WordGraph: A Python Package for Reconstructing Interactive Causal Graphical Models from Text Data

Amine Ferdjaoui
SogetiLabs & Centre Borelli
Université Paris Cité, France
amine.ferdjaoui@etu.u-paris.fr

Séverine Affeldt
Centre Borelli UMR 9010
Université Paris Cité, France
severine.affeldt@u-paris.fr

Mohamed Nadif
Centre Borelli UMR 9010
Université Paris Cité, France
mohamed.nadif@u-paris.fr

ABSTRACT

We present `WordGraph`, a Python package for exploring the topics of documents corpora. `WordGraph` provides causal graphical models from text data vocabulary and proposes interactive visualizations of terms networks. Our ease-to-use package is provided with a pre-built pipeline to access the main modules through *jupyter widgets*. It results in the encapsulation of a whole vocabulary exploration process within a single *jupyter notebook* cell, with straightforward parameters settings and interactive plots. `WordGraph` pipeline is fully customizable by adding/removing widgets or changing default parameters. To assist users with no background in Python nor *jupyter notebook*, but willing to explore large corpora topics, we also propose an automatic dashboard generation from the customizable *jupyter notebook* pipeline in a web application style. `WordGraph` is available through a GitHub repository[1].

CCS CONCEPTS

• **Computing methodologies** → **Cluster analysis**; *Probabilistic reasoning*; • **Mathematics of computing** → **Causal networks**.

KEYWORDS

co-clustering, causal network reconstruction, text data

ACM Reference Format:
Amine Ferdjaoui, Séverine Affeldt, and Mohamed Nadif. 2024. WordGraph: A Python Package for Reconstructing Interactive Causal Graphical Models from Text Data. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24), March 4–8, 2024, Merida, Mexico.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3616855.3635698

1 INTRODUCTION

1.1 Co-clustering for text data

The task of simultaneously partitioning objects (rows) and features (columns) is commonly referred to as co-clustering. It aims to reveal meaningful blocks/co-clusters within a data matrix and typically results in more relevant and interpretable row and column clusters than with one-way clustering [12]. Co-clustering thereby provides dimensionality reduction at the same time on row and column sets and therefore has more potential to discover the hidden pattern(s). Since the seminal work of Hartigan [10], co-clustering (also known as biclustering or block clustering) has found applications in multiple areas such as bioinformatics [2, 3]. Co-clustering is particularly well suited for document-term data matrices, that are by essence high dimensional, sparse and exhibit directional characteristics. The algorithms for such co-occurrences matrices can be derived from various approaches. Spectral co-clustering methods, that treat the input matrix as a bipartite graph between documents and words, approximate the normalized cut of this graph using a real relaxation [5]. The model-based methods derived from appropriate Latent Block Models (LBM) [8], rely on expectation-maximization algorithms [16]. Co-clustering can also use matrix factorization based methods such as Non-negative Matrix Factorization (NMF) [7] or Tri-factorization (NMTF) [15]. By contrast, information-theoretic based methods, used with two-way contingency tables, minimize a loss in mutual information while collapsing rows and columns of the matrix according to the co-clustering [6]; note that the optimized criterion is associated to a particular Poisson LBM [9]. Finally, the co-partitioning is also possible from an adapted version of the modularity measure usually used for networks [11].

The `CoClust` Python package is well-known among the popular packages dedicated to co-clustering for co-occurrence matrices, in particular for document-term matrices in text mining applications [14]. It provides the implementations of three algorithms that proved their efficiency in handling such matrices[2]. While word embeddings derived from sophisticated models have been largely acclaimed, we argue that the association of document-term matrices, which are straightforward *Bag-Of-Words* (*BOW*) text data representations, and co-clustering, remains a successful combination to segment textual data into meaningful topics. `WordGraph` is the first Python package that facilitates a visual exploration of large corpora topics based on co-clustering.

1.2 Causal network inference

The reconstruction of graphical models is a well-known practice in multiple areas. These models can be learned from time series data, controlled perturbation experiments, or observational data, as found in many biological contexts. Traditional methods that exploit unperturbed data, include Bayesian search-and-score, sparse inverse covariance estimation, maximum entropy or diffusion map. Beyond the reconstruction of graphical models, the discovery of causality among multiple variables is of great interest. For instance, it can lead to the identification of key regulatory transcription factors at the origin of complex diseases. Applied to textual data, it

[1] https://github.com/MLDS-software/WordGraph

[2] https://coclust.readthedocs.io/en/v0.2.1/install.html

WSDM '24, March 4–8, 2024, Merida, Mexico

ACM ISBN 979-8-4007-0371-3/24/03…$15.00
https://doi.org/10.1145/3616855.3635698

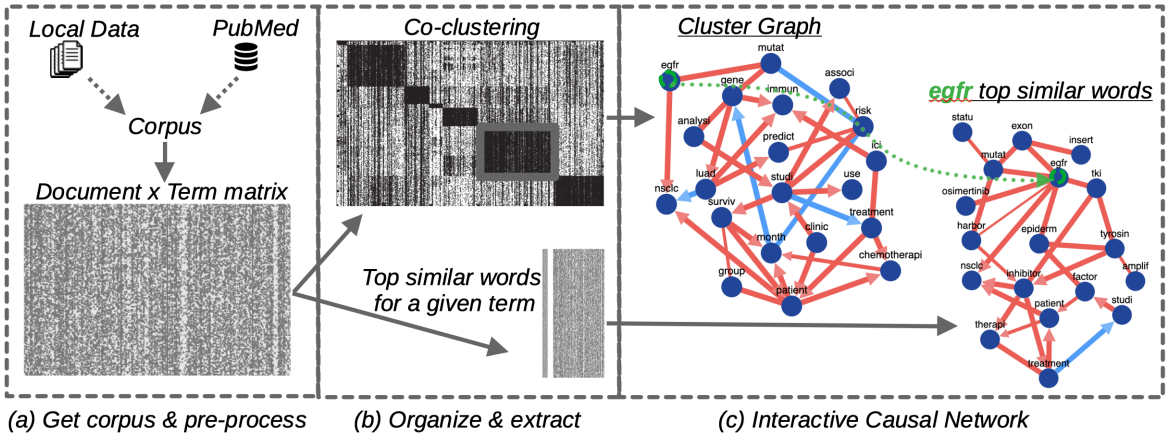

Figure 1: WordGraph workflow, from collecting documents to interactive causal networks between words. An example on lung cancer. *(a) Load or fetch documents, clean data and build document-term matrix. (b) Extract a sub-matrix from co-clustering or cosine similarity scores. (c) Explore topics with causal graphs; red edges: direct co-occurrence, blue edges: antagonist co-occurrence, arrows: word presence causality.*

could provide *graphical summaries* of topics. Causal networks could indicate the organizational flow among words through directed networks. Constraint-based methods [13] can identify structural constraints corresponding to unnecessary edges in a graph while uncovering causality from unperturbed data.

The Multivariate Information-based Inductive Causation algorithm, `MIIC`[3], relies on an information theoretic method that combines constraint-based learning and maximum likelihood framework [1, 4, 17]. Specifically, `MIIC` is based on the analysis of *multivariate information*, which extends the mutual information to more than two variables. In practice, the `MIIC` integration of constraint-based methods within an information-theoretic framework greatly improves the prediction accuracy, the running time and the scaling capabilities in terms of sample size and network size as compared to traditional approaches. Hence, while `MIIC` has demonstrated its applicability on a variety of genomic datasets at different biological size and times scales [17], it also appears to be particularly well-suited to tackle high dimensional textual data facilitating therefore the exploration of large corpora topics. `WordGraph` is the first Python overlay that applies `MIIC` to textual data. It is also the first package that combines graph reconstruction and co-clustering to discover corpus topics and their interactions.

1.3 Pipeline overall description

Our package unfolds in three parts. First, it creates a corpus of documents either locally loaded or directly fetched from the PubMed[4] online database (Fig. 1, *a*). The creation step includes the textual data pre-processing, and the construction of a document-term matrix, with a weighting TF-IDF (Term Frequency-Inverse Document Frequency) procedure. Then, `WordGraph` fits a co-clustering on the document-term matrix to explore the topics, based on `Coclust` (Fig. 1, *b*, top). In this step, a sub-matrix of the most similar terms of a given word can be extracted from the document-term matrix, based on the *cosine similarity* score (Fig. 1, *b*, bottom). As a Python overlay of the `MIIC` R package, `WordGraph` can provide a graphical model from the most representative terms – typically, the most frequent words – of a co-cluster (Fig. 1, *c*, top). The interactive graph visualizations use the `ipycytoscape`[5] Python library. The user can reorganize the nodes, obtain magnified views or click on a node to build a secondary graph, based on the *top similar words* extracted from the document-term matrix (Fig. 1, *c*, bottom).

In the following, we detail the `WordGraph` multilevel capabilities. Indeed, it can either be used as is, in a regular Python source, or the user can customize an interactive *jupyter notebook*. Most importantly, a web application can instantaneously be generated from this notebook source to allow an easy online sharing. `WordGraph` is the first package that combines efficient co-clustering and robust causal graphical model for text data. As it relies on straightforward *BOW* representations, `WordGraph` avoid the computational and memory costs of sophisticated language models.

2 A MULTILEVEL PYTHON PACKAGE

`WordGraph` uses the functionalities of `Coclust` and `MIIC` at 3 levels for exploring textual data. The main objective is to enable the exploration of corpora topics through causal graphical models. In particular, `WordGraph` proposes an interactive notebook for word graphical models reconstruction enhanced by a customizable web application.

2.1 A Python overlay for words causal network

The `WordGraph` package is the first Python package that proposes an overlay over the `MIIC` R package for textual data. Figure 2 provides the main functionalities of `WordGraph` and their associated modules.

The package can locally load a corpus or fetch documents from PubMed (*io* module). `WordGraph` embeds MIIC in the *build_graph* method, and relies on the functions *create_miic* and *preproc_graphML* (*utils* module). *utils* also enables the corpus pre-processing. A simple usage example of `WordGraph`, from the PubMed fetching to the `MIIC` graph object computation, can be found in the *Simple usage*

[3]See https://github.com/miicTeam/miic_R_package for the `MIIC` R implementation
[4]https://pubmed.ncbi.nlm.nih.gov; more than 35 millions biomedical citations
[5]https://ipycytoscape.readthedocs.io/en/latest/

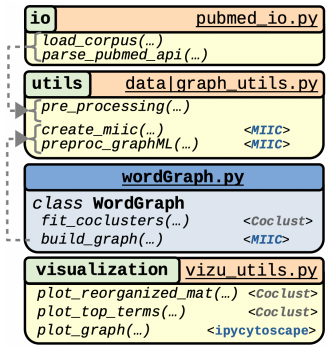

Figure 2: WordGraph package: main functionalities.

section of the online GitHub presentation of the package[6]. We provide in `WordGraph` a *jupyter notebook* that embeds this simple usage proposal (see *WordGraph simple usage.ipynb*).

2.2 An interactive notebook for an easy design

`WordGraph` offers visualization functionalities from `ipycytoscape` package. In particular, *plot_graph* displays interactive networks within a *jupyter notebook*, where the layout can be directly adjusted with the mouse. It gives a graph of clusters' top terms and hence, easily summarizes contents (Fig. 1, *(c)*, top). By clicking on a node, it displays a secondary interactive graph, in which the vertices are derived by their similarity to the selected node (Fig. 1, *(c)*, bottom).

Most importantly, `WordGraph` proposes an easily customizable *jupyter notebook* (*WordGraph Custom.ipynb*), where the whole pipeline can be embedded in a single cell. To this aim, it embeds several `ipywidgets` elements (eg. slider, button, text or numerical inputs). Figure 3 shows how the pipeline is organized in a single cell. Modifying or adding a new *widget* takes only a few lines of code. The *Customizable notebook pipeline* section of the online GitHub presentation of the package[7] provides the implementation for setting a *confirm* button and calling the corresponding function on click.

2.3 An interactive web application

The *WordGraph Custom.ipynb* was also designed to be automatically turned into a web application, with the *voilà*[8] package. Hence, the user can exploit the *WordGraph Custom.ipynb* as is or customize it, and then easily obtain the corresponding web application, dockerized and ready-to-use for users non familiar with coding (Fig. 4).

3 QUALITATIVE EVALUATIONS

We provide qualitative evaluations of `WordGraph` abilities to give topics summaries for a large corpus. First, we consider a biomedical application on **lung cancer**, based on 10,000 abstracts fetched from PubMed by our pipeline. Second, we consider the `PubMed10` [9]

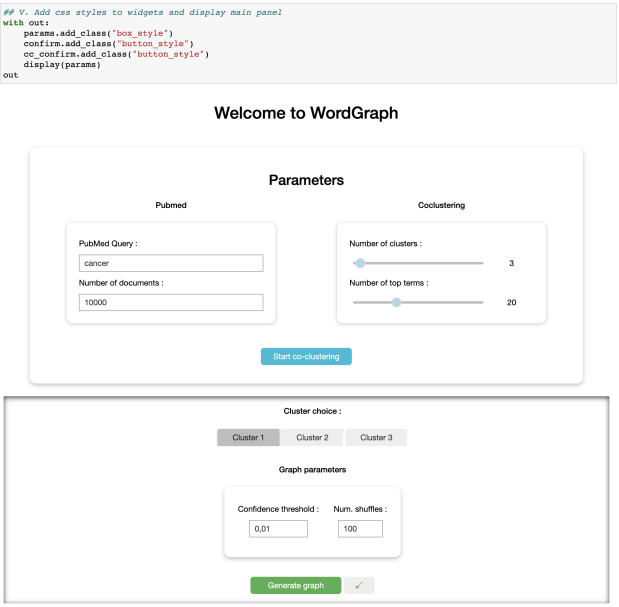

Figure 3: Interactive pipeline in a jupyter notebook.

benchmark dataset. `WordGraph` explores the topics by fitting a co-clustering that efficiently segments the vocabulary, and then automatically learns a causal graphical model for each co-cluster.

In Figure 1, the opposition (blue edge) between *NSCLC* (Non-Small Cell Lung Cancer) and *LUAD* (LUng ADenocarcinoma) suggests that documents dealing with *LUAD* ignore *NSCLC* topic. Indeed, while *LUAD* belongs to *NSCLC*, their prognosis and gene expression are distinct and they are seen as separated entities. *EGFR*-positive NSCLC emerges from mutated EGFR gene (Epidermal Growth Factor Receptor). Our graph causally relates *EGFR* to the *NSCLC* acronym. *ICI* (Immune Checkpoint Inhibitor) *treatment* is also causally related with the *immun* node. The secondary graph, based on *egfr*, focuses on the *treatment*, which usually aims to prevent abnormal tumor growth by targeting *Tyrosin* Kinase *Inhibitor* (*TKI*) as with *osimertinib* drug.

We also provide a graphical summary of the *Age-Macular Degeneration (AMD) disease* from `PubMed10` (Fig. 5, left), and compare it with a pairwise graph model using `BioWordVec` embeddings [18] and cosine similarity scores (Fig. 5, right). Edges are retained based on a threshold (typically 0.5), a small angle between two word vectors being a good indicator of similarity. `WordGraph` network thereby provides clearer interactions as it only harbors direct relationships, following on `MIIC` algorithm. Furthermore, `WordGraph` requires no arbitrary edges threshold, but automatically estimates a complexity penalty to discover unnecessary edges. Importantly, `WordGraph` needs no heavily pretrained model that would be specifically fitted on an appropriate large corpus. Finally, we can highlight the biomedical relevance of the `WordGraph` network. *edema* is upstream of *macular*, but also of *intravit*real injection, the main AMD treatment. *diabet*es, a risk factor for AMD, is also found upstream of *macular*. We also notice that *cnv* (Choroidal Neovascularization) is located dowstream of *neovascular*ization.

[6] https://github.com/MLDS-software/WordGraph#1-simple-usage
[7] https://github.com/MLDS-software/WordGraph#2-customizable-notebook-pipeline
[8] https://voila.readthedocs.io/en/stable/using.html
[9] 15,565 × 22,437 documents×term, 10 expected classes corresponding to 10 diseases

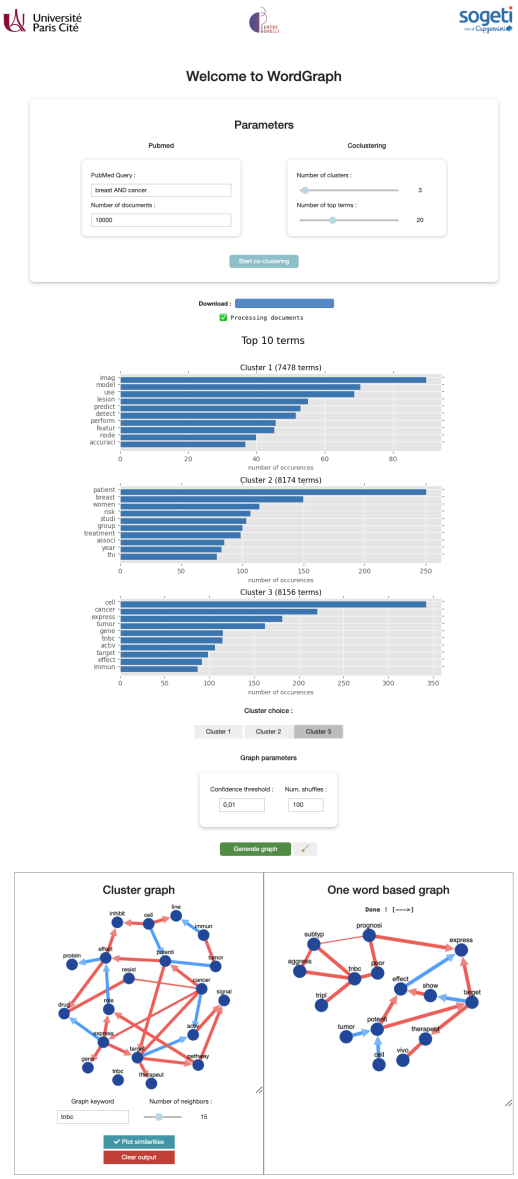

Figure 4: Customizable web application.

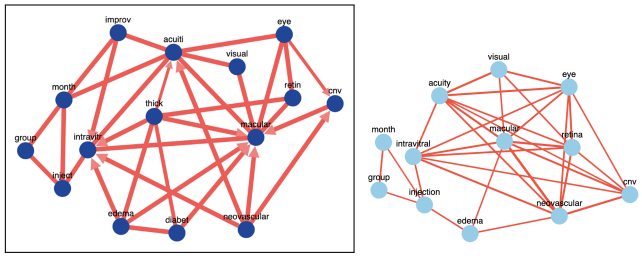

Figure 5: (left) WordGraph; (right) Embeddings similarity .

4 CONCLUSION

The exploration of large corpora topics is becoming harder as huge amount of textual data are being made available through multiple areas. We propose `WordGraph` as an ease-to-use Python package for users willing to retrieve as much information as possible from various topic areas. We have made particular efforts in order to turn `WordGraph` into a standalone web application for users with no background in Python nor *jupyter notebook*, while still allowing for customizable interface. Our qualitative evaluations on lung cancer and `PubMed10` data show promising results, in particular as compared to word embeddings graphical models. The scalability of co-clustering combined with the robustness of causal network reconstruction within `WordGraph` provides a valuable topics exploration package.

ACKNOWLEDGMENTS

This work was supported by a grant overseen by the French National Research Agency (ANR) (ANR-19-CE23-0002). We thank the Isambert Lab for its helpful support on `MIIC`.

REFERENCES

[1] Séverine Affeldt and Hervé Isambert. 2015. Robust Reconstruction of Causal Graphical Models based on Conditional 2-point and 3-point Information. In *ACI@ UAI*. 1–29.
[2] Séverine Affeldt, Lazhar Labiod, and Mohamed Nadif. 2021. Regularized bi-directional co-clustering. *Statistics and Computing* 31, 3 (2021), 32.
[3] Séverine Affeldt, Lazhar Labiod, and Mohamed Nadif. 2021. Regularized Dual-PPMI Co-clustering for Text Data. In *the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2263–2267.
[4] Séverine Affeldt, Louis Verny, and Hervé Isambert. 2016. 3off2: A network reconstruction algorithm based on 2-point and 3-point information statistics. In *BMC bioinformatics*, Vol. 17. BioMed Central, 149–165.
[5] Inderjit S Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *the 7^{th} SIGKDD*. 269–274.
[6] Inderjit S Dhillon, Subramanyam Mallela, and Dharmendra S Modha. 2003. Information-theoretic co-clustering. In *the 9^{th} SIGKDD*. 89–98.
[7] Mickael Febrissy, Aghiles Salah, Melissa Ailem, and Mohamed Nadif. 2022. Improving NMF clustering by leveraging contextual relationships among words. *Neurocomputing* 495 (2022), 105–117.
[8] Gérard Govaert and Mohamed Nadif. 2008. Block clustering with Bernoulli mixture models: Comparison of different approaches. *Computational Statistics & Data Analysis* 52, 6 (2008), 3233–3245.
[9] Gérard Govaert and Mohamed Nadif. 2018. Mutual information, phi-squared and model-based co-clustering for contingency tables. *Advances in data analysis and classification* 12 (2018), 455–488.
[10] John A Hartigan. 1972. Direct clustering of a data matrix. *Journal of the american statistical association* 67, 337 (1972), 123–129.
[11] Lazhar Labiod and Mohamed Nadif. 2011. Co-clustering for binary and categorical data with maximum modularity. In *the 11^{th} ICDM*. IEEE, 1140–1145.
[12] Lazhar Labiod and Mohamed Nadif. 2014. A unified framework for data visualization and coclustering. *IEEE Transactions on Neural Networks and Learning Systems* 26, 9 (2014), 2194–2199.
[13] Judea Pearl and Thomas S Verma. 1995. A theory of inferred causation. In *Studies in Logic and the Foundations of Mathematics*. Vol. 134. Elsevier, 789–811.
[14] François Role, Stanislas Morbieu, and Mohamed Nadif. 2019. Coclust: a python package for co-clustering. *Journal of Statistical Software* 88 (2019), 1–29.
[15] Aghiles Salah, Melissa Ailem, and Mohamed Nadif. 2018. Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering. In *the AAAI Conference on Artificial Intelligence*, Vol. 32.
[16] Aghiles Salah and Mohamed Nadif. 2019. Directional co-clustering. *Advances in Data Analysis and Classification* 13 (2019), 591–620.
[17] Louis Verny, Nadir Sella, Séverine Affeldt, Param Priya Singh, and Hervé Isambert. 2017. Learning causal networks with latent variables from multivariate information in genomic data. *PLoS computational biology* 13, 10 (2017), e1005662.
[18] Yijia Zhang, Qingyu Chen, Zhihao Yang, Hongfei Lin, and Zhiyong Lu. 2019. BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific data* 6, 1 (2019), 52.