The Journal of Open Source Software

textnets: A Python package for text analysis with networks

John D. Boy[1]

1 Assistant Professor, Faculty of Social Sciences, Leiden University

DOI: 10.21105/joss.02594

Software

- Review
- Repository
- Archive

Editor: George K. Thiruvathukal

Reviewers:

- @sara-02
- @tresoldi

Submitted: 16 June 2020
Published: 15 October 2020

Background

Social scientists increasingly rely on computational tools to make sense of vasts amounts of unstructured data generated in the wake of the ever-expanding digitization of social life. Electronic text, in particular, is a growing area of interest thanks to the social and cultural insights lurking in social media posts, digitized corpora, and web content, among other troves (Evans & Aceves, 2016; Ignatow, 2015).

This package aims to fill that need. `textnets` represents collections of texts as networks of documents and words, which provides powerful possibilities for the visualization and analysis of texts.

The package can operate on the bipartite network containing both document and word nodes. Figure 1 shows an example of a visualization created by `textnets`. The underlying corpus is a collection of statements by U.S. Senators following the conclusion of the impeachment trial against the president in February 2020. Documents appear as triangles (representing the Senators who issued the statements), and words appear as yellow squares.

`textnets` can also project one-mode networks containing only document or word nodes, and it contains tools to analyze them. For instance, it can visualize a backbone graph with nodes scaled by various centrality measures. For networks with a clear community structure, it can also output lists of nodes grouped by cluster as identified by a community detection algorithm. This can help identify latent themes in corpus texts (Gerlach, Peixoto, & Altmann, 2018).

Another implementation of the textnets technique exists in the `R` programming language by its originator (Bail, 2016); it can be found at https://github.com/cbail/textnets. Feature-wise, the two implementations are roughly on par. This implementation in Python features a modular design, which is meant to improve ergonomics for users and potential contributors alike. This package aims to make text analysis techniques accessible to a broader range of researchers and students. Particularly for use in the classroom, `textnets` aims at seamless integration with the Jupyter ecosystem (Kluyver et al., 2016).

`textnets` is well documented: its API reference, contribution guidelines, and a comprehensive tutorial can be found at https://textnets.readthedocs.io. For easy installation, the package is included in conda-forge and the Python Package Index. Its code repository and issue tracker are currently hosted on GitHub at https://github.com/jboynyc/textnets. A test suite is run using Travis, a continuous integration service, before new releases are published to avoid regressions from one version to another. Archived versions of releases are available at doi:10.5281/zenodo.3866676.

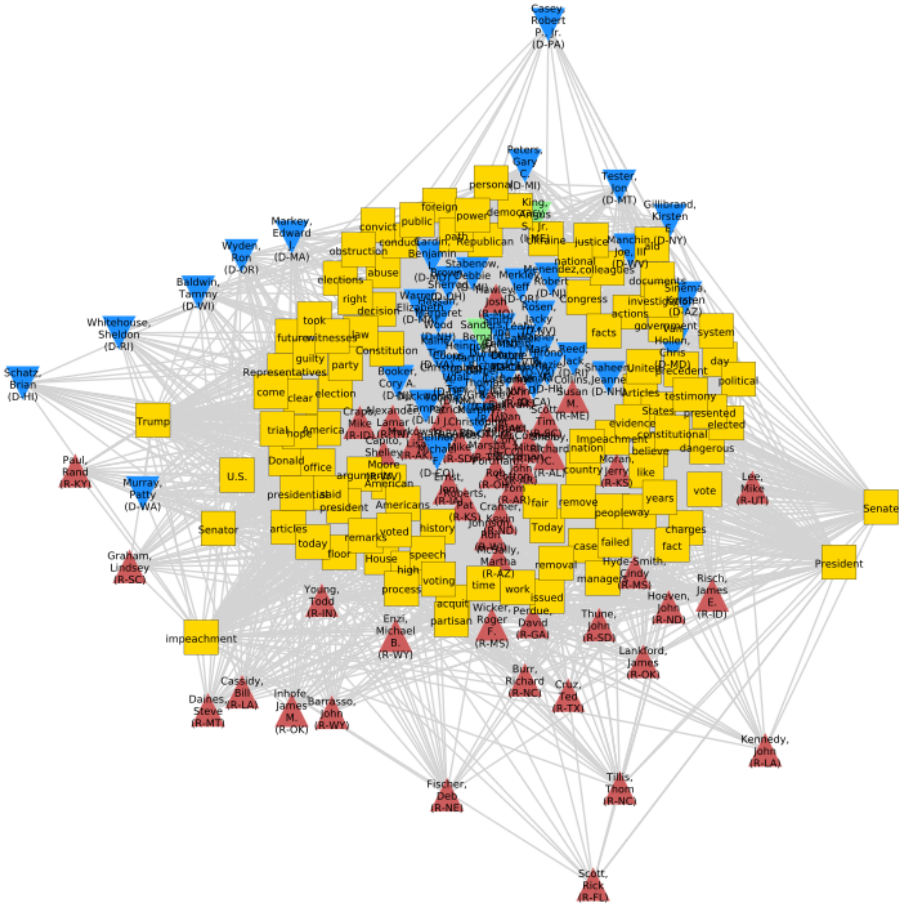

Figure 1: Network of U.S. Senators and words used in their official statements following the acquittal vote in the Senate impeachment trial in February 2020 (source: Boy, 2020).

Statement of Need

With `textnets` it is possible to visualize and analyze textual data in novel ways. These are some of the package's distinguishing features:

- Existing text analysis packages, such as Benoit et al. (2018), typically visualize texts as word clouds, not as network graphs. Unlike word clouds, network graphs can visualize not just the frequency and co-occurrence of text features, but also their linking role within corpora.
- The discovery of topics is typically performed using latent Dirichlet allocation (LDA), while `textnets` uses community detection on the term graph for that purpose. Unlike topic modeling using LDA, this does not require specifying a fixed number of topics.
- `textnets` can also cluster documents using community detection on the document graph. This can serve as an alternative to techniques like k-means clustering.

Dependencies

For most heavy lifting, `textnets` uses data structures and methods from `igraph` (Csárdi & Nepusz, 2006), `numpy`, and `pandas` (McKinney, 2013). It leverages `spacy` for natural language processing (Honnibal & Montani, 2017). For community detection, it relies on the Leiden algorithm in its implementation by Traag (2020). It also depends on `scipy` (Virtanen et al., 2020) to implement the backbone extraction algorithm.

References

Bail, C. A. (2016). Combining natural language processing and network analysis to examine how advocacy organizations stimulate conversation on social media. *Proceedings of the National Academy of Sciences*, *113*(42), 11823–11828. doi:10.1073/pnas.1607151113

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

Boy, J. D. (2020, June 11). Enemies, foreign and partisan. Retrieved from https://www.jboy.space/blog/enemies-foreign-and-partisan.html

Csárdi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal: Complex Systems*, *1695*, 2006.

Evans, J. A., & Aceves, P. (2016). Machine translation: Mining text for social theory. *Annual Review of Sociology*, *42*(1), 21–50. doi:10.1146/annurev-soc-081715-074206

Gerlach, M., Peixoto, T. P., & Altmann, E. G. (2018). A network approach to topic models. *Science Advances*, *4*(7), eaaq1360. doi:10.1126/sciadv.aaq1360

Honnibal, M., & Montani, I. (2017). *Spacy: Industrial-strength natural language processing*. Retrieved from https://spacy.io

Ignatow, G. (2015). Theoretical foundations for digital text analysis. *Journal for the Theory of Social Behaviour*, *46*(1), 104–120. doi:10.1111/jtsb.12086

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., et al. (2016). Jupyter notebooks: A publishing format for reproducible computational workflows. In F. Loizides & B. Schmidt (Eds.), *Positioning and power in academic publishing: Players, agents and agendas. Proceedings of the 20th International Conference on Electronic Publishing* (pp. 87–90). Amsterdam: IOS Press. doi:10.3233/978-1-61499-649-1-87

McKinney, W. (2013). *Python for data analysis*. Sebastopol, Calif.: O'Reilly.

Traag, V. (2020). *Leidenalg*. doi:10.5281/zenodo.1469356

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., et al. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. doi:10.1038/s41592-019-0686-2