Information Retrieval & Document Understanding (IT MSc)

Topic Modeling & Clustering

Carlos-Emiliano González-Gallardo
carlos.gonzalez_gallardo@univ-lr.fr

Laboratoire L3i
La Rochelle Université

October, 2022

Summary

Overview

Topic modeling & clustering

- "classical" approach (tfi-df, bag-of-words, LDA, ...)
- "advanced" approach (Transformers, SBERT, UMAP, HDBSCAN, ...)

Planning

- 2 x (1.5h CM + 3h TP)

TP

- apply classical & advanced approaches on Europarl corpus
- scientific presentation and comparative results between both approaches (end of second session)
- work to be done in pairs

Why Topic Modeling ?

Back to the basics

Preprocessing for query and documents

- tokenization, normalization (e.g., lemmatization / stemming), filtering, treatment of synonyms and antonyms ...

Vector space model (VSM) $\rightarrow$ *bag-of-words*

- binary : 1 if term j in sentence μ, 0 otherwise ;
- frequency ($\mathbf{tf}_{\mu,j}$) : number of occurrences of j in μ ;
- corrective : corrected $\mathbf{tf}_{\mu,j}$ taking into account distribution of j in corpus (e.g., $\mathbf{tf}_{\mu,j} \times \mathbf{idf}_j = \mathbf{tf}_{\mu,j} \times \ln \frac{N}{\mathbf{df}_j}$).

Back to the basics

Distance between vectors

- $sim(q, A) := \cos(q, A) = \frac{q \cdot A}{|q| \cdot |A|}$

Ranking

- $sim(q, A) > sim(q, B) > sim(q, C) > sim(q, D)$

Indexing

- Avoids going through all the documents to find the relevant ones.

Back to the basics

Evaluation metrics

- Confusion matrix

		Reference	
		Relevant	**Not relevant**
Predicted	**Retrieved**	**TP**	**FP**
	Not retrieved	**FN**	**TN**

- Precision : fraction of retrieved documents that are relevant

$$P = \frac{TP}{TP + FP}$$

- Recall : fraction of relevant documents that are retrieved

$$R = \frac{TP}{TP + FN}$$

- F-score : weighted harmonic mean of P and R

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R}$$

Bag-of-words limits

- large sparse matrices
- unable to handle synonymy and polysemy
- no relation between words (2-grams, 3-grams, ...)
- . . .
- How to sort document into topics ?

Topic modeling

Refers to :

Unsupervised machine learning technique capable of scanning a set of documents, detecting patterns of words and phrases within them, and automatically clustering groups of similar words and expressions that best characterize a set of documents.

- Latent Semantic Indexing (LSI)
- probabilistic latent semantic indexing (PLSI)
- **Latent Dirichlet Allocation (LDA)**
- ...

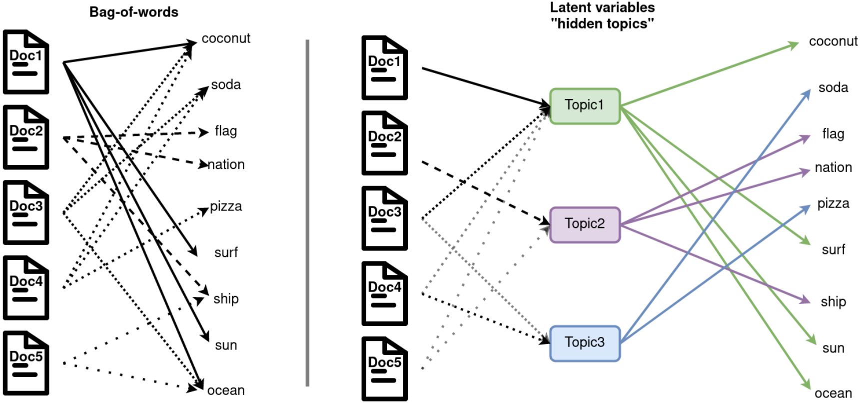
Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA)

"...a generative statistical model that explains a set of observations through unobserved groups, and each group explains why some parts of the data are similar."

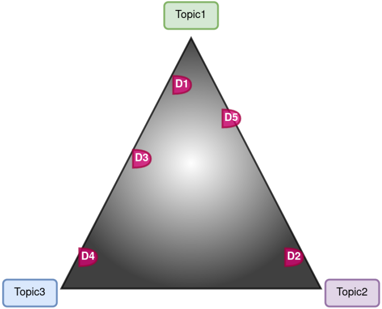
How does LDA works ?

LDA *extracts* certain sets of topic according to documents we fed to it.

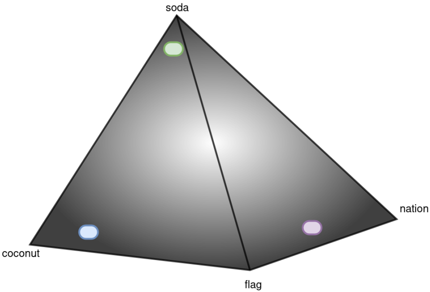


How does LDA works ?

- documents represented in a space of topics
- topics represented in a space of words



Space of topics



Space of words

- **geometric approach :** documents are closer to the corner of the topic they belong to

How does LDA works ?

Assumptions :

- documents with similar topics use similar groups of words ;
- latent variables "hidden topics" can be extracted by searching for groups of **words that frequently occur together in documents across the corpus** (distributional semantics) ;
- documents and topics are Dirichlet probability distributions.

Probability density function for Dirichlet distributions

$$f(x_1, \ldots, x_K; \alpha_1, \ldots, \alpha_K) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}; \alpha > 0$$

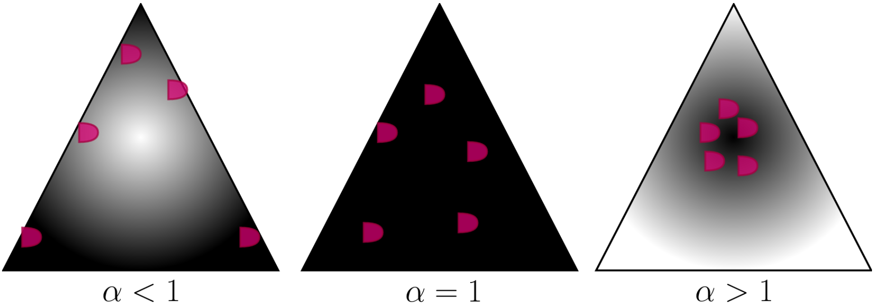
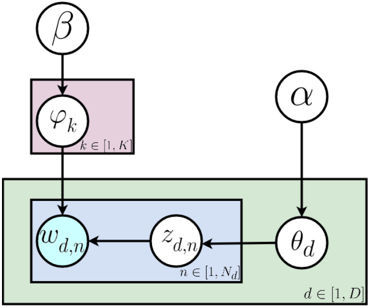


Plate notation of LDA

- K : topics
- D : documents
- N_d : words in document d
- $w_{d,n}$: word n in document d
- $z_{d,n}$: topic of $w_{d,n}$
- φ_k : word distribution for topic k
- θ_d : topic distribution for document d
- α : controls per-document topic distribution
- β : controls per-topic word distribution



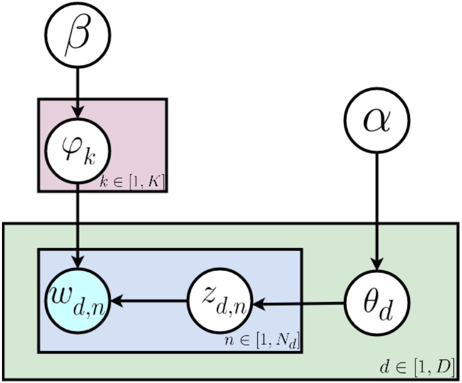
Total probability of the model

$$P(\boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\theta}, \boldsymbol{\varphi}; \alpha, \beta) = \prod_{d=1}^{D} P(\theta_d; \alpha) \prod_{k=1}^{K} P(\varphi_k; \beta) \left(\prod_{n=1}^{N_d} P(Z_{d,n}|\theta_d) P(W_{d,n}|\varphi_{Z_{d,n}}) \right)$$

- α, β : Dirichlet distributions
- θ, φ : Multinomial distributions

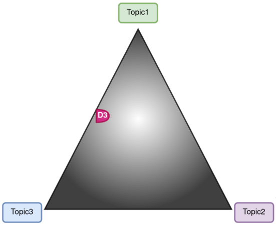
Goal

Maximize $P(\boldsymbol{W}, \boldsymbol{Z}, \boldsymbol{\theta}, \boldsymbol{\varphi}; \alpha, \beta)$

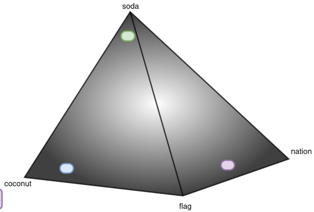


Constructing documents

$$\prod_{d=1}^{D} P(\theta_d; \alpha) \qquad \prod_{k=1}^{K} P(\varphi_k; \beta) \qquad \prod_{n=1}^{N_d} P(Z_{d,n}|\theta_d) \qquad P(W_{d,n}|\varphi_{Z_{d,n}})$$



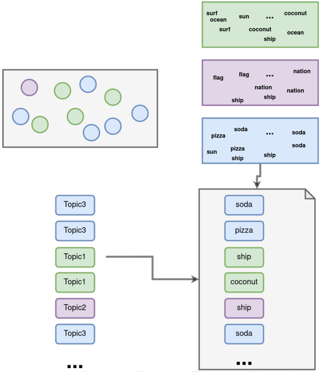
0.4	0.1	0.5



coconut	soda	flag	nation	pizza	surf	ship	sun	ocean
0.2	0.05	0.05	0.05	0.05	0.2	0.1	0.1	0.2

coconut	soda	flag	nation	pizza	surf	ship	sun	ocean
0.05	0.05	0.2	0.3	0.05	0.05	0.2	0.05	0.05

coconut	soda	flag	nation	pizza	surf	ship	sun	ocean
0.1	0.3	0.05	0.05	0.2	0.05	0.1	0.1	0.05



In practice

Find the most optimal representation of the document-topic matrix and the topic-word matrix to find the most optimized document-topic distribution (α) and topic-word distribution (β).

Document Term Matrix (DTM)

	W1	W2	W3	W4	W5	W6	W7	W8
D1	0	1	1	0	1	1	0	1
D2	1	1	1	1	0	1	1	0
D3	1	0	0	0	1	0	0	1
D4	1	1	0	1	0	0	1	0
D5	0	1	0	1	0	0	1	0

Shape: 5 * 8

Document Topic Matrix

	K1	K2	K3	K4	K5	K6
D1	1	0	0	0	0	0
D2	0	1	0	0	1	1
D3	1	1	0	0	0	0
D4	1	0	0	1	0	1
D5	0	0	1	1	0	0

Shape: 5 * 6

Topic Word Matrix

	W1	W2	W3	W4	W5	W6	W7	W8
K1	0	1	1	0	1	0	1	0
K2	1	1	1	1	0	1	1	1
K3	1	0	0	0	0	1	0	0
K4	1	1	0	1	1	0	0	1
K5	0	0	1	1	0	1	1	1
K6	1	0	1	1	1	0	0	1

Shape: 6 * 8

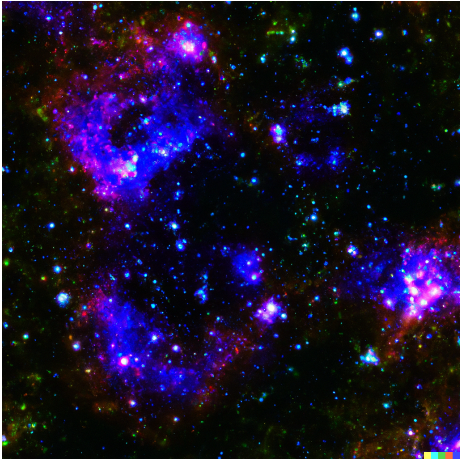
https ://editor.analyticsvidhya.com/uploads/26864dtm.JPG

Clustering

Introduction

Clustering

- is unsupervised learning approach ;
- aims to assign a set of document samples into groups such that documents in a group are more similar to each other than documents in different groups ;
- can be classified based in **exclusivity** and **hierarchy**.



Exclusivity

The property of the clustering algorithm to assign a document to one or more groups.

- **exclusive / hard clustering :** assigns a document to one and only one group
- **non-exclusive / soft clustering :** allows a document to belong to one or more groups with a certain degree of membership

Hierarchy

Takes into consideration the structure produced by the clustering algorithm.

flat / non-hierarchical clustering

- Produces a number of groups wit an undetermined relation between them.
- Normally originated by iterative algorithms which start with a determined number of groups thus reallocating the documents by an iterative process.

hierarchical clustering

- Produce a stratified relation between groups where each group corresponds to a subgroup of its parent.
- The tree structure can be constructed bottom-up (*divisive*) or top-down (*agglomerative*).

Some clustering algorithms

		Exclusivity	
		exclusive	**non-exclusive**
Hierarchy	**flat**	k-means [11, 12], mean-shift [5], DBSCAN [8], OPTICS [2], affinity propagation [9]	EM [7], fuzzy [21], LSI [6]
	hierarchical	{*divisive*}{Min-cut [14], DIANA [19]}, {*agglomerative*} {Ward [20], CURE [10], HDBSCAN [3] }	

https://scikit-learn.org/stable/modules/clustering.html#overview-of-clustering-methods

Evaluation

The ideal clustering is characterised by **minimal intra-cluster distance** and **maximal inter-cluster distance**.

Extrinsic measures

- need of ground truth labels
- e.g., Rand index, Mutual Information, homogeneity_completeness_V-measure, Fowlkes-Mallows score, etc.

Intrinsic measures

- do not require ground truth labels
- e.g., **Silhouette coefficient**, Calinski-Harabasz index, Davies-Bouldin index, etc.

https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation

Silhouette coefficient

$$S = \frac{1}{N}\sum_{i=1}^{N}\frac{b_i - a_i}{\max(a_i, b_i)}; S \in [-1, 1]$$

- How well samples are clustered with samples that are similar to themselves.
- A higher score relates to a model with better defined clusters.
- a_i : mean intra-cluster distance of sample i
- b_i : mean nearest-cluster distance of sample i
- $S \approx 1$: well defined clusters
- $S \approx 0$: overlapping clusters
- $S \approx -1$: samples has been assigned to the wrong cluster

TP1 : What is the European Parliament talking about ?

TP1 : What is the European Parliament talking about ?

Objectives :

- Integrate knowledge from previous sessions
- Apply topic modeling and clustering to real data
- Interpret and analyse results

Prerequisites :

- Python 3
- Jupyter Notebook
- Google Colab

What to do ? :

- Download the Europarl corpus from the site course.
- Load, pre-process, transform documents into weighted vectors and train an LDA model.
- Choose a clustering algorithm and group documents before and after being process by LDA.
- Evaluate clusters and compare results.

Useful links :

- `https://scikit-learn.org/`
- `https://radimrehurek.com/gensim/index.html`
- `https://www.nltk.org/`

Distributed Representations of Topics (top2vec)

Generative statistical model limits

- number of topics are required a priori
- language and corpus specific tokenization, normalization (e.g., lemmatization / stemming), filtering and treatment of synonymy & polysemy
- word order and semantics are ignored
- ...

Distributed representations of words & documents

- `word2vec` [16, 17] : It learns word similarity by predicting which adjacent words should be present to a given context.
- `doc2vec` [13] : In addition to the context window of words, a paragraph vector is also used to predict which adjacent words should be present.

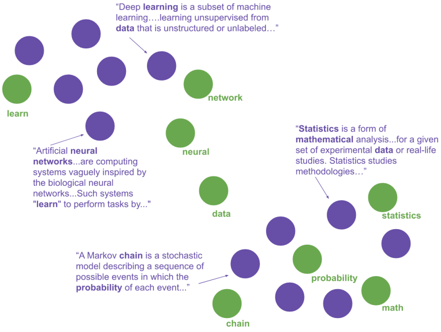
`top2vec` [1]

- leverage document and word semantic embeddings to find topic vectors
- resulting topics are jointly embedded with the **document** and **word** vectors
- distances represent semantic similarity

The semantic space 1/2

The **semantic space** is a continuous representation of **topics** in which each point is a different topic best summarized by its nearest words.

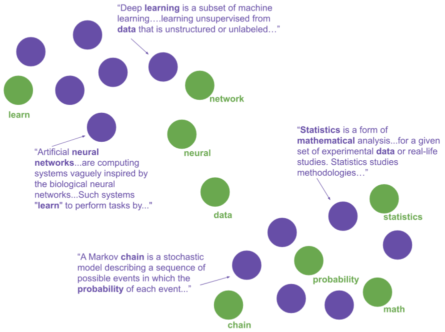
- jointly embedded document and word vectors
- words are closer to the documents they best represent
- similar documents are close together
- doc2vec [13], Universal Sentence Encoder [4], Sentence-BERT [18]



https ://github.com/ddangelov/Top2Vec

The semantic space 2/2

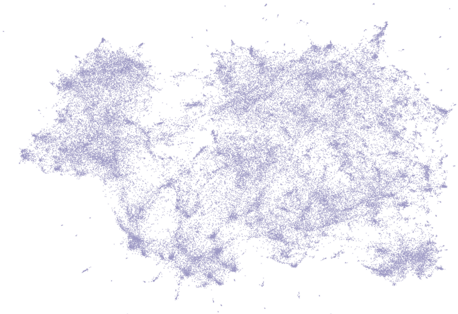
- dense area of documents → many documents that have a similar topic
- the number of dense areas is assumed to be the number of prominent topics
- topics vectors are calculated as the **centroids** of each dense area of document vectors



https ://github.com/ddangelov/Top2Vec

Low dimensional document embedding

- dimension reduction helps for finding dense areas
- Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP) [15] :
  - preserves local & global structure
  - scalable to large datasets

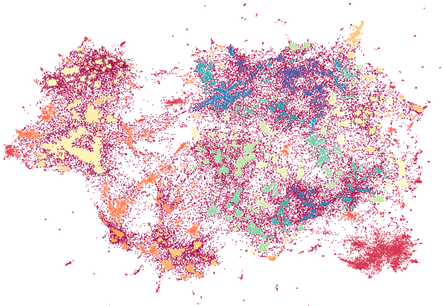


https ://github.com/ddangelov/Top2Vec

Find dense clusters of documents

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [3]

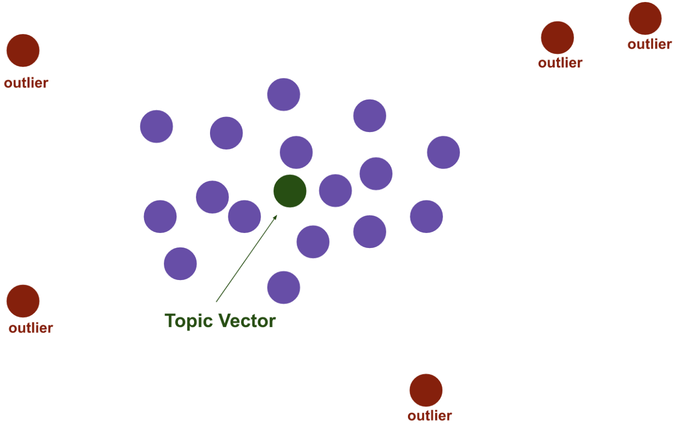
- clusters as areas of high density separated by areas of low density
- assigns a label to each dense cluster
- documents vectors not in a dense cluster → noise



https ://github.com/ddangelov/Top2Vec

Calculate topic vectors

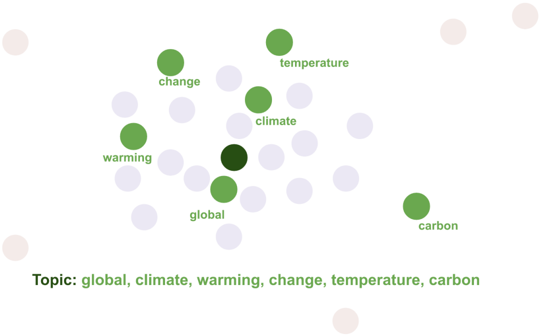
Topic vectors → centroids of dense areas (in original semantic space)



https ://github.com/ddangelov/Top2Vec

Retrieve topic words

- The word vectors closest to a topic vector are those that are most semantically representative of it.
- Common words are in regions of the semantic space that are equally distant from all documents.



https ://github.com/ddangelov/Top2Vec

TP2 : Semantic space of the European Parliament

TP2 : Semantic space of the European Parliament

Objectives :

- Integrate knowledge from previous sessions
- Apply `top2vec` to real data
- Interpret, analyse & compare results

Prerequisites :

- Python 3
- Jupyter Notebook
- Google Colab

What to do ? :

- Download the Europarl corpus from the site course.
- Install, configure and run `top2vec` over data
- Evaluate clusters and compare results.

Useful links :

- `https://github.com/ddangelov/Top2Vec`
- `https://top2vec.readthedocs.io/en/latest/api.html`
- `https://www.sbert.net/docs/pretrained_models.html`
- `https://umap-learn.readthedocs.io/en/latest/index.html`
- `https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html`

References I

1. Angelov, D. Top2vec : Distributed representations of topics. *arXiv preprint arXiv :2008.09470* (2020).
2. Ankerst, M., Breunig, M. M., Kriegel, H.-P. & Sander, J. OPTICS : ordering points to identify the clustering structure. *ACM Sigmod record* **28,** 49-60 (1999).
3. Campello, R. J., Moulavi, D. & Sander, J. *Density-based clustering based on hierarchical density estimates.* in *Pacific-Asia conference on knowledge discovery and data mining* (2013), 160-172.
4. Cer, D. *et al. Universal sentence encoder for English.* in *Proceedings of the 2018 conference on empirical methods in natural language processing : system demonstrations* (2018), 169-174.
5. Comaniciu, D. & Meer, P. Mean shift : A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence* **24,** 603-619 (2002).

References II

6. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. & Harshman, R. Indexing by latent semantic analysis. *Journal of the American society for information science* **41,** 391-407 (1990).
7. Dempster, A. P., Laird, N. M. & Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society : Series B (Methodological)* **39,** 1-22 (1977).
8. Ester, M., Kriegel, H.-P., Sander, J., Xu, X. *et al. A density-based algorithm for discovering clusters in large spatial databases with noise..* in *Kdd* **96** (1996), 226-231.
9. Frey, B. J. & Dueck, D. Clustering by passing messages between data points. *science* **315,** 972-976 (2007).
10. Guha, S., Rastogi, R. & Shim, K. CURE : an efficient clustering algorithm for large databases. *ACM Sigmod record* **27,** 73-84 (1998).
11. Hartigan, J. A. *Clustering algorithms.* (John Wiley & Sons, Inc., 1975).

References III

12. Hartigan, J. A. & Wong, M. A. Algorithm AS 136 : A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)* **28,** 100-108 (1979).

13. Le, Q. & Mikolov, T. *Distributed representations of sentences and documents.* in *International conference on machine learning* (2014), 1188-1196.

14. Lengauer, T. Combinatorial algorithms for integrated circuit layout. (1990).

15. McInnes, L., Healy, J. & Melville, J. Umap : Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv :1802.03426* (2018).

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

References IV

17. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* **26** (2013).

18. Reimers, N. & Gurevych, I. Sentence-bert : Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv :1908.10084* (2019).

19. Rousseeuw, P. J. & Kaufman, L. Finding groups in data. *Hoboken : Wiley Online Library* **1** (1990).

20. Ward Jr, J. H. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association* **58,** 236-244 (1963).

21. Yan, J.-T., Hsiao, P.-Y. *et al.* A Fuzzy clustering-algorithm for graph bisection. *Information Processing Letters* **52,** 259-263 (1994).