The Journal of Open Source Software

# embedplyr: Tools for Working With Text Embeddings

**Louis Teitelbaum** [1]

**1** Ben-Gurion University of the Negev, Israel

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:** Britta Westner

**Reviewers:**

- @cmaimone
- @orhunulusahin

**Submitted:** 15 January 2025
**Published:** unpublished



## Summary

Dense vector embeddings are the fundamental building block of modern natural language processing (Lauriola, Lavelli, & Aiolli, 2022). The ability to represent the meaning of texts as a continuous, high dimensional space underlies the recent successes of large language models (LLMs). Embeddings have also revolutionized research methods and inspired new theoretical frameworks in linguistics, psychology, sociology, and neuroscience (see e.g. Duran, Paxton, & Fusaroli, 2019; Feuerriegel et al., 2025; Grand, Blank, Pereira, & Fedorenko, 2022; Hamilton, Leskovec, & Jurafsky, 2018; Kjell, Giorgi, & Schwartz, 2023; Kozlowski, Taddy, & Evans, 2019; Schrimpf et al., 2021). As text embeddings become ubiquitous in the social and behavioral sciences, the need for flexible, easy-to-learn tools increases. Answering this need, **embedplyr** (pronounced "embe-DEE-plier") enables common operations with word and text embeddings within familiar analysis workflows.

## Statement of Need

**embedplyr** is designed for integration with a **tidyverse** (Wickham et al., 2019) and/or **quanteda** (Benoit et al., 2018) workflow, as demonstrated in Teitelbaum & Simchon (2024). Much as **dplyr** is "a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges" (Wickham, François, Henry, Müller, & Vaughan, 2025), **embedplyr** is a grammar of embeddings manipulation, designed to facilitate the use of word and text embeddings in common analysis workflows without introducing new syntax or unfamiliar data structures to R users.

Existing tools for working with embeddings in R are generally specific to particular model architectures. For example, **word2vec** (Wijffels, Watanabe, & Fomichev, 2023) provides access to word2vec models (Mikolov, Chen, Corrado, & Dean, 2013; Mikolov, Sutskever, Chen, Corrado, & Dean, 2013), **text2vec** (Selivanov, Bickel, & Wang, 2023) provides access to LDA, LSA, and GloVe models (Blei, Ng, & Jordan, 2003; Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990; Pennington, Socher, & Manning, 2014), **fastText** (Mouselimis, 2024) provides access to fastText models (Bojanowski, Grave, Joulin, & Mikolov, 2017a, 2017b; Facebook, 2016; Grave, Bojanowski, Gupta, Joulin, & Mikolov, 2018), and **text** (Kjell et al., 2023) provides access to transformers-based LLMs (Wolf et al., 2020). While these tools are already invaluable in the social and behavioral sciences, using different syntax and separate data structures for each model architecture can be cumbersome. This is especially true given the prevalence of studies that make use of multiple model architectures in parallel analyses (e.g. Carrella et al., 2023; Hussain, Mata, Newell, & Wulff, 2024; Markus, Levi, Sheafer, & Shenhav, 2024). **embedplyr** fills this gap with a model-agnostic approach; it can be used to work with embeddings from any model framework using syntax that will be familiar to any **tidyverse** user. While some existing tools focus on convenience functions for encouraging expert-recommended best practices (e.g. Kjell et al., 2023), **embedplyr** prioritizes flexibility—much like its namesake, **dplyr** (Wickham et al., 2025). This approach yields simple, modular functions that are useful both for educating students (see Teitelbaum & Simchon, 2024) and experimenting with novel methods.

# Features

## Loading Pretrained Token Embeddings

**embedplyr** does not include tools for training new embedding models, but it can load embeddings from a file or download them from online. This is especially useful for pretrained word embedding models like GloVe (Pennington et al., 2014), word2vec (Mikolov, Chen, et al., 2013; Mikolov, Sutskever, et al., 2013), HistWords (Hamilton et al., 2018), and fastText (Bojanowski et al., 2017b). Hundreds of these models can be conveniently downloaded from online sources with `load_embeddings()`, forgoing the need to search for model files online and juggle incompatible file types.

One particularly useful feature of `load_embeddings()` is the optional `words` parameter, which allows the user to specify a subset of words to load from the model. This allows users to work with large models, which are often too large to load into an interactive environment in their entirety.

```r
# load 25d GloVe model trained on Twitter
glove_twitter_25d <- load_embeddings("glove.twitter.27B.25d")

# load words from model trained on Google Books English Fiction 1800-1810
eng.fiction.all_sgns.1800 <- load_embeddings(
    "eng.fiction.all_sgns.1800",
    words = c("word", "token", "lemma")
    )
```

The output of `load_embeddings()` is an embeddings object. An embeddings object is simply a numeric matrix with fast hash table indexing by rownames (generally tokens). This means that it can be easily coerced to a dataframe or tibble, while also allowing special embeddings-specific methods and functions, such as `emb()` and `find_nearest()`:

```r
moral_embeddings <- emb(glove_twitter_25d, c("good", "bad"))
moral_embeddings
```

```
## # 25-dimensional embeddings with 2 rows
##       dim_1 dim_2 dim_3 dim_4 dim_5 dim_6 dim_7 dim_8 dim_9 dim..
## good -0.54  0.60 -0.15 -0.02 -0.14  0.60  2.19  0.21 -0.52 -0.23 ...
## bad   0.41  0.02  0.06 -0.01  0.27  0.71  1.64 -0.11 -0.26  0.11 ...
```

```r
find_nearest(glove_twitter_25d, "dog", 5L, method = "cosine")
```

```
## # 25-dimensional embeddings with 5 rows
##         dim_1 dim_2 dim_3 dim_4 dim_5 dim_6 dim_7 dim_8 dim_9 dim..
## dog    -1.24 -0.36  0.57  0.37  0.60 -0.19  1.27 -0.37  0.09  0.40 ...
## cat    -0.96 -0.61  0.67  0.35  0.41 -0.21  1.38  0.13  0.32  0.66 ...
## dogs   -0.63 -0.11  0.22  0.27  0.28  0.13  1.44 -1.18 -0.26  0.60 ...
## horse  -0.76 -0.63  0.43  0.04  0.25 -0.18  1.08 -0.94  0.30  0.07 ...
## monkey -0.96 -0.38  0.49  0.66  0.21 -0.09  1.28 -0.11  0.27  0.42 ...
```

Whereas indexing a regular matrix by rownames gets slower as the number of rows increases, **embedplyr**'s hash table indexing means that token embeddings can be retrieved in milliseconds even from models with millions of rows (see the performance vignette).

## Embed Texts of Interest

Given a tidy dataframe of texts, `embed_docs()` will generate embeddings by averaging the embeddings of tokens in each text (see Ethayarajh, Duvenaud, & Hirst, 2019; Teitelbaum & Simchon, 2024, chap. 18). By default, `embed_docs()` uses a simple unweighted mean, but many other averaging methods are available.

The following example embeds three texts, which for the sake of the example can be considered to have been written by one participant diagnosed with depression, one diagnosed with anxiety, and one control.

```
library(dplyr)

psych_df <- tribble(
    ~id,            ~text,
    "control",      "yesterday I took my dog for a walk",
    "depression",   "I slept all day and cried in the evening",
    "anxiety",      "I kept thinking of all the things I needed to do"
    )

# add embeddings to data frame
psych_embeddings_df <- psych_df |>
    embed_docs("text", glove_twitter_25d, id_col = "id", .keep_all = TRUE)
```

`embed_tokens()` is similar to `embed_docs()`, but returns the embedding of each individual token, rather than averaging within documents.

## Embed Dictionaries

Distributed Dictionary Representation (DDR) enables the application of validated psychometric lexicons (e.g. Boyd, Ashokkumar, Seraj, & Pennebaker, 2022) to rich, embedding-based semantic representation (Garten et al., 2018). This is achieved by retrieving pretrained word embeddings for each word in the dictionary, and averaging them to create a single vector—the DDR. The dictionary construct can then be measured by comparing text embeddings to the DDR.

In the following example, DDRs are constructed for high and low anxiety using example dictionaries. Once embeddings are produced for each word in the dictionary, they are averaged using `average_embedding()`. By default this is a simple mean, but `average_embedding()` also supports the geometric median (Cardot, 2022), weighted geometric median, and weighted mean, including weighting by word frequency or smooth inverse frequency (Arora, Liang, & Ma, 2017).

```
# positive and negative construct dictionaries
high_anx_dict <- c("anxious", "overwhelmed", "nervous", "stressed")
low_anx_dict <- c("relaxed", "calm", "mellow")

# embed dictionaries
high_anx_dict_embeddings <- emb(glove_twitter_25d, high_anx_dict)
low_anx_dict_embeddings <- emb(glove_twitter_25d, low_anx_dict)

# average embeddings to create DDR
high_anx_DDR <- average_embedding(high_anx_dict_embeddings)
low_anx_DDR <- average_embedding(low_anx_dict_embeddings)
```

`average_embedding()` could be used in a similar manner to construct contextualized construct representations (Atari, Omrani, & Dehghani, 2023).

## Calculate Similarity Metrics

To complete the DDR analysis initiated above, the embeddings of each the corpus texts are compared to that of the DDR. This could be done by computing cosine similarity between each text and `high_anx_DDR` (`"cosine"` is the default method for `get_sims()`). In this case however, an anchored vector is used to quantify the extent to which these texts reflect high anxiety *as opposed to low anxiety*. `method = "anchored"` gives the position of each embedding on the spectrum between two anchor points, where vectors aligned with `pos` are given a score of 1 and those aligned with `neg` are given a score of 0. This approach is also known as semantic projection (Grand et al., 2022).

```
anxiety_scores_df <- psych_embeddings_df |>
  get_sims(
    dim_1:dim_25,
    list(anxiety = list(pos = high_anx_DDR, neg = low_anx_DDR)),
    method = "anchored"
    )
anxiety_scores_df
```

```
## # A tibble: 3 x 3
##   id         text                                      anxiety
##   <chr>      <chr>                                       <dbl>
## 1 control    yesterday I took my dog for a walk          0.210
## 2 depression I slept all day and cried in the evening    0.338
## 3 anxiety    I kept thinking of all the things I needed to do   0.354
```

Note that `get_sims()` requires only a dataframe, tibble, or embeddings object with numeric columns; the embeddings can come from any source.

# Licensing and Availability

**embedplyr** is licensed under the GNU General Public License (v3.0). All of its source code is stored publicly on Github (https://github.com/rimonim/embedplyr), with a corresponding issue tracker.

# References


Arora, S., Liang, Y., & Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. *International conference on learning representations*.

Atari, M., Omrani, A., & Dehghani, M. (2023, February). Contextualized Construct Representation: Leveraging Psychometric Scales to Advance Theory-Driven Text Analysis. PsyArXiv. doi:10.31234/osf.io/m93pd

Benoit, K., Watanabe, K., Wang, H., Nulty, P., Obeng, A., Müller, S., & Matsuo, A. (2018). quanteda: An R package for the quantitative analysis of textual data. *Journal of Open Source Software*, *3*(30), 774. doi:10.21105/joss.00774

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, *3*, 993–1022. doi:10.7551/mitpress/1120.003.0082

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017a). Bag of Tricks for Efficient Text Classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers* (pp. 427–431). Association for Computational Linguistics. doi:10.48550/arXiv.1607.01759

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017b, June). Enriching Word Vectors with Subword Information. arXiv. doi:10.48550/arXiv.1607.04606

Boyd, R. L., Ashokkumar, A., Seraj, S., & Pennebaker, J. W. (2022). The development and psychometric properties of LIWC-22. *Austin, TX: University of Texas at Austin*, *10*.

Cardot, H. (2022). *Gmedian: Geometric Median, k-Medians Clustering and Robust Median PCA*. doi:10.32614/cran.package.gmedian

Carrella, F., Aroyehun, S. T., Lasser, J., Simchon, A., Garcia, D., & Lewandowsky, S. (2023, December). The 'Truth Contagion' Effect in the US Political Online Debate. OSF. doi:10.31234/osf.io/qx34w

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391–407. doi:10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

Duran, N. D., Paxton, A., & Fusaroli, R. (2019). ALIGN: Analyzing Linguistic Interactions With Generalizable techNiques-A Python Library. *Psychological methods*, *24*(4), 419–438. doi:10.31234/osf.io/a5yh9

Ethayarajh, K., Duvenaud, D., & Hirst, G. (2019, August). Towards Understanding Linear Word Analogies. arXiv. doi:10.48550/arXiv.1810.04882

Facebook, I. (2016). *fastText: Library for fast text representation and classification*. Retrieved from https://github.com/facebookresearch/fastText

Feuerriegel, S., Maarouf, A., Bär, D., Geissler, D., Schweisthal, J., Pröllochs, N., Robertson, C. E., et al. (2025). Using natural language processing to analyse text data in behavioural science. *Nature Reviews Psychology*, 1–16. doi:10.1038/s44159-024-00392-z

Garten, J., Hoover, J., Johnson, K. M., Boghrati, R., Iskiwitch, C., & Dehghani, M. (2018). Dictionaries and distributions: Combining expert knowledge and large scale textual data content analysis : Distributed dictionary representation. *Behavior Research Methods*, *50*(1), 344–361. doi:10.3758/s13428-017-0875-9

Grand, G., Blank, I. A., Pereira, F., & Fedorenko, E. (2022). Semantic projection recovers rich human knowledge of multiple object features from word embeddings. *Nature Human Behaviour*, *6*(7), 975–987. doi:10.1038/s41562-022-01316-8

Grave, E., Bojanowski, P., Gupta, P., Joulin, A., & Mikolov, T. (2018, March). Learning Word Vectors for 157 Languages. arXiv. doi:10.48550/arXiv.1802.06893

Hamilton, W. L., Leskovec, J., & Jurafsky, D. (2018, October). Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. arXiv. doi:10.48550/arXiv.1605.09096

Hussain, Z., Mata, R., Newell, B. R., & Wulff, D. U. (2024, December). Probing the contents of semantic representations from text, behavior, and brain data using the psychNorms metabase. arXiv. doi:10.48550/arXiv.2412.04936

Kjell, O. N. E., Giorgi, S., & Schwartz, H. A. (2023). The text-package: An R-package for Analyzing and Visualizing Human Language Using Natural Language Processing and Deep Learning. PsyArXiv. doi:10.31234/osf.io/293kt

Kozlowski, A. C., Taddy, M., & Evans, J. A. (2019). The Geometry of Culture: Analyzing the Meanings of Class through Word Embeddings. *American Sociological Review*, *84*(5), 905–949. doi:10.1177/0003122419877135

Lauriola, I., Lavelli, A., & Aiolli, F. (2022). An introduction to Deep Learning in Natural Language Processing: Models, techniques, and tools. *Neurocomputing*, *470*, 443–456. doi:10.1016/j.neucom.2021.05.103

Markus, D. K., Levi, E., Sheafer, T., & Shenhav, S. R. (2024, April). Reap the Wild Wind: Detecting Media Storms in Large-Scale News Corpora. arXiv. doi:10.48550/arXiv.2404.09299

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013, September). Efficient Estimation of

Word Representations in Vector Space. arXiv. doi:10.48550/arXiv.1301.3781

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013, October). Distributed Representations of Words and Phrases and their Compositionality. arXiv. doi:10.48550/arXiv.1310.4546

Mouselimis, L. (2024). *fastText: Efficient Learning of Word Representations and Sentence Classification using R*. Retrieved from https://CRAN.R-project.org/package=fastText

Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543). Doha, Qatar: Association for Computational Linguistics. doi:10.3115/v1/D14-1162

Schrimpf, M., Blank, I. A., Tuckute, G., Kauf, C., Hosseini, E. A., Kanwisher, N., Tenenbaum, J. B., et al. (2021). The neural architecture of language: Integrative modeling converges on predictive processing. *Proceedings of the National Academy of Sciences*, *118*(45), e2105646118. doi:10.1073/pnas.2105646118

Selivanov, D., Bickel, M., & Wang, Q. (2023, November). text2vec: Modern Text Mining Framework for R. doi:10.32614/cran.package.text2vec

Teitelbaum, L., & Simchon, A. (2024). *Data Science for Psychology: Natural Language*. Computational Social Psychology Lab. doi:10.5281/zenodo.10908367

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., et al. (2019). Welcome to the Tidyverse. *Journal of Open Source Software*, *4*(43), 1686. doi:10.21105/joss.01686

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2025). *Dplyr: A grammar of data manipulation*. Retrieved from https://dplyr.tidyverse.org

Wijffels, J., Watanabe, K., & Fomichev, M. (2023, October). word2vec: Distributed Representations of Words. doi:10.32614/cran.package.word2vec

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., et al. (2020). Transformers: State-of-the-Art Natural Language Processing. In Q. Liu & D. Schlangen (Eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38–45). Online: Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-demos.6