Hashtag Discernability - Competitiveness Study of Graph Spectral and Other Clustering Methods

Bartłomiej Starosta, Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, Dariusz Czerski
Institute of Computer Science, Polish Academy of Sciences
ul. Jana Kazimierza 5, 01-248 Warsaw, Poland
Email: b.starosta,klopotek,stw,d.czerski@ipipan.waw.pl

***Abstract*—Spectral clustering methods are claimed to possess ability to represent clusters of diverse shapes, densities etc. They constitute an approximation to graph cuts of various types (plain cuts, normalized cuts, ratio cuts). They are applicable to unweighted and weighted similarity graphs. We perform an evaluation of these capabilities for clustering tasks of increasing complexity.**

I. Introduction

Document clustering (or text clustering) has a multitude of applications, including automatic document organization, topic extraction, fast information retrieval, filtering, authorship discovery, topic drift detection in news streams and social media etc.

Two clustering methods are of particular interest in this area, the Graph Spectral Clustering (GSC) and spherical k-means.

Graph Spectral Clustering methods [20] are generally praised for possessing ability to represent clusters of diverse shapes, densities etc. They constitute an approximation to graph cuts of various types (plain cuts, normalized cuts, ratio cuts). They are applicable to unweighted and weighted similarity graphs.

Spherical k-means [3] is a variant of k-means that measure similarity of documents based on their cosine similarity, that is quite popular in the domain of text analysis (e.g. for search engines).

In this paper we pose the question: If the grouping method correctly groups certain datasets, can we expect that a combination of these datasets will also be correctly clustered? We will examine the following problem in more detail. Assume that a clustering method can cluster correctly documents from categories $[A, B]$, $[B, C]$, and $[C, A]$. Can we expect the algorithm to cluster correctly data from the mixed set $[A, B, C]$?. Let us illustrate this with three datasets, tweets, marked with (single) tags 'lolinginlove', 'tejran', 'anjisalvacion'.

The Python implementation of spectral clustering with the affinity matrix constructed from a k-nearest neighbors connectivity matrix with $k = 10$,[1] produced for the hashtags 'lolinginlove', 'tejran', in one of the runs the clustering illustrated in Fig. 1.

For the hashtags 'tejran', 'anjisalvacion' the nearest neighbour spectral clustering achieves the best clustering agreement visible in Fig. 2.

[1] Consult https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html for details.

```
               T\P |     0     1
                 -------------------------
  lolinginlove:  0|  1258     0 | 1258
        tejran:  1|     8   337 |  345
                 -------------------------
                   |  1266   337 | 1603\1603
F-score: 0.990046
```

Fig. 1. Spectral clustering with affinity "nearest neighbours" example

```
               T\P |     0     1
                 -------------------------
        tejran:  0|   324    21 |  345
 anjisalvacion:  1|     5   727 |  732
                 -------------------------
                   |   329   748 | 1077\1077
F-score: 0.968385
```

Fig. 2. Spectral clustering with affinity "nearest neighbours" example 2

For the hashtags 'lolinginlove', 'anjisalvacion', the nearest neighbour spectral clustering achieves the clustering agreement visible in Fig. 3.

So, for each pair of the three hashtags we see a very good agreement of clusterings with the target (hashtags). If we look at the hashtags ['lolinginlove', 'tejran', 'anjisalvacion'], we get clustering agreement visible in Fig. 4.

In this paper we study the extent to which this behaviour extends to larger number of clusters. This study is a starting point for a future revision of the studied clustering algorithms.

```
               T\P |     0     1
                 -------------------------
  lolinginlove:  0|  1258     0 | 1258
 anjisalvacion:  1|     0   732 |  732
                 -------------------------
                   |  1258   732 | 1990\1990
F-score: 1.000000
```

Fig. 3. Spectral clustering with affinity "nearest neighbours" example 3

```
       T\P |     0     1     2
               -----------------------
 loling:  0|  1258     0     0 | 1258
 tejran:  1|     7   314    24 |  345
anjisal:  2|     0     5   727 |  732
               -----------------------
           |  1265   319   751 | 2335\2335
F-score: 0.970334
```

Fig. 4. Spectral clustering with affinity "nearest neighbours" example 4

II. Conceptual Considerations

Despite the example shown above, it is not entirely obvious that given a grouping method that allows to correctly group documents from the categories $[A, B]$, $[B, C]$, $[C, A]$, we can expect that the algorithm will correctly group data from the mixed set $[A, B, C]$.

If the sets $A \cup B$, $B \cup C$ and $C \cup A$ have block diagonal document similarity matrices (after proper reordering the documents), and the blocks are actually within A, B, C then in fact the $[A, B, C]$ similarity matrix will be block diagonal too so that GSC algorithm will cluster A, B, C correctly. This can be seen immediately by inspection of block matrix structure, i.e.

$$S_{A,B} = \begin{bmatrix} S_{A,A} & 0 \\ 0 & S_{B,B} \end{bmatrix} \; S_{B,C} = \begin{bmatrix} S_{B,B} & 0 \\ 0 & S_{C,C} \end{bmatrix}$$

$$S_{A,C} = \begin{bmatrix} S_{A,A} & 0 \\ 0 & S_{C,C} \end{bmatrix}$$

implies

$$S_{A,B,C} = \begin{bmatrix} S_{A,A} & 0 & 0 \\ 0 & S_{B,B} & 0 \\ 0 & 0 & S_{C,C} \end{bmatrix}$$

Recall that combinatorial Laplacian is computed as $L = D - S$, where S is the similarity matrix and D is the diagonal matrix with elements being sums of corresponding rows of S. Hence

$$L_{A,B} = \begin{bmatrix} L_{A,A} & 0 \\ 0 & L_{B,B} \end{bmatrix} \text{, etc.}$$

and

$$L_{A,B,C} = \begin{bmatrix} L_{A,A} & 0 & 0 \\ 0 & L_{B,B} & 0 \\ 0 & 0 & L_{C,C} \end{bmatrix}$$

Eigenvalues of $L_{A,B}, L_{B,C}, L_{A,C}$ will become eigenvalues of $L_{A,B,C}$ with corresponding eigenvectors being only extended with zeros appropriately. So theoretically it should be easy to separate the sets A, B, C based on eigenvectors of $L_{A,B,C}$. However, this enthusiasm needs to be mitigated because such a pure block structure rarely occurs, see our example figures 1, 3, 2, so the "noise" is inherited in sets with more hashtags as visible in Figure 4. But there are also further concerns. Spectral clustering is based on lowest eigenvalue eigenvectors of respective Laplacians. But as shown in [19], the two lowest eigenvectors of $L_{A,B}, L_{B,C}, L_{A,C}$ do not need to be lowest three eigenvectors of $L_{A,B,C}$. For higher number of clusters, the situation may be more complex.

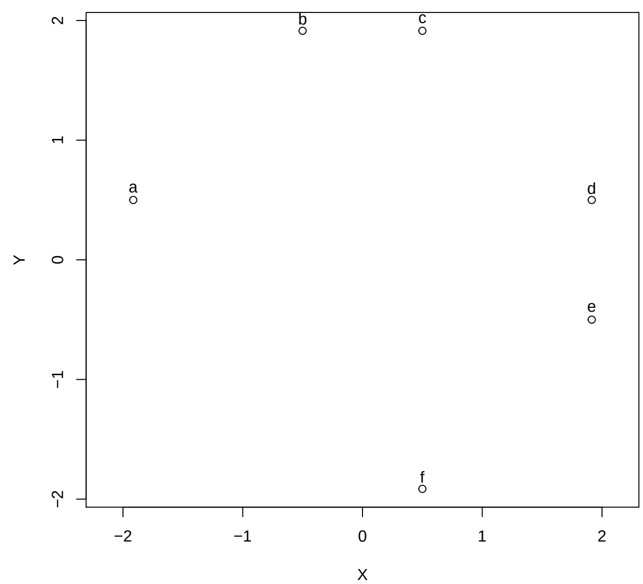

Fig. 5. Visualization of datapoints used to illustrate the increasing clustering problem for k-means

If the dataset $A \cup B \cup C$ is well separated in the sense of k-means algorithm, so that a clustering with k-means will yield A, B, C as clusters, then its application to $A \cup B$, $B \cup C$ or $C \cup A$ will also return correct pairs of clusters. But this is not necessarily true for k-means in the reverse direction. Well-separatedness of $A \cup B$, $B \cup C$ and $C \cup A$ does not imply well-separatedness of $A \cup B \cup C$. Let us illustrate this point with a bit artificial example. Consider the datapoints $\mathbf{a} = (-(0.5+\sqrt{2}), 0.5)$, $\mathbf{b} = (-0.5, 0.5+\sqrt{2})$, $\mathbf{c} = (0.5, 0.5+\sqrt{2})$, $\mathbf{d} = (0.5+\sqrt{2}, 0.5)$, $\mathbf{e} = (0.5+\sqrt{2}, -0.5)$, $\mathbf{f} = (0.5, -(0.5+\sqrt{2}))$, see Fig.5 for visualization. Consider "hashtags" with their "documents" $A = \{\mathbf{a}, \mathbf{b}\}$, $B = \{\mathbf{c}, \mathbf{d}\}$, $C = \{\mathbf{e}, \mathbf{f}\}$, Clustering with k-means of $A \cup C$ into two clusters will yield, A, C, similarly any two hashtag combinations. But clustering with k-means of $A \cup B \cup C$ will yield three clusters $\{\mathbf{a}\}, \{\mathbf{b}, \mathbf{c}\}, \{\mathbf{d}, \mathbf{e}, \mathbf{f}\}$. not A, C, E.

In all these cases, if some noise is added to fuzzify the well-separatedness, the noise can be more destructive for the set A, B, C than for any of the three mentioned subsets – this affects GSC as well as k-means clustering. This is easily imagined by considering k-means algorithm: The cluster center of A when clustering fuzzified A and B may lie in a different position than when clustering fuzzified A and C.

This behavior will be subsequently illustrated by a series of experiments.

No.	hashtag	count
0	90dayfiance	316
1	tejran	345
2	ukraine	352
3	tejasswiprakash	372
4	nowplaying	439
5	anjisalvacion	732
6	puredoctrinesofchrist	831
7	1	1105
8	lolinginlove	1258
9	bbnaija	1405

TABLE I
TWT.10 DATA SET - HASHTAGS AND CARDINALITIES OF THE SET OF RELATED TWEETS USED IN THE EXPERIMENTS

III. DATA

We used tweets provided by Twitter (a random sample of about 1% of English tweets) collected for the time period from mid September 2019 till end of May 2022 From this set in the experiments, we use the set TWT.10, being a collection of tweets related to hashtags listed in table I, collected from the Twitter system. The tweets had to have one single hashtag (which we treated as an indication of being devoted to a single theme).

IV. METHODS

We study two standard versions of Graph Spectral Clustering, available from scikit-learn, and the 6 versions of spherical k-means and 6 versions of our proprietary so-called K-embedding based clustering algorithm.

More precisely the clustering experiments were performed with popular Python libraries: numpy [6], scipy [17], scikit-learn [2] and soyclustering [8] which is an implementation of spherical k-means [9]. In particular, we used

1) `SpectralClustering` class from scikit-learn with two distinct settings of the `affinity` parameter: `precomputed` (affinity from similarity matrix) and `nearest_neighbors` (affinity from graph of nearest neighbors) - as a representative of the spectral clustering, and
2) `SphericalKMeans` class from soyclustering with the following combinations of (`init`, `sparsity`) parameter pairs (the mentioned 6 versions) (short names given for reference): "sc.n": ('similar_cut', None), "sc.sc": ('similar_cut', 'sculley'), "sc.md": ('similar_cut', 'minimum_df'), "k++.n": ('k-means++', None), "k++.sc": ('k-means++', 'sculley'), "k++.md": ('k-means++', 'minimum_df'), and
3) K-embedding clustering (our implementation, exploiting spherical k-means – see subsection IV-C).Same combinations of parameter pairs (versions) were used as for `SphericalKMeans` above. The following numbers of eigenvectors were tried: $r = 12+$.

The advantages and disadvantages of these methods are briefly discussed below.

A. Spectral analysis

In fact spectral clustering algorithms constitute a large family, see e.g. [18], [13], [21], which have numerous desirable properties (like detection of clusters with various shapes, applicability to high dimensional datasets, capability to handle categorical variables), yet they suffer from various shortcomings, common to other sets of algorithms, including multiple possibilities of representation of the same dataset, producing results in a space different from the space of original problem, curse of dimensionality, etc. These shortcomings are particularly grieving under large and sparse data set scenario, like in Twitter data.

Let us briefly recall the typical spectral clustering algorithm in order to make it understandable, how distant the clustering may be from the applier's comprehension [18]. The first step consists in creating a similarity matrix of objects (in case of documents based on tf, tfidf, in unigram or n-gram versions, or some transformer based embeddings are the options – consult e.g. [14] for details), then mixing them in case of multiple views available. The second step is to calculate a Laplacian matrix. There are at least three variants to use: combinatorial, normalized, and random-walk Laplacian, [18]. But other options are also possible, like: some kernel-based versions, non-backtracking matrix [12], degree-corrected versions of the modularity matrix [1] or the Bethe-Hessian matrix [16]. Then computing eigenvectors and eigenvalues, eigenvector smoothing (to remove noise and/or achieve robustness against outliers) choice of eigenvectors, and finally clustering in the space of selected eigenvectors (via e.g. k-means). The procedure may be more complex, e.g. one may add loops back to preceding steps based on feedback from quality analysis, like degree of deviation from block-structure of the Laplacian.

From this diversified set we chose the two mentioned implementations available from scikit-learn.

B. Spherical k-means

Spherical k-means was developed in [3] by observing that the squared Euclidean distance between two vectors, $\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \|\mathbf{x}_i\|^2 - 2\mathbf{x}_i^T\mathbf{x}_j + \|\mathbf{x}_j\|^2$, in case of normalized vectors reduces to

$$\|\mathbf{x}_i - \mathbf{x}_j\|^2 = 2(1 - \mathbf{x}_i^T\mathbf{x}_j) \ , \qquad (1)$$

and $\mathbf{x}_i^T\mathbf{x}_j = \cos\angle(\mathbf{x}_i, \mathbf{x}_j)$. This makes it very efficient in case of sparse vectors, a typical representation of text documents. Such a variant of k-means suffers dependence on initialization, thus further improvements are proposed, e.g. [4], [7], [11] and [15].

C. K-embedding

K-embedding has the following underlying idea. Let us think for a moment about a particular embedding of the nodes of the graph, based on [10]. Let A be a matrix of the form:

$$A = \mathbf{1}\mathbf{1}^T - I - S \ , \qquad (2)$$

where S stands for an affinity matrix, I is the identity matrix, and $\mathbf{1}$ is the (column) vector consisting of ones, both of appropriate dimensions. (Note that here we have to assume that the diagonal of S consists of zeros). Let K be the matrix of the (double centered) form [5]:

$$K = -\frac{1}{2}(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)A(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)\ , \tag{3}$$

with $n \times n$ being the dimension of S. $\mathbf{1}$ is an eigenvector of K, with the corresponding eigenvalue equal to 0. All the other eigenvectors must be orthogonal to it as K is real and symmetric, so for any other eigenvector $\mathbf{v}$ of K we have: $\mathbf{1}^T\mathbf{v} = 0$.

Let Λ be the diagonal matrix of eigenvalues of K, and V the matrix where columns are corresponding (unit length) eigenvectors of K. Then $K = V\Lambda V^T$. Let $\mathbf{z}_i = \Lambda^{1/2}V_i^T$, where V_i stands for i-th row of V. Let $\mathbf{z}_i, \mathbf{z}_\ell$ be the embeddings of the nodes i, ℓ, resp. This embedding shall be called K-*embedding*. Then

$$\|\mathbf{z}_i - \mathbf{z}_\ell\|^2 = 1 - S_{i\ell} \tag{4}$$

for $i \neq \ell$. Hence upon performing k-means clustering in this space we *de facto* try to maximize the sum of similarities within a cluster. Note that $K = V\Lambda V^T$ may be quite well approximated if we drop from Λ low eigenvalues and from V their corresoponding eigenvectors (which we do in our experiments).

V. Evaluation

For each of the algorithms we perform the following tests. For each pair of datasets associated with two hashtags from Table I (45 pairs in all) the clustering will be performed by each of the mentioned algorithms 10 times (due to stochastic nature of these algorithms) and the average F-score will be computed. Ten pairs with the highest with highest average F-scores will be taken for the next phase. Now datasets associated with 3 hashtags will be created out of these selected pairs plus each of the hashtags not present in the selected pairs. This process is continued till all 10 hashtags are exhausted. In figures, the average value of F over all computations with the given hashtag cardinality is presented plus the average of the top 10 groups of hashtags.

VI. Results

As visible from the figures 6-19, the increase of the number of intended clusters to be discovered constitutes a problem for the clustering algorithms, with even 9-fold decrease of F-score when going from 2 to 10 clusters. This behaviour is consistent throughout all the investigated methods though minor variations of the shape of the curves may be observed.

Spherical k-means clustering with sc.n configuration appears to perform best for the 10 top pairs of hashtags, followed by K-embedding based clustering with most configurations,

In most cases the top average of the F-score for next higher number of cluster is usually higher than the average score for the entire previous number of clusters, which indicates that better separation of subgroups gives some advantage for the capability to separate the entire group.

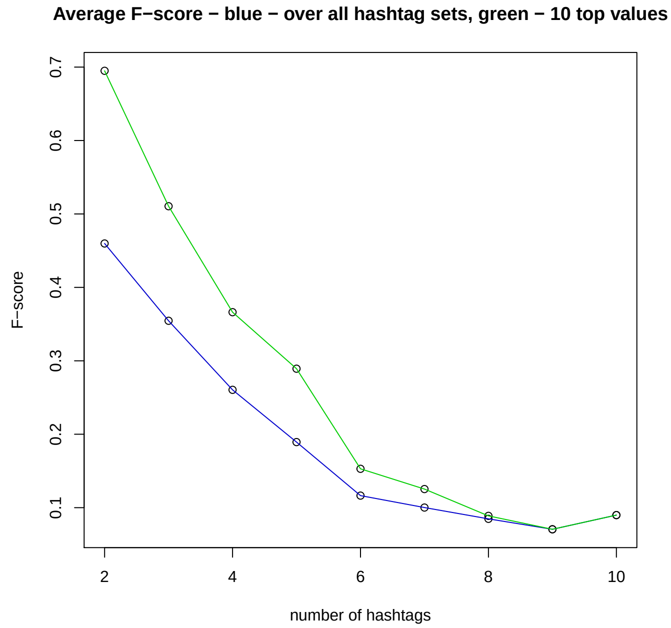

Fig. 6. F-scores for various numbers of hashtags; spectral clustering with affinity nearest_neighbors

A more detailed relationship is presented in Figs 20 and 21 and Table II. Table II shows Spearman and Pearson correlations between the F-score achieved by grouping a dataset related to a given set of hashtags and by grouping datasets obtained by removing data of one of the hashtags. The correlations are generally high and are statistically very significant. This means that clustering capability of subsets of hashtags can be a good indicator of clustering capability for the set of hashtags. But a look at the Fig. 21 convinces us that generqally this capabnility decreases. Fig. 20 ashows additiobnally, that the high correlations are to be expected rather for low values of F-score. Higher F-score values are responsible for higher variation in supergroup F-score.

VII. Conclusions

The performed experiments demonstrate that, in spite of the generally praised properties, graph spectral clustering methods have still a large space for improvements with respect to increasing number of of clusters to be detected. Even if all the subsets of intended clusters may be well separated by the algorithms, their mixture does not so. Same observation can be made about the spherical k-means algorithm.

References

[1] Hafiz Tiomoko Ali and Romain Couillet. Improved spectral community detection in large heterogeneous networks. *Journal of Machine Learning Research*, 18(225):1–49, 2018.

[2] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML*

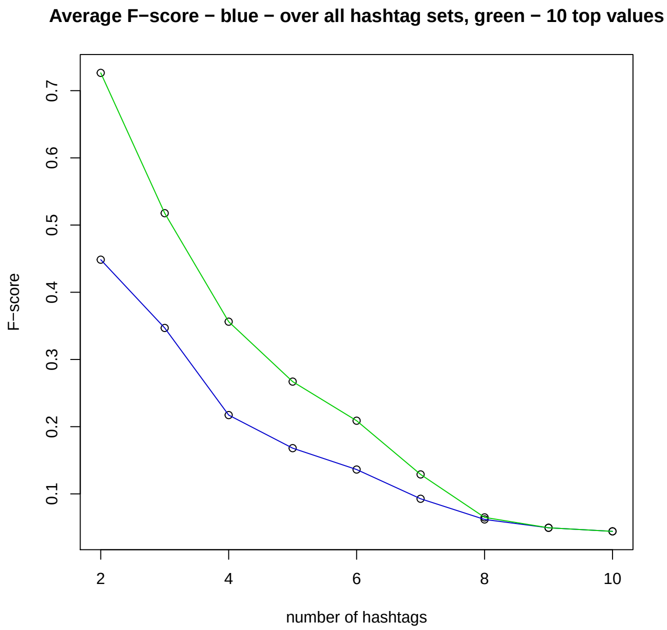

Fig. 7. F-scores for various numbers of hashtags; spectral clustering with affinity precomputed

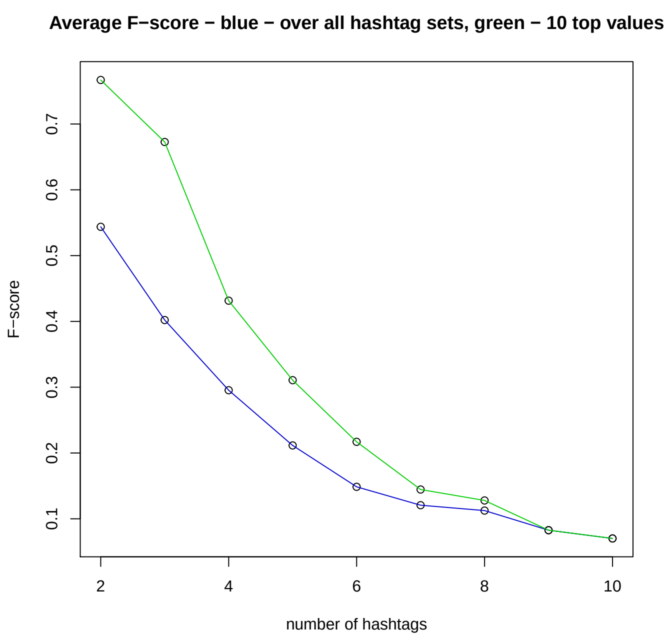

Fig. 8. F-scores for various numbers of hashtags; spherical k-means clustering with sc.md configuration

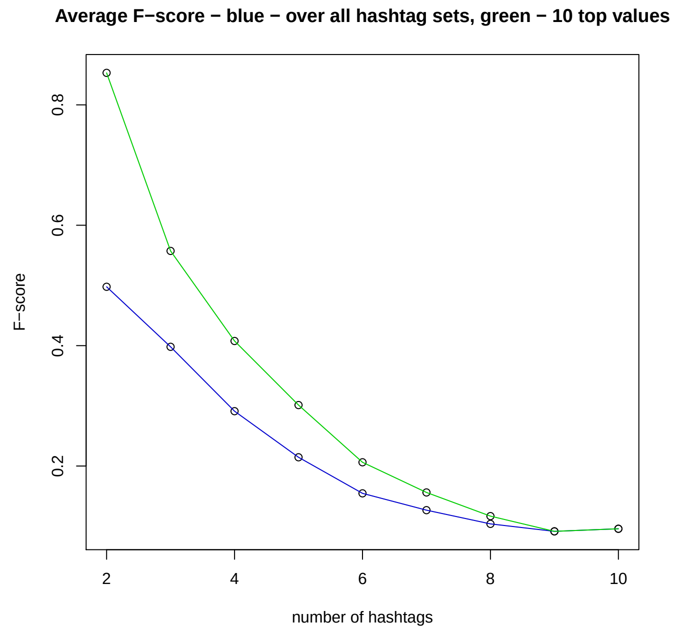

Fig. 9. F-scores for various numbers of hashtags; spherical k-means clustering with sc.sc configuration

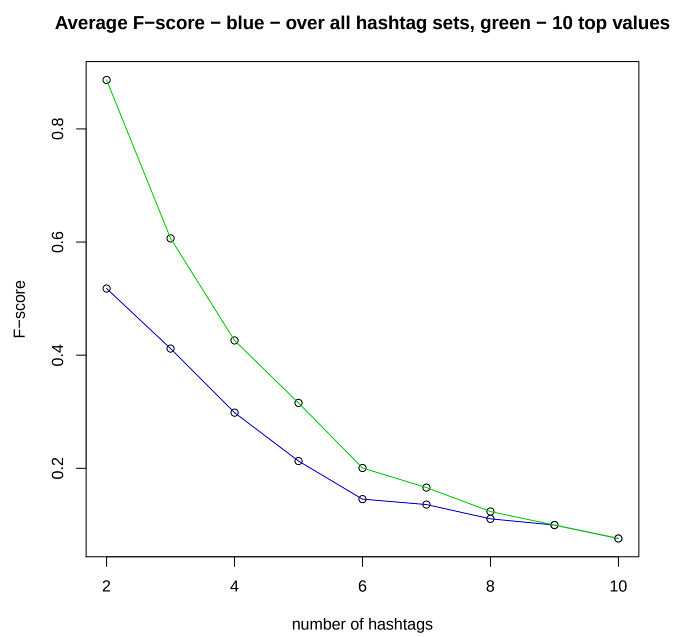

Fig. 10. F-scores for various numbers of hashtags; spherical k-means clustering with sc.n configuration

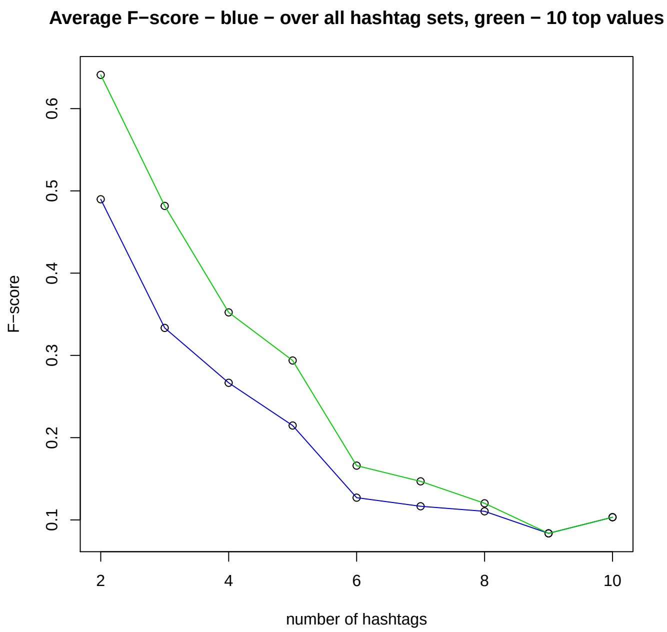

Fig. 11. F-scores for various numbers of hashtags; spherical k-means clustering with k++.md configuration

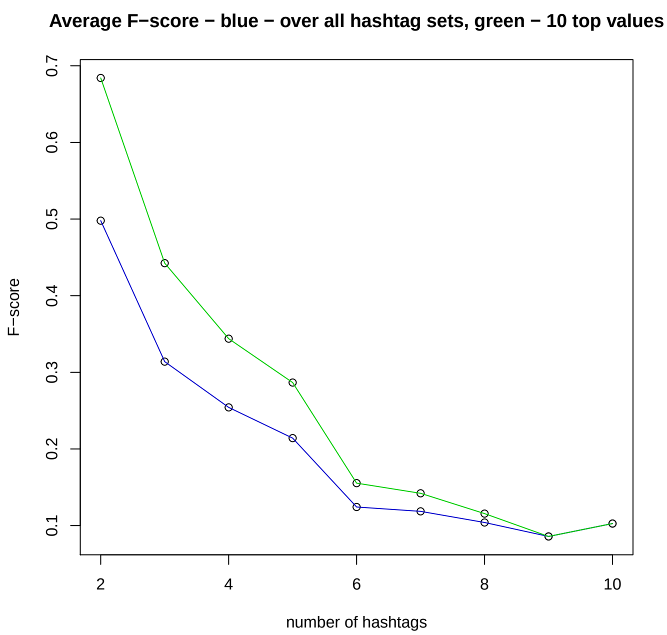

Fig. 12. F-scores for various numbers of hashtags; spherical k-means clustering with k++.n configuration

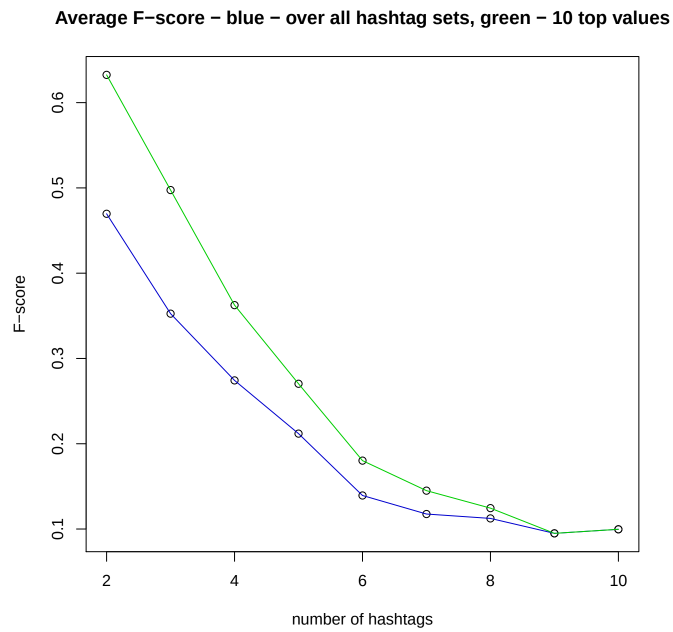

Fig. 13. F-scores for various numbers of hashtags; spherical k-means clustering with k++.sc configuration

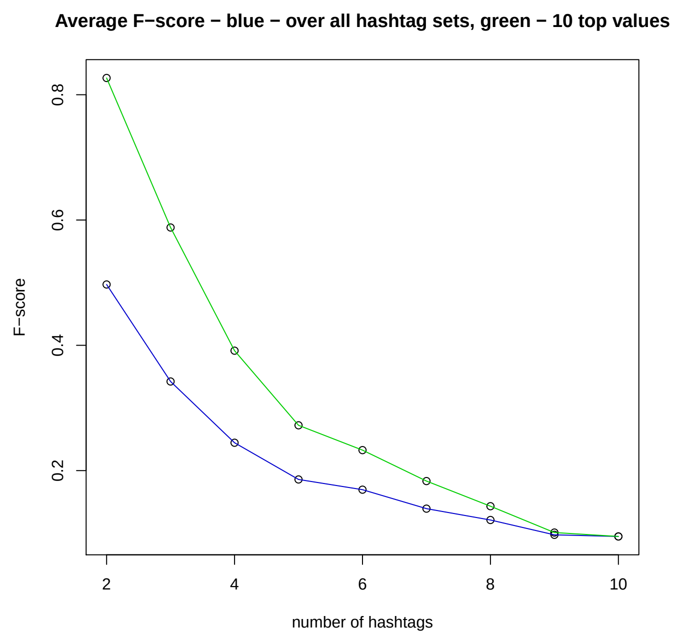

Fig. 14. F-scores for various numbers of hashtags; K-embedding based clustering with sc.md configuration

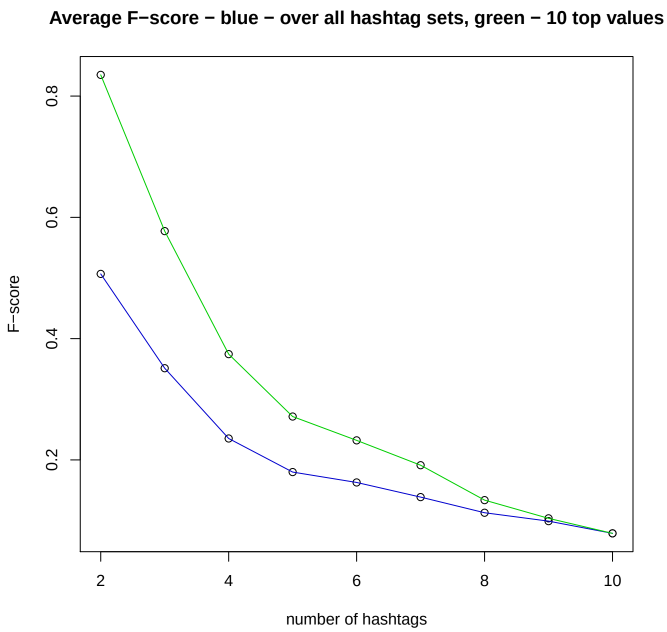

Fig. 15. F-scores for various numbers of hashtags; K-embedding based clustering with sc.sc configuration

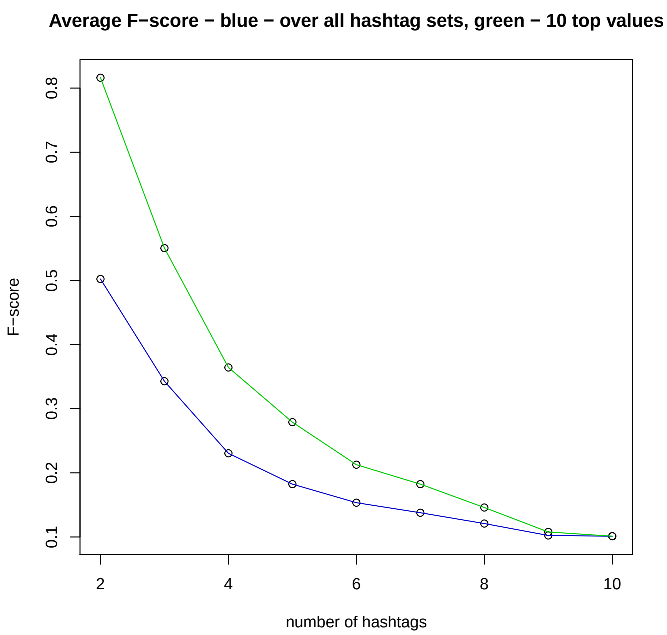

Fig. 16. F-scores for various numbers of hashtags; K-embedding based clustering with sc.n configuration

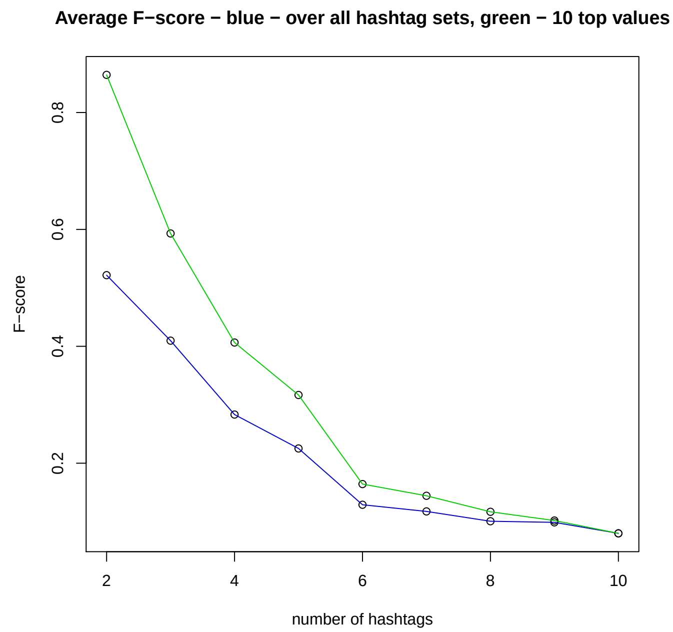

Fig. 17. F-scores for various numbers of hashtags; K-embedding based clustering with k++.md configuration

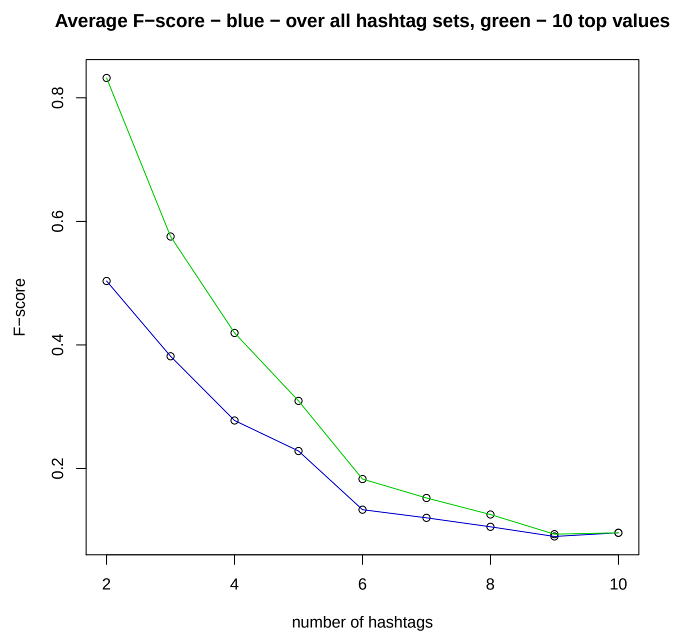

Fig. 18. F-scores for various numbers of hashtags; K-embedding based clustering with k++.n configuration

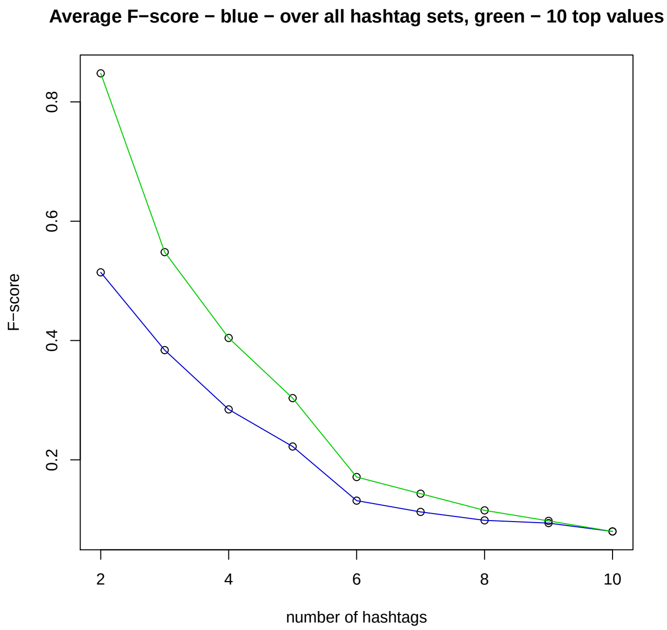

Fig. 19. F-scores for various numbers of hashtags; K-embedding based clustering with k++.sc configuration

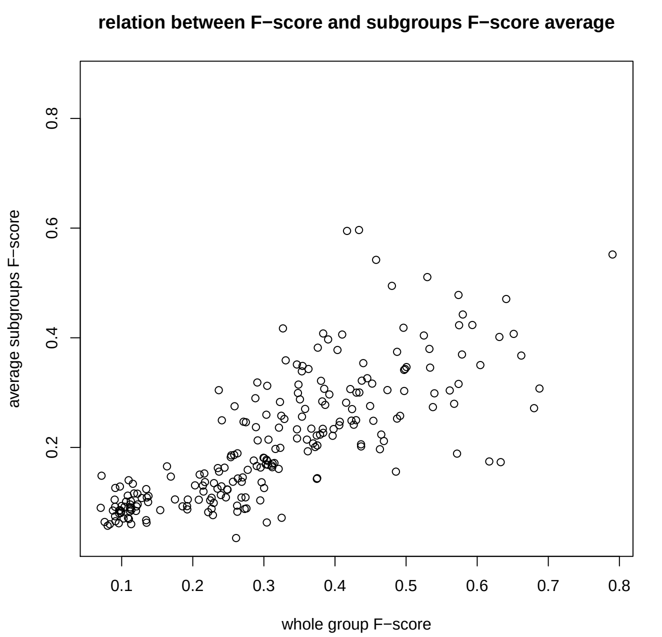

Fig. 20. Relationship between F-score of the given group that was clusters and the average F-score of its subgroups (with one less hashtag); spectral clustering with affinity nearest_neighbors

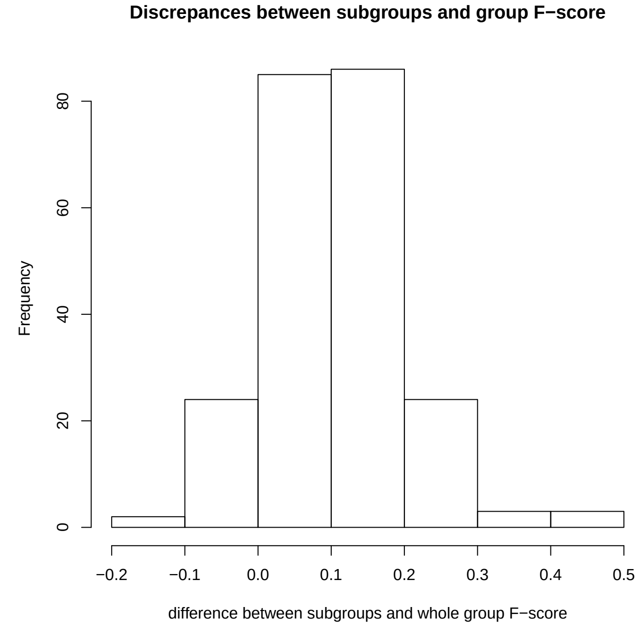

Fig. 21. Difference (negated) between F-score of the given group that was clustered and the average F-score of its subgroups (with one less hashtag); spectral clustering with affinity nearest_neighbors

algprithm	pearson	p.value	spearman	p.value
spectral nearest_neighbors	0.7745	0	0.8358	0
spectral precomputed	0.7374	0	0.7437	0
spherical sc.md	0.7036	0	0.7711	0
spherical sc.sc	0.8306	0	0.8538	0
5 spherical k++.n	0.7647	0	0.8189	0
6 spherical sc.n	0.7778	0	0.8167	0
7 spherical k++.md	0.7796	0	0.8129	0
8 spherical k++.sc	0.8099	0	0.8502	0
9 K-embedding.12plus sc.md	0.6057	0	0.6041	0
10 K-embedding.12plus sc.sc	0.6948	0	0.6678	0
11 K-embedding.12plus k++.n	0.7975	0	0.8294	0
12 K-embedding.12plus sc.n	0.6901	0	0.7113	0
13 K-embedding.12plus k++.md	0.7976	0	0.8483	0
14 K-embedding.12plus k++.sc	0.7924	0	0.8460	0

TABLE II
CORRELATION BETWEEN THE F-SCORE OF A GIVEN GROUP OF HASHTAGS AND THEIR SUBGROUPS OF CARDINALITY LOWER BY ONE.

PKDD Workshop: Languages for Data Mining and Machine Learning, pages 108–122, 2013. https://scikit-learn.org.

[3] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, Jan 2001.

[4] Yasunori Endo and Sadaaki Miyamoto. Spherical k-means++ clustering. In Vicenc Torra and Torra Narukawa, editors, *Modeling Decisions for Artificial Intelligence*, pages 103–114, Cham, 2015. Springer International Publishing.

[5] J.C. Gower. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, 53(3-4):325—-338, 1966.

[6] Charles R. Harris, K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020. https://numpy.org.

[7] Sai Ji, Dachuan Xu, Longkun Guo, Min Li, and Dongmei Zhang. The seeding algorithm for spherical k-means clustering with penalties. *J. Comb. Optim.*, 44(3):1977–1994, oct 2022.

[8] Hyunjoong Kim and Han Kyul Kim. clustering4docs github repository, 2020. https://pypi.org/project/soyclustering/.

[9] Hyunjoong Kim, Han Kyul Kim, and Sungzoon Cho. Improving spherical k-means for document clustering: Fast initialization, sparse centroid projection, and efficient cluster labeling. *Expert Systems with Applications*, 150:113288, 2020.

[10] R. Kłopotek, M. Kłopotek, and S Wierzchoń. A feasible k-means kernel trick under non-euclidean feature space. *International Journal of Applied Mathematics and Computer Science*, 30(4):703–715, 2020. Online publication date: 1-Dec-2020.

[11] Johannes Knittel, Steffen Koch, and Thomas Ertl. Efficient sparse spherical k-means for document clustering. In *Proceedings of the 21st ACM Symposium on Document Engineering, DocEng '21*, pages 1–4. ACM, New York, NY, United States, 2021. https://doi.org/10.1145/3469096.3474937.

[12] F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborov, and P. Zhang. Spectral redemption in clustering sparse networks. In *Proc. the National Academy of Sciences*, volume 110[50], pages 20935–20940, 2013.

[13] Peter Macgregor and He Sun. A tighter analysis of spectral clustering, and beyond. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14717–14742. PMLR, 17–23 Jul 2022.

[14] Ch.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008. https://doi.org/10.1017/CBO9780511809071.

[15] Rameshwar Pratap, Anup Deshmukh, Pratheeksha Nair, and Tarun Dutt. A faster sampling algorithm for spherical k-means. In Jun Zhu and Ichiro Takeuchi, editors, *Proceedings of The 10th Asian Conference on Machine Learning*, volume 95 of *Proceedings of Machine Learning Research*, pages 343–358. PMLR, 14–16 Nov 2018.

[16] Alaa Saade, Florent Krzakala, and Lenka Zdeborová. Spectral clustering of graphs with the bethe hessian, 2014.

[17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. https://scipy.org.

[18] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[19] Slawomir T. Wierzchon and Mieczyslaw A. Klopotek. Spectral cluster maps versus spectral clustering. In *Computer Information Systems and Industrial Management*, volume 12133 of *LNCS*, pages 472–484. Springer, 2020.

[20] S.T. Wierzchoń and M.A.Kłopotek. *Modern Clustering Algorithms*, volume 34 of *Studies in Big Data*. Springer Verlag, 2018.

[21] Yang Xu, Arun Srinivasan, and Lingzhou Xue. *A Selective Overview of Recent Advances in Spectral Clustering and Their Applications*, pages 247–277. Springer International Publishing, Cham, 2021.