Kernel based methods for text clustering

Maria Gkotsopoulou & Ricard Monge
Universitat Politècnica de Catalunya-Barcelona Tech

Abstract

We explore the use of kernel methods, *kernel k-means* and *spectral clustering*, for clustering on a set of text documents. For this purpose, we study both methods, research the different sets of algorithms and finally implement them in **R**. We design a set of experiments to evaluate their performance and use the classical *k-means* as benchmark. Finally, we present our findings and results and discuss possible future lines of work.

Introduction

K-means algorithm is a very popular clustering algorithm due to its simplicity and effectiveness on linearly separable numeric data, although it lacks in flexibility with more complex data.

A way to extend the concepts of typical k-means algorithm is by using kernel functions. With them, we can project the data into a higher dimensional feature space where it is linearly separable. In addition, we can apply the algorithm directly to non-numeric data without the need of numeric features extraction.

In this paper, we present two extensions of the traditional algorithm using kernel methods: kernel k-means, version of the usual algorithm with the projected data points (using kernel functions); and spectral clustering, using the graph properties of the kernel matrix (kernel function applied to all pairs of points) to perform regular k-means.

Moreover, we apply these extensions to the *Reuters* document collection and compare it with the regular version applied to the text extracted features.

Previous work

The study and comparison of the regular and kernelized versions of K-means together with the Spectral Clustering algorithm, using k-means clustering, on the *Reuters* document collection has been previously done by (Karatzoglou and Feinerer 2006).

In particular, the authors took from the Reuter dataset 580 documents about *crude* topic, 280 about *corn* topic and 1100 about *acq* (acquisitions) topic. In our case we decide to use different topics (more details in the *experiment* section). With this data, the aim of the authors is that the clustering algorithms find the same kind of clusters by identifying semantic relationships.

For this purpose, they used kernelized k-means and spectral clustering algorithms on the text samples by means of a simple string kernel, as the one explained in the following section. They compared these methods with regular k-means algorithm using a document embedding to numeric vectors with the inverse frequencies of each term as values for the data points' features. In comparison, we use a different embedding of the document texts as numeric vectors, by computing the *term frequency · inverse document frequency* values as features.

To evaluate the performance of their clusters and perform hyper-parameter tuning, the authors used the mean of the recall of each class, called *recall rate* with the following expression:

$$R = \frac{\sum_{\gamma=1}^{k} n_{\gamma\Gamma}}{\sum_{\Gamma=1}^{k} N_{\Gamma}}$$

where γ are the found clusters, Γ the reference groups, $n_{\gamma\Gamma}$ the number of documents of reference cluster Γ assigned to γ and N_{Γ} the total number of documents on reference cluster Γ. In our case, we use the metric *variation of information* (Meilă 2007) as the cluster comparison metric between the found and reference clusters.

Theory

The k-means clustering algorithm is one of the most commonly used clustering methods providing solid results but also having some drawbacks. Having a sample $\{x_1, ..., x_n\}$ and k clusters π_j for $j = 1, ..., k$, we assign a point x_i to a cluster iff:

$$k = argmin_j\{d(x_i, c_j)\} = argmin_j\{\|x_i - c_j\|^2\}$$

$$where\ c_j = \frac{1}{|\pi_j|} \sum_{x \in \pi_j} x$$

As such, this algorithm minimizes the within clusters sum of squares:

$$WSS = \sum_{j=1}^{k} \sum_{x \in \pi_j} \|x - c_j\|^2$$

A major drawback of k-means is that it cannot separate clusters that are not linearly separable in input space. Both kernel k-means and spectral clustering address this problem by projecting the data points into a features space in which it is separable.

Kernel K-means

Our aim is to use the kernel trick to project the data points in the input space onto a higher dimensional feature space in which the points are linearly separable. What's more, we can apply kernels to non-numeric types of data and use the k-means algorithm to the projected versions directly.

To kernelize the method, we look at the expression of the distance between data points and centers of the clusters in feature space. We denote a scalar product with $< \cdot ; \cdot >$ and the feature map as ϕ. It is known that it can be expressed as:

$$d(x,y) = \|x-y\|^2 =< x-y \ ; \ x-y >= ... =$$
$$= k(x,x) + k(y,y) - 2k(x,y)$$

where $k(x,y) =< \phi(x), \phi(y) >$ is the kernel function, inner product of points in the feature space. Next, we express the distance of a point x_i to the center c_j of the cluster π_j:

$$\|\phi(x_i) - \phi(c_j)\|^2 =< \phi(x_i) - \phi(c_j) \ ; \ \phi(x_i) - \phi(c_j) >= ... =$$
$$= k(x_i,x_i) + f(x_i,c_j) + g(c_j)$$

where the functions f represent a sort of point-cluster kernel distance

$$f(x_i,c_j) = \frac{-2}{|\pi_j|} \sum_{l=1}^{n} z(x_l,\pi_j) k(x_i,x_l)$$

with $z(x_l,c_j)$ indicator function which is 1 if x_l belongs to π_j and 0 otherwise. On the other hand, g represent a sort of within-cluster kernel distance:

$$g(c_j) = \frac{1}{|\pi_j|^2} \sum_{l=1}^{n} \sum_{m=1}^{n} z(x_l,\pi_j) z(x_m,\pi_j) k(x_l,x_m)$$

From the expression of $\|\phi(x_i) - \phi(c_j)\|^2$ we see that $k(x_i,x_i)$ does not change with the cluster considered. Furthermore we see g is just cluster dependent and can be precomputed at each iteration of the algorithm.

With these considerations, in the kernelized k-means algorithm we will assign a point x_i to a cluster π_k iff:

$$k = argmin_j \{ f(x_i,c_j) + g(c_j) \}$$

We define an indicator matrix Z with as many rows as data points and as many columns as clusters with elements $z_{ij} = z(x_i,\pi_j)$ given by the previous indicator function. Moreover, we define the kernel matrix K that contains elements $k_{ij} = k(x_i,x_j)$ given by the kernel function of each pair of points. Finally, we define diagonal matrix L which contains in the diagonal the elements $l_{jj} = 1/|\pi_j|$ with the inverse size of the clusters at any given point. With these definitions, the previous functions f can be easily computed for all points in a matrix F by:

$$F = K \cdot Z \cdot -2L$$

in addition, if we restrict the Kernel matrix to the points of a given cluster π_j in a given time in a matrix G_j, the functions g can be easily computed for a cluster π_j as:

$$g_j = \left(\sum G_j \right) \cdot l_{jj}^2$$

where $\sum G_j$ denotes the sum of all elements of G_j.

In practice, for each iteration of the algorithm we compute these measures and find the assignments which minimize the sum $f+g$ for each point and cluster. We stop iterating when the maximum number of iterations is reached or when there are now reassignments.

Spectral clustering

Given the previous data sample $\{x_1, ..., x_n\}$ and a similarity matrix S with elements s_{ij} that measure similarity between points x_i and x_j, one can view the dataset as a graph (V,S) with each point as a node in V and each edge as the directed connection from x_i to x_j with the weight given by s_{ij}. If two points have similarity $s_{ij} = 0$ they won't be connected. We assume the similarity matrices in the following discussion are symmetric and thus the graph is undirected.

We define the degree of a node (point) x_i of this graph as the sum of it's outgoing edge's weights, given by:

$$d_i = \sum_{j=1}^{n} s_{ij}$$

and build the diagonal matrix D with the degrees of each node at the diagonal, and 0 elsewhere.

Given the undirected graph associated with the previous data point sample and its similarity matrix S, we define the following Laplacian matrices:

- Unnormalized Laplacian matrix $L = D - S$.
- Normalized Symmetric Laplacian matrix $L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}SD^{-1/2}$.
- Normalized Random Walk Laplacian matrix $L_{rw} = D^{-1}L = I - D^{-1}S$.

These matrices give insights on the properties of the graph.

The problem of clustering the data points $\{x_1, ..., x_n\}$ into k clusters with an associated symmetric similarity matrix S can be viewed as a graph partition problem of the associated graph (V, S) into k groups such that the similarities between points of different groups are small and between points of the same group are big.

This problem is also called *normalized cut problem* and, as seen in (Dhillon, Guan, and Kulis 2004), is equivalent to solving the following minimization problem:

$$minimize\ \frac{1}{k}trace(Z^T SZ)$$

where $Z = X(X^T DX)^{-1/2}$ and X is an indicator matrix with as many rows as data points and columns as groups, with elements $x_{ij} = 1$ iff point i is in group j. By letting $\tilde{Z} = D^{1/2}Z$ and relaxing the constraint that X is and indicator matrix, just imposing that $\tilde{Z}^T\tilde{Z} = Id_k$, we obtain that the solution to the equivalent problem can be obtained by taking the matrix $\tilde{Z}$ to be the first (smallest) k eigenvectors of the Laplacian matrix L (Luxburg 2007). The assignments to the groups can then be derived by discretizing or clustering this eigenvectors themselves.

According to (Luxburg 2007), the normalized Laplacians L_{sym} and L_{rw}, are preferred over the unormalized. In addition, the L_{sym} might lead to undesired artifacts so, the L_{sym} should be preferred. We will use the normalized Laplacian L_{rw} in our experiments (although our function offers the choice of all the explained Laplacians).

In practice, the algorithm for spectral clustering goes as follows. First, we take the data points and build the similarity matrix S. From this matrix we compute one of the Laplacian matrices L and take the first k eigenvectors. Finally, we use regular k-means algorithm to cluster these eigenvectors into k groups.

It is important to know that we can use as similarity matrix the kernel matrix K, which is symmetric, so we can apply this algorithm with the same representation as in the Kernel K-means algorithm.

String kernel

In both considered algorithms we input the kernel matrix of the data and obtain the clustering. In order to be able to apply them to non-numeric samples, in particular text, and project them in a space where they are linearly separable; we use a simple string kernel. We define our string kernel on two sequences of characters x and y as:

$$k(x, y) = \sum_{r \in x,y} num_r(x) \cdot num_r(y)$$

Intuitively, this kernel is the inner product between two feature vectors in the projected space where each component is determined by the number of matches of a substring r on the sequence. This kernel is implemented by the *stringot* function in the *kernlab* package. In particular, one can specify whether the components in the feature space correspond to the substrings of an exact length l, hyper-parameter, known as *spectrum* string kernel; or to the substrings of length less or equal to l, known as *boundrange* string kernel. As the previous work in (Karatzoglou and Feinerer 2006) uses the latter we also consider this version.

Experiments

We compared the performance of the above mentioned clustering techniques on text data by running a series of experiments on the Reuters text data set. The Reuters-21578 dataset (Lewis 1997) contains stories for the Reuters news agency. A Reuters category can contain from 1 up to 2877 documents. To run our *kernel k-means* and *spectral clustering* methods, we used a preprocessed version of the original dataset made available by (Cardoso-Cachopo 2007). The following steps were applied to obtain it:

- Substitute TAB, NEWLINE and RETURN characters by SPACE.
- Keep only letters and turn them to lowercase.
- Substitute multiple SPACES by a single SPACE.
- The title/subject of each document is simply added in the beginning of the document's text.
- Remove words that are less than 3 characters long.
- Remove the 524 SMART stopwords.
- Apply Porter's Stemmer.

Due to limited computational resources we used a subset of the data so that the computation of a full kernel matrix in memory was not a concern. We used the *crude* category, which includes 374 documents, the *interest* category, which includes 271 documents, and *coffee* category, which includes 112 documents. Our dataset thus consists of 757 documents. Furthermore, for the classical *k-means* method we create a *term document matrix* using the *term frequency–inverse document frequency*, so instead of the frequencies of the term as entries we are ultimately measuring relative importance of a word to a document. In addition, to reduce model complexity, we remove any tokens which are missing from more than 95% of the documents in the corpus.

For the *kernel k-means* and *spectral clustering* methods we use string kernels implemented in *kernlab* to create the *kernel matrix*. The *boundrange* string kernel has a hyper-parameter that needs to be tuned which is the length of substrings considered. We use the values $n = \{2, 5, 7, 10, 12, 15, 20, 25\}$ and normalize in order to remove any bias introduced by document length. In order to study the effect of the hyper-parameter on the clustering result, we run the *k-means* step of the *spectral clustering* method, as well as, the center assignment step of the *kernel k-means* method 10 times. Similarly to (Poon et al. 2012), we evaluated the quality of an obtained partition by comparing it with the true partition using the variation of information (*VI*) (Meilă 2007). Note that the lower the *VI* indicates better performance. We thus simulate to a certain degree a cross-validation setup.

Moreover, we need to associate the k clusters obtained by our methods to the labels of the dataset. To this extent, we assume there is a one-to-one relation between the 3 clusters and the topics and thus define the linking of the clusters to the topics based on the permutation that minimized the global *error rate*.

Results

The main goal of the experiments is to establish if kernel methods along with kernel strings are a viable solution for grouping a set of text documents. From the experiments it became obvious that length parameter influences the result.

Since the *R k-means* method does not depend on the string kernel it's *variation of information* distance is constant across all values. Interestingly, we see a downward trend as the length grows and then a stabilizing trend. For the case of *kernel k-means* the minimum occurs for a lower value string length value than for the case of *spectral clustering*. Moreover, the classical *k-means* outperforms *spectral clustering*, where as *kernel k-means* seems to be at par at its minimum.

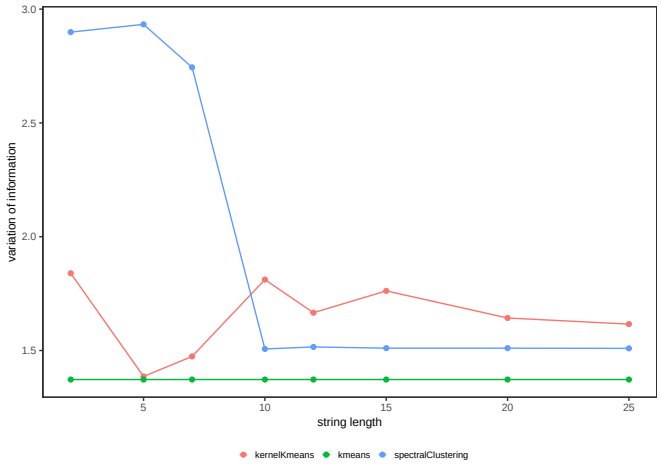

Figure 1: Length Hyperparameter effect across clustering methods

In continuation, we run our methods using as a length parameter the one that minimized the *VI* and calculated the following performance measures to compare their performance.

Table 1: Performance Metrics

	kernel kmeans	spectral clustering	kmeans
vi	1.369	1.523	1.372
error.rate	0.313	0.498	0.312
accuracy	0.687	0.502	0.688
sensitivity	0.539	0.341	0.541
precision	0.549	0.500	0.549
recall	0.539	0.341	0.541

Note that *VI*, *error rate* and *accuracy* are global measures while *sensitivity*, *precision* and *recall* are mean values of the respective measures per cluster. Overall, *kernel k-means* and *k-means* display a similar performance considering all measures. On the other hand, both aforementioned methods outperform *spectral clustering* and thus are in line with our findings from the tuning step.

Conclusions & future work

The experimental results lead us to two conclusions. First, given the parity of performance of *kernel k-means* and classical *k-means* we conclude that both the string

kernel, as well as, the document-term matrix capture the same amount of semantic information from the text. On the other hand, the low performance of the *spectral clustering* method signals that the string kernel used combined with a graph partitioning method does not produce good results.

Moreover, we should remark that our results are worse compared to (Karatzoglou and Feinerer 2006). However, we are unable to make a direct comparison since our choice of topics differs. In addition, we could hypothesize that there is a higher semantic overlap in our topics subset and this leads to a lower performance. Needless to say, we consider it necessary to run further experiments before concluding that *kernel k-means* outperforms *spectral clustering*.

In terms of future work, we could focus on investigating different string kernels that better capture semantic relations between text. So, perhaps we should use a different kernel matrix for *spectral clustering* from the one used to apply the *kernel k-means* method.

References

"A Large Scale Clustering Scheme for Kernel K-Means." 2002. In *Proceedings of the 16 Th International Conference on Pattern Recognition (Icpr'02) Volume 4 - Volume 4*, 40289. ICPR '02. Washington, DC, USA: IEEE Computer Society. http://dl.acm.org/citation.cfm?id=846227.848565.

Cardoso-Cachopo, Ana. 2007. "Improving Methods for Single-label Text Categorization." PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa.

Dhillon, Inderjit S., Yuqiang Guan, and Brian Kulis. 2004. "Kernel K-Means: Spectral Clustering and Normalized Cuts." In *Proceedings of the Tenth Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 551–56. KDD '04. New York, NY, USA: ACM. https://doi.org/10.1145/1014052.1014118.

Karatzoglou, Alexandros, and Ingo Feinerer. 2006. "Text Clustering with String Kernels in R." In *GfKl*.

Lewis, David D. 1997. "Reuters-21578 Text Categorization Test Collection, Distribution 1.0." In.

Luxburg, Ulrike. 2007. "A Tutorial on Spectral Clustering." *Statistics and Computing* 17 (4): 395–416. https://doi.org/10.1007/s11222-007-9033-z.

Meilă, Marina. 2007. "Comparing Clusterings—an Information Based Distance." *J. Multivar. Anal.* 98 (5): 873–95. https://doi.org/10.1016/j.jmva.2006.11.013.

Ng, Andrew Y., Michael I. Jordan, and Yair Weiss. 2001. "On Spectral Clustering: Analysis and an Algorithm." In *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, 849–56. NIPS'01. Cambridge, MA, USA: MIT Press. http://dl.acm.org/citation.cfm?id=2980539.2980649.

Poon, Leonard K.M., April H. Liu, Tengfei Liu, and Nevin L. Zhang. 2012. "A Model-Based Approach to Rounding in Spectral Clustering." In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, 685–94. UAI'12. Arlington, Virginia, United States: AUAI Press. http://dl.acm.org/citation.cfm?id=3020652.3020724.

Shi, Jianbo, and Jitendra Malik. 2000. "Normalized Cuts and Image Segmentation." *IEEE Trans. Pattern Anal. Mach. Intell.* 22 (8): 888–905. https://doi.org/10.1109/34.868688.

Welling, Max. 2013. "Kernel K-Means and Spectral Clustering." *2013-03-15]. Http://Www, Ics. Uci. Edu/-Welling/Teaching/273 ASpring09/SpectralClustering. Pdf.*