Journal of Machine Learning Research 26 (2025) 1-6 Submitted 10/22; Revised 7/25; Published 8/25

WEFE: A Python Library for Measuring and Mitigating Bias in Word Embeddings

Pablo Badilla PABLO.BADILLA@UG.UCHILE.CL
Felipe Bravo-Marquez FBRAVO@DCC.UCHILE.CL
María José Zambrano MZAMBRANO@UG.UCHILE.CL
Department of Computer Science, University of Chile, CENIA & IMFD

Jorge Pérez PEREZ@CERO.AI
CERO.AI

Editor: Pradeep Ravikumar

Abstract

Word embeddings, which are a mapping of words into continuous vectors, are widely used in modern Natural Language Processing (NLP) systems. However, they are prone to inherit stereotypical social biases from the corpus on which they are built. The research community has focused on two main tasks to address this problem: 1) how to measure these biases, and 2) how to mitigate them.

Word Embedding Fairness Evaluation (WEFE) is an open source library that implements many fairness metrics and mitigation methods in a unified framework. It also provides a standard interface for designing new ones.

The software follows the object-oriented paradigm with a strong focus on extensibility. Each of its methods is appropriately documented, verified and tested. WEFE is not limited to just a library: it also contains several replications of previous studies as well as tutorials that serve as educational material for newcomers to the field. It is licensed under BSD-3 and can be easily installed through `pip` and `conda` package managers.

Keywords: Fairness, Bias Measurement, Bias Mitigation, Word Embeddings, Natural Language Processing.

1. Introduction

Word embeddings are a set of techniques for representing words as continuous vectors in a dense space. These representations are learned from document corpora by exploiting the contexts in which the words occur (e.g., surrounding words within a fixed-size window). Consequently, and in accordance with the distributional hypothesis, words that appear in similar contexts (such as dog and cat) will tend to have similar representations. Embeddings are commonly built using the methods described in Word2vec (Mikolov et al., 2013), Glove (Pennington et al., 2014) or Fasttext (Bojanowski et al., 2017) among others. Needless to say, they have become an essential component of almost all NLP systems.

Several studies have shown that word embeddings can learn and exhibit stereotypical social biases from the corpus on which they are built (Caliskan et al., 2017; Garg et al., 2018; Bolukbasi et al., 2016).

An important line of research to address this problem is the development of metrics to quantify these biases. However, it is difficult to compare these metrics, as both their inputs (i.e., the sets of words on which they operate) and their outputs (i.e., the numerical score they return) are in most cases incompatible. In addition, each metric offers independent implementations with a different software interface. This makes it difficult to replicate and compare study results associated with each metric.

Another line of research is the development of debiasing methods that directly manipulate the embedding space to mitigate the underlying bias. These methods suffer from the same problems as measurement methods: each algorithm requires different inputs, there is no standardized process for running a debiasing method on a model, and there is no standard way to compare whether a method was effective or not.

To solve the above problems, we developed WEFE: the Word Embedding Fairness Evaluation framework[1]. WEFE is a Python library focused on standardizing and implementing bias measurement and mitigation methods for word embeddings. The main goal of the library is to provide a ready-to-use tool that allows the user to run bias measures and mitigation methods in a straightforward manner through well-designed and documented interfaces.

A preliminary version of this software was developed to run experiments in a previous publication (Badilla et al., 2020). The current version of WEFE has undergone several updates from that version, including the implementation of a debiasing module with five debiasing methods and the addition of three new metrics based on user contributions. The documentation has also been greatly expanded. For a complete overview of the changes implemented, please refer to the provided link[2].

2. The WEFE Library

WEFE implements a number of measurement and bias mitigation methods through its `metrics` and `debias` modules respectively. The library is based on several open source packages such as `numpy`, `scikit-learn`, `gensim`, `pandas` and `plotly`, and is available under the BSD 3 license.

The package is designed to be highly extensible through its inheritance-based class design. WEFE has an extensive documentation that is compiled and published online. The documentation includes several examples, replications of previous work and tutorials that explain to the community how to develop their own studies and contribute with new methods.

WEFE has a growing community, and that the time of writing, has been downloaded more than 42,000 times[3], and has received major external contributions, adding additional bias measurement metrics to those originally published, as well as several usage examples.

2.1 Measurements

WEFE specifies three building blocks to run a bias measurement on embedding models: *queries*, *embedding models*, and *metrics*. These blocks were originally proposed in (Badilla et al., 2020) and are encapsulated in classes as explained below:

[1] `https://wefe.readthedocs.io`

[2] `https://wefe.readthedocs.io/en/latest/benchmark/changes.html`

[3] `https://pepy.tech/project/wefe`

`WordEmbeddingModel` is a class in charge on containing a word embedding model. It is based on the `BaseKeyedVectors` `gensim` class which, in practical terms, provides compatibility with any model loaded with `gensim`.

The class `Query` contains the sets of words that are used as inputs for the bias metrics. A query is composed of two sets of words: target and attribute words.

Target words are a set of words that denote a certain social group according to a given criterion, which can be any character, trait or origin that distinguishes some groups of people from others, for example, gender, social class, age or ethnicity. Attribute words are a set of words that represent some attitude, characteristic, trait, or occupational field that can be associated with individuals from any social group. An example query focused on measuring gender bias might consist of two target words for the groups women and men, and two attribute words related to the sciences and the arts. The number of target sets and attribute sets required by a metric is called a *template* and is defined by the variables (t, a).

`Metric` classes implement specific bias metrics, each quantifying how strongly a target group is associated with certain attributes, based on its mathematical formulation. All metrics must inherit from `BaseMetric`, define a template specifying the number of target and attribute words, and implement the `run_query` method, which computes the bias from a query and a word embedding model. Importantly, the WEFE framework only supports metrics based on word sets (queries). Therefore, methods that require additional data, such as raw vector inputs or debiasing procedures, cannot be implemented straightforwardly within this framework.

In the current state, WEFE implements six metrics: Word Embedding Association Test (`WEAT`) (Caliskan et al., 2017), Relative Norm Distance (`RND`) (Garg et al., 2018), Relative Negative Sentiment Bias (`RNSB`) (Sweeney and Najafian, 2019), Mean Average Cosine (`MAC`) (Manzini et al., 2019), Embedding Coherence Test (`ECT`) (Dev and Phillips, 2019), and Relational Inner Product Association Test (`RIPA`) (Dev and Phillips, 2019).

2.2 Mitigation

WEFE standardizes all mitigation methods through an interface inherited from scikit-learn basic data transformations: the `fit-transform` interface. The first step `fit`, consists in learning the corresponding mitigation transformation which in some cases corresponds to a matrix projection of the embedding space. This method is quite flexible: it can accept multiple sets of words and other parameters.

The `transform` method applies the transformation learned in the previous step to words residing in the original embedding space. The method is rigid and only accepts lists of words that should be mitigated (target) or words that should be omitted (ignore).

Each class implementing a bias mitigation method must extend the `BaseDebias` class. Consequently, they must implement the corresponding fit and transform methods.

WEFE implements five bias mitigation techniques: Hard Debias (Bolukbasi et al., 2016), Multiclass Hard Debias (Manzini et al., 2019), Double Hard Debias (Wang et al., 2020), Repulsion Attraction Neutralization (Kumar et al., 2020b) and Half Sibling Regression (Yang and Feng, 2020).

3. Benchmark

To the best of our knowledge, there are only three other libraries besides WEFE that implement bias measurement and mitigation methods for word embeddings: Fair Embedding Engine (Kumar et al., 2020a), ResponsiblyAI (Hod, 2018) and Embedding Bias Score[4].

As part of our evaluation, we benchmarked WEFE against these libraries using key criteria such as installation ease, source code quality, documentation, model loading, ease of running bias measurements and bias mitigation. The comprehensive benchmark results can be accessed via the provided link[5], and a summary of the main findings is presented in Table 1.

	WEFE	FEE	Responsibly	EmbeddingBiasScores
Implemented metrics	7	5	3	6
Implemented mitigation Algorithms	5	3	1	0
Well-defined interface for metrics	✓	×	×	✓
Well-defined interface for mitigation algorithms	✓	×	×	×

Table 1: Comparative table

The key findings of our benchmark study, reveal that, in general, the aforementioned software tools present similar approaches for mitigating and measuring bias. However, WEFE outperforms the other three libraries by offering a wider selection of debiasing algorithms. Notably, WEFE distinguishes itself by providing a unifying perspective of the field through the implementation of standardization mechanisms and empowering users with the ability to customize the bias criteria. This standardization, however, limits WEFE to metrics based solely on query inputs, excluding those that rely on vector space operations or the application of external debiasing methods. In addition, also WEFE stands out for its comprehensive documentation, user examples, and ease of installation.

4. Conclusion and Future Work

We have presented WEFE, a tool for measuring and mitigating bias in word embeddings. The main contribution of the tool is a standardized view of the field that allows easy comparison of both bias measurements and mitigation techniques. It also provides ample material for reproducing existing work and conducting new experiments. We hope that WEFE will help raise awareness of the importance of considering bias and fairness in NLP and help design fairer systems in the future. For future work, we will extend the tool to operate with contextualized embeddings such as those proposed by Kurita et al. (2019) and Zhao et al. (2019).

Acknowledgments

This work was supported by ANID FONDEF grant ID25I10330, the National Center for Artificial Intelligence CENIA FB210017, and ANID -Millennium Science Initiative Program - Code ICN17_002.

[4] `https://github.com/FEE-Fair-Embedding-Engine/FEE`

[5] `https://wefe.readthedocs.io/en/latest/benchmark/benchmark.html`

References

Pablo Badilla, Felipe Bravo-Marquez, and Jorge Pérez. Wefe: The word embeddings fairness evaluation framework. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 430–436. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/60. URL `https://doi.org/10.24963/ijcai.2020/60`. Main track.

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

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. *Advances in neural information processing systems*, 29, 2016.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186, 2017.

Sunipa Dev and Jeff Phillips. Attenuating bias in word vectors. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 879–887. PMLR, 2019.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644, 2018.

Shlomi Hod. Responsibly: Toolkit for auditing and mitigating bias and fairness of machine learning systems, 2018. URL `http://docs.responsibly.ai/`. [Online; accessed 26.05.2022].

Vaibhav Kumar, Tenzin Bhotia, and Vaibhav Kumar. Fair embedding engine: A library for analyzing and mitigating gender bias in word embeddings. In *Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS)*, pages 26–31, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlposs-1.5. URL `https://aclanthology.org/2020.nlposs-1.5`.

Vaibhav Kumar, Tenzin Singhay Bhotia, Vaibhav Kumar, and Tanmoy Chakraborty. Nurse is closer to woman than surgeon? mitigating gender-biased proximities in word embeddings. *Transactions of the Association for Computational Linguistics*, 8:486–503, 2020b.

Keita Kurita, Nidhi Vyas, Ayush Pareek, Alan W Black, and Yulia Tsvetkov. Measuring bias in contextualized word representations. In *Proceedings of the First Workshop on Gender Bias in Natural Language Processing*, pages 166–172, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-3823. URL `https://aclanthology.org/W19-3823`.

Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. Black is to criminal as caucasian is to police: Detecting and removing multiclass bias in word embeddings. 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 615–621, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1062. URL `https://www.aclweb.org/anthology/N19-1062`.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Chris Sweeney and Maryam Najafian. A transparent framework for evaluating unintended demographic bias in word embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1662–1667, 2019.

Tianlu Wang, Xi Victoria Lin, Nazneen Fatema Rajani, Bryan McCann, Vicente Ordonez, and Caiming Xiong. Double-hard debias: Tailoring word embeddings for gender bias mitigation. In *Association for Computational Linguistics (ACL)*, July 2020.

Zekun Yang and Juan Feng. A causal inference method for reducing gender bias in word embedding relations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34 (05):9434–9441, 2020.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. Gender bias in contextualized word embeddings. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *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 629–634, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1064. URL `https://aclanthology.org/N19-1064/`.