# Word vectors, reuse, and replicability:
# Towards a community repository of large-text resources

**Murhaf Fares, Andrey Kutuzov, Stephan Oepen, Erik Velldal**
Language Technology Group
Department of Informatics, University of Oslo
`{murhaff|andreku|oe|erikve}@ifi.uio.no`

## Abstract

This paper describes an emerging shared repository of large-text resources for creating word vectors, including pre-processed corpora and pre-trained vectors for a range of frameworks and configurations. This will facilitate reuse, rapid experimentation, and replicability of results.

## 1 Introduction

Word embeddings provide the starting point for much current work in NLP, not least because they often act as the input representations for neural network models. In addition to be being time-consuming to train, it can be difficult to compare results given the effects of different pre-processing choices and non-determinism in the training algorithms. This paper describes an initiative to create a shared repository of large-text resources for word vectors, including pre-processed corpora and pre-trained vector models for a range of frameworks and configurations.[1] This will facilitate rapid experimentation and replicability. The repository is available for public access at the following address:

> `http://vectors.nlpl.eu`

To demonstrate the impact of different pre-processing choices and parameterizations, we provide indicative empirical results for a first set of embeddings made available in the repository (Section 3). Using an interactive web application, users are also able to explore and compare different pre-trained models on-line (Section 4).

## 2 Motivation and background

Over the last few years, the field of NLP at large has seen a huge revival of interest for distributional semantic representations in the form of word vectors

---

[1] Our repository in several respects complements and updates the collection of Wikipedia-derived corpora and pre-trained word embeddings published by Al-Rfou et al. (2013).

(Baroni et al., 2014). In particular, the use of dense vectors embedded in low-dimensional spaces, so-called *word embeddings*, have proved popular. As recent studies have shown that beyond their traditional use for encoding word-to-word semantic similarity, word vectors can also encode other relational or 'analogical' similarities that can be retrieved by simple vector arithmetic, these models have found many new use cases. More importantly, however, the interest in word embeddings coincides in several ways with the revived interest in neural network architectures: Word embeddings are now standardly used for providing the input layer for neural models in NLP, where their low dimensionality is key. Some of the most popular frameworks for training embeddings are also themselves based on neural nets, like the neural language models underlying the *word2vec*-like algorithms (Mikolov, Sutskever, et al., 2013).

These models are sometimes referred to as 'prediction-based', and contrasted with traditional 'count-based' models based on representing co-occurrence counts, as the vector values of these embeddings are optimized for predicting neighboring words. In practice, this distinction is not clear-cut, and a more useful distinction can be made between explicit representations, where each dimension of a high-dimensional and sparse vector directly corresponds to a contextual feature, and embeddings in the sense of dimensionality-reduced continuous representations. Of course, research on vectorial representations of distributional semantics dates back several decades, including dimensionality reduced variants like *Latent Semantic Analysis* (LSA) based on Singular Value Decomposition (Landauer & Dumais, 1997), *Probabilistic LSA* based on a version of the EM algorithm (Hofmann, 1999), *Random Indexing* based on random projections (Kanerva et al., 2000; Karlgren & Sahlgren, 2001; Widdows & Ferraro, 2009; Velldal, 2011), *Locality Sensitive Hashing* (LSH) (Ravichandran

et al., 2005; Durme & Lall, 2010), and others.

It it is important to note that although the practical examples, experimental results and discussion in this paper will be concerned with embeddings generated with neural skipgram models and *Global Vectors* (GloVe) (Pennington et al., 2014), most of the issues will apply to word vectors more generally. The repository itself is also intended to host word vectors across all paradigms.

Deep learning and neural network architectures are now increasingly replacing other modeling frameworks for a range of core different NLP tasks, ranging from tagging (Plank et al., 2016) and parsing (Dyer et al., 2015; Straka et al., 2015) to named entity recognition (Lample et al., 2016) and sentiment analysis (Socher et al., 2013; Kim, 2014). Word embeddings typically provide the standard input representations to these models, replacing traditional feature engineering. Training is usually done on large amounts of unlabeled text, and all stages involved in the data preparation can potentially affect the resulting embeddings; content extraction from mark-up, sentence segmentation, tokenization, normalization, and so on. Just as important, the process for generating embeddings in with the aforementioned algorithms is *non-deterministic*: For the same set of hyperparameters and the same input data, different embeddings can (and most probably will) be produced. In sum these factors pose serious challenges for replicability for any work relying on word embeddings (Hellrich & Hahn, 2016).

The availability of pre-trained models is important in this respect. It can ensure that the same embeddings are re-used across studies, so that effects of other factors can be isolated and tested. Pre-trained models are currently available for a range of different algorithms: *Continuous Skipgram*, *Continuous Bag-of-Words*, *GloVe*, *fastText* (Bojanowski et al., 2016) and others. However, even when comparing the results for different pre-trained embeddings for a given data set, it can still be hard to know whether an observed difference is due to the embedding algorithm or text pre-processing.

Moreover, the available choices for pre-trained vectors are very limited in terms of data sets and pre-processing. Typically only embeddings trained on full-forms are available. However, given the convenience of using pre-trained vectors – training your own can often take several days and be computationally demanding – many studies use embeddings not ideally suited for the particular task.

For many semantically oriented tasks for example, embeddings trained on PoS-disambiguated lemmas would make more sense than using full-forms.

Given the considerations above, we find it important to establish a shared repository where it is possible to share training data, pre-processing pipelines, and pre-trained embeddings. Whenever possible, the training texts should be made available with various levels of pre-processing (e.g., lemmatized and PoS-tagged). In cases where licensing does not permit this, standardized pipelines for pre-processing should still be shared. In addition, a selection of sets of pre-trained word vectors should be made available for a few different parameterizations across different modeling frameworks, trained on data with varying degrees of pre-processing.

This will facilitate reuse and rapid experimentation. Should you still need to train you own vectors, you can use a standardized pipeline for pre-processing the training data. Most importantly, such a repository will help to ensure the replicability of results.

## 3 On the effects of corpus preparation

Levy et al. (2015) show that careful optimization of hyperparameters is often a more important factor for performance than the choice of embedding algorithm itself. The explicit specification of these hyperparameters is therefore essential to achieving a nuanced comparison between different word embedding approaches as well as replicability – inasmuch as replicating word embeddings is possible. As discussed in Section 2, the space of parameters associated with text pre-processing prior to training is also an important factor. To the best of our knowledge, however, there has been little research on the effect of corpus preparation on the training and performance of word embeddings.

In addition to the choice of training corpus itself, e.g. Wikipedia or Gigaword (Parker et al., 2011), there are many pre-processing steps involved in creating word embeddings. These steps include, but are not limited to, defining the basic token unit (full-form vs. lemma vs. PoS-disambiguated lemma), stop-word removal, downcasing, number normalization, phrase identification, named entities recognition and more. Other pre-processing steps depend on the nature of the training corpus; for example in training embeddings on text extracted from Wikipedia, the actual training corpus depends

on the content-extraction tools used to interpret Wiki markup. Moreover, in most cases the choice of the particular *tool* to use for steps like tokenization and sentence splitting will also make a difference. One of the important considerations we take in creating a shared repository of word embeddings is to spell out such choices.

**A pilot study**   To empirically demonstrate the impact of text pre-processing on word embeddings, we here present a pilot experiment, presenting intrinsic evaluation of a suite of embeddings trained for different choices of training corpora and pre-processing.

We trained twelve word embedding models on texts extracted from the English Wikipedia dump from September 2016 (about 2 billion word tokens) and Gigaword Fifth Edition (about 4.8 billion word tokens). We extracted the content from Wikipedia using *WikiExtractor*.[2] Further, we sentence-split, tokenized, and lemmatized the text in Wikipedia and Gigaword using *Stanford CoreNLP Toolkit 3.6.0* (Manning et al., 2014). We also removed all stop-words using the stop list defined in *NLTK* (Bird et al., 2009). In terms of pre-processing, the models differ in whether they were trained on full-forms or lemmas. Additionally, the models differ in the training text: Wikipedia (words with frequency less than 80 were ignored), Gigaword (frequency threshold 100) and Wikipedia concatenated with Gigaword (frequency threshold 200). All the corpora were shuffled prior to training.

The combination of the token choices and the training corpora leads to six different configurations. To eliminate the possibility of the effect of text pre-processing being approach-specific, we trained embeddings using both GloVe (Pennington et al., 2014) and Continuous Skipgram (Mikolov, Chen, et al., 2013) with negative sampling (SGNS). In terms of hyperparameters, we aligned GloVe and SGNS hyperparameters as much as possible: in both approaches we set the dimensionality to 300 and the symmetric context window size to 5. The SGNS models were trained using the Gensim implementation (Řehůřek & Sojka, 2010), using identical seed for all models; the GloVe models were trained with the reference implementation published by the authors.

We then evaluated the resulting models on two standard test datasets: SimLex-999 semantic sim-

ilarity dataset (Hill et al., 2015) and the Google Analogies Dataset (Mikolov, Chen, et al., 2013). The former contains human judgments on which word pairs are more or less semantically similar than the others (for example 'sea' and 'ocean' are more similar than 'bread' and 'cheese'). The task for the model here is to generate similarity values most closely correlating with those in the dataset. We follow the standard approach of evaluating performance towards SimLex-999 by computing Spearman rank-order correlation coefficients (Spearman, 1904), comparing the judgments on word pair similarities according to a given embedding model and the gold data.

The Google Analogies Dataset contains question pairs with proportional analogies: $a : a* :: b : b*$. For example, 'Oslo' is to 'Norway' as 'Stockholm' is to 'Sweden'. The task for the model is, given the vectors $(a, a*, b)$, to generate a vector, for which the closest vector in the model is $b*$. As a rule, the models solve this using the *3CosAdd* approach (Levy & Goldberg, 2014): $b* = a* + b - a$.

Results for the Google analogies test are standardly reported for two distinct sets of analogies in the data: 5 sections of 'semantic' relations (8,869 in total) and 9 sections of 'syntactic' relations (10,675 in total). The semantic relations are similar to the example with the capitals, while the syntactic part features analogies like 'walk' is to 'walks' as 'run' is to 'runs'. Measuring the effect of choices like using lemmas or full-forms only makes sense for the semantic tests, so we will not focus on the morphological and derivational analogies in our experiments.[3]

**Results and discussion**   Table 1 presents the results of evaluating our trained models on the benchmark datasets described above, showing how the results depend both on linguistic pre-processing and on the embeddings algorithm used. In Table 1, 'wiki', 'giga' and 'comb' denotes our 3 training corpora. The GloVe embeddings were trained with the default parameters except for the initial learning rate (0.02), number of iterations (100) and the window and vector size (cf. Section 3), 'SGNS' denotes Continuous Skipgram embeddings using

| Model | SimLex | Analogy |
|---|---|---|
| GloVe wiki lemmas | 36.13 | **83.08** |
| GloVe wiki forms | 31.27 | 81.80 |
| GloVe giga lemmas | 37.74 | 73.37 |
| GloVe giga forms | 32.36 | 72.20 |
| GloVe comb lemmas | 39.96 | 78.90 |
| GloVe comb forms | 34.81 | 77.46 |
| SGNS wiki lemmas | 40.19 | 78.86 |
| SGNS wiki forms | 35.54 | 77.60 |
| SGNS giga lemmas | 41.90 | 67.47 |
| SGNS giga forms | 37.96 | 66.84 |
| SGNS comb lemmas | **42.58** | 72.62 |
| SGNS comb forms | 38.21 | 72.54 |

Table 1: Results for SimLex-999 and the semantic sections of the Google Analogies Dataset.

10 negative samples.

Analysis of the evaluation results shows several important issues. First, while our SGNS models perform slightly better for the semantic similarity task, our GloVe models are more efficient in the semantic analogy test. The latter observation is perhaps not so surprising given that analogical inference was one of the primary aims of its authors.

Second, we see that more data is not necessarily better. For the benchmarking against SimLex-999, we do see that more data consistently leads to higher scores. For the semantic analogies task, however, the Gigaword corpus consistently results in models performing worse than Wikipedia, despite the fact that it is 2.5 times larger. Combining Gigaword and Wikipedia still yields lower scores than for Wikipedia alone. Moreover, with an accuracy of 83.08 for the semantic analogies, the GloVe model trained on the lemmatized version of Wikipedia outperforms the GloVe model trained on 42 billion tokens of web data from the Common Crawl reported in (Pennington et al., 2014), which at an accuracy of 81.9 was the best result previously reported for this task.

Finally, for both the semantic analogy task and the similarity task, we observe that the models trained on the lemmatized corpora are consistently better than the full-form models. In the future we plan to also evaluate our models on more balanced analogy datasets like that of Gladkova et al. (2016).

## 4 Infrastructure: Embeddings on-line

To achieve our goals of increased re-use and replicability, we are providing a public repository of texts, tools, and ready-to-use embeddings in the context of the Nordic e-Infrastructure Collaboration[4] and with support from national supercomputing centers. A comprehensive collection of resources for English and Norwegian[5] is available for download as well as for direct access by supercomputer users, combined with emerging documentation on the complete process of their creation, 'getting started' guides for end users, as well as links to indicative empirical results using these models. We invite feedback by academic peers on the repository already in this early stage of implementation and will welcome contributions by others.

In ongoing work, we are extracting even larger text corpora from web-crawled data and collaborating with other Nordic research centers (notably the University of Turku) to provide resources for additional languages. As the underlying supercomputing infrastructure is in principle open to all (non-commercial) researchers in the Nordic region, we hope that this repository will grow and develop into a community-maintained resource that greatly reduces the technological barrier to using very large-scale word vectors. The exact procedures for community contributions have yet to be determined, but we anticipate a very lightweight governing scheme. We intend to 'snapshot' versions of the repository at least once a year and publish these releases through the Norwegian Data Archive, to ensure long-term accessibility and citability.

The repository also provides the *WebVectors* web-service featuring pre-trained vectors for English and Norwegian.[6] Serving as an interactive explorer for the models, it allows users to retrieve nearest semantic associates, calculate similarities, apply algebraic operations to word vectors and perform analogical inference. It also features visualizations for semantic relations between words in the underlying models. This web service is thoroughly described by Kutuzov & Kuzmenko (2017).

---

## Acknowledgments

## References

Al-Rfou, R., Perozzi, B., & Skiena, S. (2013). Polyglot. Distributed word representations for multilingual NLP. In *Proceedings of the 17th Conference on Natural Language Learning* (p. 183–192). Sofia, Bulgaria.

Baroni, M., Dinu, G., & Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (p. 238–247). Baltimore, Maryland: Association for Computational Linguistics.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. Beijing: O'Reilly.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Durme, B. V., & Lall, A. (2010). Online generation of locality sensitive hash signatures. In *Proceedings of the 48th Meeting of the Association for Computational Linguistics* (p. 231–235). Uppsala, Sweden.

Dyer, C., Ballesteros, M., Ling, W., Matthews, A., & Smith, N. (2015). Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Meeting of the Association for Computational Linguistics and of the 7th International Joint Conference on Natural Language Processing.* Bejing, China.

Gladkova, A., Drozd, A., & Matsuoka, S. (2016). Analogy-based Detection of Morphological and Semantic Relations with Word Embeddings: What Works and What Doesn't. In *Proceedings of the NAACL Student Research Workshop* (p. 8–15). San Diego, California: Association for Computational Linguistics.

Hellrich, J., & Hahn, U. (2016). Bad Company—Neighborhoods in Neural Embedding Spaces Considered Harmful. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers* (p. 2785–2796). Osaka, Japan: The COLING 2016 Organizing Committee.

Hill, F., Reichart, R., & Korhonen, A. (2015). SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation. *Computational Linguistics*, *41*(4), 665–695.

Hofland, K. (2000). A self-expanding corpus based on newspapers on the Web. In *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC-2000).*

Hofmann, T. (1999). Probabilistic latent semantic analysis. In *Proceedings of the fifteenth conference on uncertainty in artificial intelligence* (p. 289–296). Stockholm, Sweden.

Kanerva, P., Kristoferson, J., & Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd annual conference of the cognitive science society* (p. 1036). PA, USA.

Karlgren, J., & Sahlgren, M. (2001). From words to understanding. In Y. Uesaka, P. Kanerva, & H. Asoh (Eds.), *Foundations of real-world intelligence* (p. 294–308). Stanford, CA, USA: CSLI Publications.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (p. 1746–1751). Doha, Qatar.

Kutuzov, A., & Kuzmenko, E. (2017). Building Web-Interfaces for Vector Semantic Models with the WebVectors Toolkit. In *Proceedings of the Demonstrations at the 15th Conference of the European Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Meeting of the North American Chapter of the Association for Computational Linguistics and Human Language Technology Conference* (p. 260–270). San Diego, CA, USA.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*(104), 211–240.

Levy, O., & Goldberg, Y. (2014). Linguistic Regularities in Sparse and Explicit Word Representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning* (p. 171–180). Ann Arbor, Michigan: Association for Computational Linguistics.

Levy, O., Goldberg, Y., & Dagan, I. (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association of Computational Linguistics*, *3*, 211–225.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations* (p. 55–60).

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space.

*arXiv preprint arXiv:1301.3781.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *Advances in Neural Information Processing Systems 26*, 3111–3119.

Parker, R., Graff, D., Kong, J., Chen, K., & Maeda, K. (2011). *English Gigaword Fifth Edition LDC2011T07* (Tech. Rep.). Technical Report. Linguistic Data Consortium, Philadelphia.

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (p. 1532–1543). Doha, Qatar: Association for Computational Linguistics.

Plank, B., Søgaard, A., & Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Meeting of the Association for Computational Linguistics.* Berlin, Germany.

Ravichandran, D., Pantel, P., & Hovy, E. (2005). Randomized algorithms and NLP: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Meeting of the Association for Computational Linguistics* (p. 622–629). Ann Arbor, MI, USA.

Řehůřek, R., & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (p. 1631–1642). Seattle, WA, USA.

Solberg, L. J. (2012). *A corpus builder for Wikipedia*. Unpublished master's thesis, University of Oslo, Norway.

Spearman, C. (1904). The Proof and Measurement of Association between Two Things. *The American Journal of Psychology*, *15*(1), 72–101.

Straka, M., Hajič, J., Straková, J., & Hajič jr., J. (2015). Parsing universal dependency treebanks using neural networks and search-based oracle. In *Proceedings of the 14th International Workshop on Treebanks and Linguistic Theories.* Warsaw, Poland.

Velldal, E. (2011). Random indexing re-hashed. In *Proceedings of the 18th Nordic Conference of Computational Linguistics* (p. 224–229). Riga, Latvia.

Widdows, D., & Ferraro, K. (2009). Semantic vectors: a scalable open source package and online technology management application. In *Proceedings of the 6th International Conference on Language Resources and Evaluation.* Marrakech, Morocco.