibility, we publish our experimental code online.[5]

Experimental setup In our evaluation, we trained a single-layer recurrent network similar to Bojanowski et al. (2017) [2, Section 5.6] with the following architecture:

1. an input layer with a map from a vocabulary V of $|V| = 2 \cdot 10^5$ most frequent words to frozen word vectors, followed by
2. an LSTM unit with a recurrent hidden output of size $D = 300$, followed by
3. a fully-connected linear layer of size $|V|$, followed by
4. a softmax output layer that computes probability over the vocabulary V.

We used negative log-likelihood loss, batch size 100, tied weights [7], dropout 0.2, SGD with learning rate 20, and gradient clipping to keep ℓ_2-norm below 0.25.

Datasets As our datasets, we used the data from the 2013 ACL Workshop on Machine Translation[6] with `news-train2011` over five epochs as the training set, `news-commentary-v8` as the validation set, and 10% of `news-train2012` as the test set. We report test perplexity and test loss, and we show learning curves for the training and validation perplexity.

[4] https://radimrehurek.com/gensim/similarities/termsim.html
[5] http://drive.google.com/drive/folders/1fVdNO8ZpfMtdJOWnD3Z1nu78UQPMcItG
[6] https://www.statmt.org/wmt13/translation-task.html

5 Results

In this section, we will show and discuss the results of our experiments on the text classification and language modeling extrinsic end tasks.

5.1 Text classification

Table 1 shows that the positional model outperforms the baseline model on the BBCSPORT, RECIPE, OHSUMED, CLASSIC, and AMAZON text classification datasets and only significantly underperforms the baseline model on the REUTERS text classification dataset. This shows that positional weighting produces word vectors that are better for text classification applications than the baseline.

Table 1 also shows that the positional model with the WMD significantly underperforms the baseline model with the WMD on the TWITTER sentiment analysis dataset. This is because word vectors capture relatedness and not sentiment. Antonyms such as *good* and *bad* appear in similar sentences, which makes their word vectors similar as well due to the training objective of CBOW. [18] Positional weighting makes it easier to satisfy the training objective, which naturally increases the test error on sentiment analysis datasets. This shows that word vectors are generally not well-suited to sentiment analysis applications.

In Table 1, the test error of the WMD with the baseline and positional models was computed using only the first of five train-test splits of the TWITTER and RECIPE datasets for speed. Since the 95% confidence intervals for the SCM are only $\pm$0.81% on TWITTER and $\pm$1.19% on RECIPE with the baseline model and $\pm$1.03% on TWITTER and $\pm$0.97% on RECIPE with the positional model, we conclude that the datasets are well-balanced and that the measurements are accurate.

Table 1. Classification error of the baseline and positional models with the WMD and SCM measures and the *k*NN classifier on the text classification test sets. For the WMD, we also list the results of Kusner et al. (2015) [8] for comparison. The best results are **emphasized**.

		BBCSPORT	TWITTER	RECIPE	OHSUMED	CLASSIC	REUTERS	AMAZON
WMD	Kusner [8]	**4.6%**	29%	43%	44%	**2.8%**	**3.5%**	**7.4%**
	Baseline		**23.78%**	43.47%	46.16%			
	Positional		38.20%	**34.23%**	46.32%			
SCM	Baseline	6.64%	29.03%	45.63%	41.32%	4.85%	7.58%	10.27%
	Positional	5.82%	28.54%	43.52%	**38.93%**	4.40%	8.73%	9.81%

5.2 Language modeling

Figure 1 shows that the positional model consistently improves the performance during the whole training compared to the baseline model. Figure 1 also shows that our language models have not plateaued and should train for more epochs.

Table 2 shows that the positional model outperforms the baseline on language modeling datasets. This shows that positional weighting produces word vectors that are better for language modeling applications than the baseline.

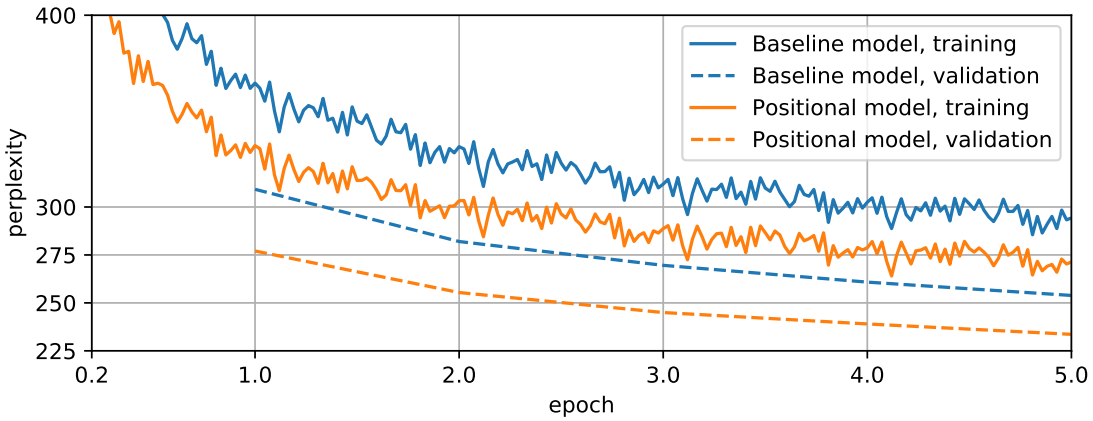

Fig. 1. Learning curves for the perplexity of the baseline and positional models on the language modeling training and validation sets.

Table 2. The perplexity and the loss of the baseline and positional models on the language modeling test set. The best results are **emphasized**.

	Test perplexity	Test loss
Baseline model	270.34	5.60
Positional model	**251.69**	**5.53**

6 Conclusion

We have shown that the positional model produces word vectors that are better-suited to text classification and language modeling NLP applications than the word vectors produced by baseline models. We have also described our steps in preprocessing the TREC information retrieval dataset for word vector evaluation.

In future work, we will evaluate both the positional model and other extensions of the baseline log-bilinear models on information retrieval, dependency parsing, word sense disambiguation, and other extrinsic end tasks using multilingual datasets. We will also train our word vector models using larger corpora such as Common Crawl to enable meaningful comparison to SOTA results.

Acknowledgments. First author's work was funded by the South Moravian Centre for International Mobility as a part of the Brno Ph.D. Talent project.

References

1. Bojanowski, P., et al.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
2. Charlet, D., Damnati, G.: Simbow at SemEval-2017 task 3: Soft-cosine semantic similarity between questions for community question answering. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). pp. 315–319 (2017)

3. Chen, X., Liu, Z., Sun, M.: A unified model for word sense representation and disambiguation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1025–1035 (2014)
4. Clark, J.: Comparison of SGML and XML. World Wide Web Consortium Note (1997)
5. Clark, K., Khandelwal, U., Levy, O., Manning, C.D.: What does BERT look at? An analysis of BERT's attention. arXiv preprint arXiv:1906.04341 (2019)
6. Gutmann, M.U., Hyvärinen, A.: Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. The Journal of Machine Learning Research, JLMR **13**(1), 307–361 (2012)
7. Inan, H., Khosravi, K., Socher, R.: Tying Word Vectors and Word Classifiers: A Loss Framework for Language Modeling. CoRR **abs/1611.01462** (2016), http://arxiv.org/abs/1611.01462
8. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From Word Embeddings To Document Distances. In: Bach, F., Blei, D. (eds.) International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 957–966. PMLR, Lille, France (07–09 Jul 2015), http://proceedings.mlr.press/v37/kusnerb15.html
9. McGrath, S.: XML by Example: Building E-Commerce Applications. Prentice Hall PTR (1999)
10. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
11. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in Pre-Training Distributed Word Representations. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (2018)
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed Representations of Words and Phrases and their Compositionality. In: Advances in Neural Information Processing Systems. pp. 3111–3119. Curran Associates, Inc. (2013), http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf
13. Mnih, A., Kavukcuoglu, K.: Learning word embeddings efficiently with noise-contrastive estimation. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems. vol. 26, pp. 2265–2273. Curran Associates, Inc. (2013), https://proceedings.neurips.cc/paper/2013/file/db2b4182156b2f1f817860ac9f409ad7-Paper.pdf
14. NIST: Data – English documents (2019), https://trec.nist.gov/data/docs_eng.html
15. NIST: Data – English documents introduction (2019), https://trec.nist.gov/data/intro_eng.html
16. NIST: Test Collections (2019), https://trec.nist.gov/data/test_coll.html
17. Novotný, V.: Implementation Notes for the Soft Cosine Measure. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. p. 1639–1642. CIKM '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3269206.3269317, https://doi.org/10.1145/3269206.3269317
18. Novotný, V., Ayetiran, E.F., Stefánik, M., Sojka, P.: Text classification with word embedding regularization and soft similarity measure. CoRR **abs/2003.05019** (2020), https://arxiv.org/abs/2003.05019
19. Novotný, V., Sojka, P., Štefánik, M., Lupták, D.: Three is better than one. In: CEUR Workshop Proceedings. Thessaloniki, Greece (2020), http://ceur-ws.org/Vol-2696/paper_235.pdf

20. Voorhees, E.M., Harman, D.: Overview of the sixth Text REtrieval Conference (TREC-6). Proceedings of the Sixth Text REtrieval Conference (TREC-6) pp. 1–24 (1998), https://trec.nist.gov/pubs/trec6/t6_proceedings.html, NIST Special Publication 500-240
21. Voorhees, E.M., Harman, D.: Overview of the seventh Text REtrieval Conference (TREC-7). Proceedings of the Seventh Text REtrieval Conference (TREC-7) pp. 1–23 (1999), https://trec.nist.gov/pubs/trec7/t7_proceedings.html, NIST Special Publication 500-242
22. Voorhees, E.M., Harman, D.: Overview of the eighth Text REtrieval Conference (TREC-8). Proceedings of the Eighth Text REtrieval Conference (TREC-8) pp. 1–23 (2000), https://trec.nist.gov/pubs/trec8/t8_proceedings.html, NIST Special Publication 500-246
23. Walsh, N.: Converting an SGML DTD to XML (1998), https://www.xml.com/pub/a/98/07/dtd/index.html