The Journal of Open Source Software

SINr: a python package to train interpretable word and graph embeddings

Thibault Prouteau [1¶], **Nicolas Dugué** [1¶], **Simon Guillot**[1], **and Anthony Perez**[2]

1 Le Mans Université, LIUM, EA 4023, Laboratoire d'Informatique de l'Université du Mans, France 2 Univ. Orléans, INSA Centre Val de Loire, LIFO EA 4022, Orléans, France ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

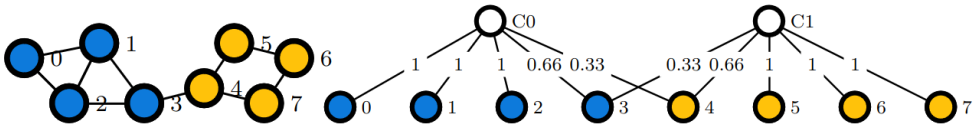
Summary

In this paper, we introduce SINr, a Python package to train word and graph embeddings. Word embeddings allows to encompass the meaning of words into vectorial representations that can be used by algorithms. The principle was then transfered to the study of graphs, mathematical representations that represent entities of the real words and their relations (social networks for instance). Those vectors encompassing information are often the first building block of more complex artificial intelligence pipelines. With SINr, we focus on providing frugal and interpretable algorithms to build this first block. Indeed, because of the growing complexity of the algorithms used in artificial intelligence (deep learning), frugality and interpretability became major concerns. The SINr approach is based on community detection: a vector for a node is built upon the distribution of its connections through the communities detected on the graph at hand. The algorithm thus runs very efficiently, and does not require GPUs to proceed allowing a broad usage. Furthermore, the dimensions of the embedding space are interpretable, they are based on the communities extracted. One can thus interact with the dimensions of the vectors and inspect the meaning of the representation.

Statement of need and functionalities

The SINr package can be used in the field of natural language processing (NLP), digital humanities, or networks analysis. It allows to represent words and networks entities as vectors in mathematical spaces named embedding spaces.

With neural approaches, tremendous progress was made in NLP, notably to represent the vocabulary of the language at hand. Those representations are then used as input for machine learning algorithms in the form of dense numeric vectors named word embeddings. Some examples of approaches to train such vectors using large textual corpora are Word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), and the Transformer-based approaches for contextualized representations, CamemBERT (Martin et al., 2019) or FlauBERT (Le et al., 2019) in French. This progress was transfered to the graph universe, allowing the emergence of graph embedding, a whole field of research with Word2vec inspired approaches such as Node2vec (Grover & Leskovec, 2016), matrix factorization methods like HOPE (Ou et al., 2016) and auto-encoding paradigms (Cao et al., 2016).



(a) A graph $G = (V, E)$ partitioned in two communities.

(b) Bipartite projection of G into graph $G\prime = (\top, \bot, E)$ along the communities. Weight on the edges is the proportion of neighbors in that community.

	C1	*C2*
0	1	0
1	1	0
2	1	0
3	0.66	0.33
4	0.33	0.66
5	0	1
6	0	1
7	0	1

(c) Adjacency matrix of $G\prime$, each row is a `SINr` embedding.

Figure 1: Illustration of SINr, vertices are represented based on the communities they are linked to.

SINr was introduced to take advantage of this progress, it allows to embed words and nodes just like the aforementioned methods. However, it is based on community detection: for each node, the embedding vector is calculated as the proportion of its links going to the communities as described Figure 1. This approach allows to avoid some flaws inherent to the usual approaches:

- As far as we know, SINr is the first approach specifically designed to deal with both word and graph embeddings. Text corpora are represented as graphs, and with the adequate preprocessing provided by the package, word embeddings can easily be extracted with SINr. For graph embedding, no specific pre-processing is required.
- Contrary to the neural approaches that require complex GPU calculations, SINr is based on the Louvain (Blondel et al., 2008) algorithm to detect community and thus runs in linear-time, it can be executed on standalone laptops. Still, it can easily be interfaced with the gensim package for word embedding, or with the karateclub package for graph embedding.
- Contrary to the usual approaches, because dimensions are based on the communities, the space in which words and graphs are embedded with SINr is interpretable.

Design patterns were used to allow the usage of SINr with graphs or text in the same manner, which makes it -as far as we know- the first package to deal by design with both of these data types. For usage in the context of digital humanities or NLP, standard preprocessing algorithms are also provided, mostly relying on spaCy. Finally, the most important aspect of SINr is that it allows interaction with the embedding space. SINr brings the option to probe and get an understanding of the resulting embedding space as one can see Figure 2. Indeed, using SINr leads to sparse vectors: a node is not connected to all the communities of a graph, and similarly, a word is not related to all the topics of a corpus. As shown by Subramanian et al. (2018), sparsity is one of the features required to enforce interpretability.

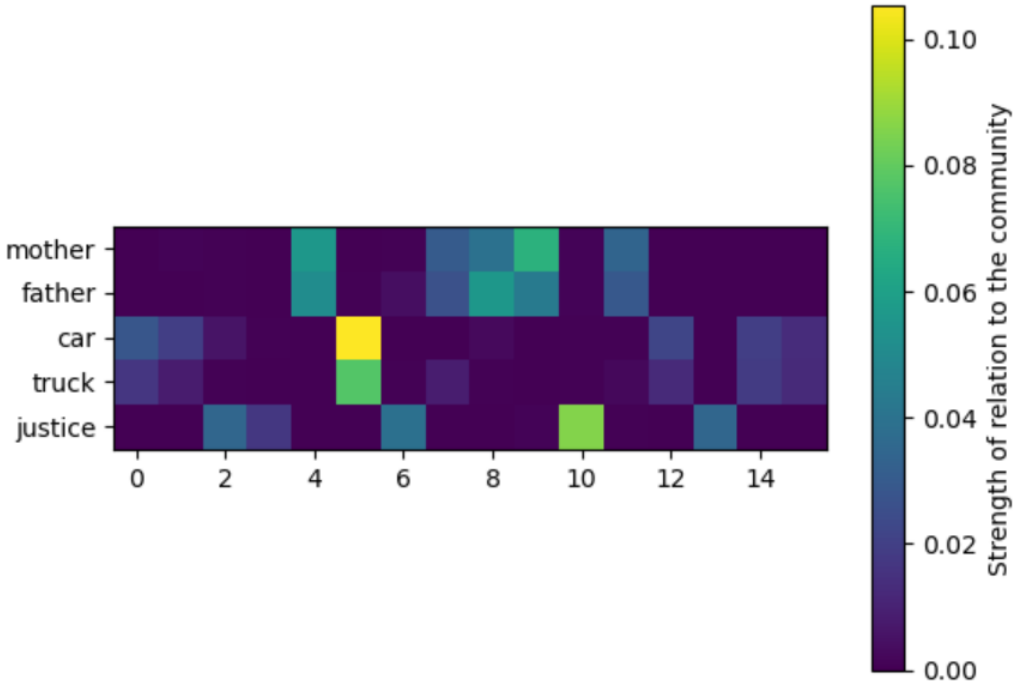


Figure 2: Using the visualization features of the package, one can see that related words have non-zero values for the same dimensions (abscissa), mother and father for dimensions 4, 7, 8, 9 and 11 for instance. The non-zero dimensions are distinct when comparing mother and car that are unrelated.

Performances of the SINr approach

The performances of SINr were evaluated on several tasks, including link prediction on graphs, and pair of words similarities for textual data in Prouteau et al. (2021). While providing good performances, it runs faster than most of the other embedding approaches. Furthermore, the interpretability of the model was also demonstrated to be comparable to the state-of-the-art when considering word embedding (Prouteau et al., 2022).

Availability

The SINr package is distributed on https://github.com/SINr-Embeddings/sinr with its documentation and notebooks. It can also be found on Pypi to be installed with pip.

Scope and Alternatives

For graph embedding, the Karateclub library designed by Rozemberczki et al. (2020) allows to try and test recent approaches. However, so far, none of them is known to be as fast as SINr or interpretable. For statistical semantics on textual corpora, the gensim package (Řehřek et al., 2011) is still a good option. For more recent approaches, the Flair (Akbik et al., 2019) and spaCy (Vasiliev, 2020) packages can be used, allowing to build complete NLP pipelines. SINr embeddings can actually be used complementary to these packages.

Acknowledgements

This research benefited from the support of the ANR project 'Dynamic and Interpretable Graph-based word embeddINGs' (ANR-21-CE23-0010).

References

Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., & Vollgraf, R. (2019). FLAIR: An easy-to-use framework for state-of-the-art NLP. *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 54–59.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008.

Cao, S., Lu, W., & Xu, Q. (2016). Deep neural networks for learning graph representations. *Proceedings of the AAAI Conference on Artificial Intelligence*, *30*.

Grover, A., & Leskovec, J. (2016). node2vec: Scalable feature learning for networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 855–864.

Le, H., Vial, L., Frej, J., Segonne, V., Coavoux, M., Lecouteux, B., Allauzen, A., Crabbé, B., Besacier, L., & Schwab, D. (2019). Flaubert: Unsupervised language model pre-training for french. *arXiv Preprint arXiv:1912.05372*.

Martin, L., Muller, B., Suárez, P. J. O., Dupont, Y., Romary, L., La Clergerie, É. V. de, Seddah, D., & Sagot, B. (2019). CamemBERT: A tasty french language model. *arXiv Preprint arXiv:1911.03894*.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv Preprint arXiv:1301.3781*.

Ou, M., Cui, P., Pei, J., Zhang, Z., & Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1105–1114.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*, 1532–1543.

Prouteau, T., Connes, V., Dugué, N., Perez, A., Lamirel, J.-C., Camelin, N., & Meignier, S. (2021). SINr: Fast computing of sparse interpretable node representations is not a sin! *IDA*, 325–337.

Prouteau, T., Dugué, N., Camelin, N., & Meignier, S. (2022). Are embedding spaces interpretable? Results of an intrusion detection evaluation on a large french corpus. *LREC*.

Řehřek, R., Sojka, P., & others. (2011). Gensim—statistical semantics in python. *Retrieved from Genism. Org*.

Rozemberczki, B., Kiss, O., & Sarkar, R. (2020). Karate club: An API oriented open-source python framework for unsupervised learning on graphs. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 3125–3132.

Subramanian, A., Pruthi, D., Jhamtani, H., Berg-Kirkpatrick, T., & Hovy, E. (2018). Spine: Sparse interpretable neural embeddings. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*.

Vasiliev, Y. (2020). *Natural language processing with python and spaCy: A practical introduction*. No Starch Press.