Spectral Clustering Implementation and Appliacation

Qi Wang Hanqiu Xia

April 21, 2016

Abstract

Spectral clustering is widely used in image segmentation. The main idea of spectral clustering is to use the spectrum (eigenvalues) of the similarity matrix of the data to perform dimensionality reduction and then perform clustering in fewer dimensions. In this project, we investigated the normalized spectral decomposition algorithm from the paper "On Spectral Clustering: Analysis and an algorithm" by A. Ng, M. Jordan, and Y. Weiss [1]. Their algorithm improved upon the existing spectral clustering algorithm by resolving the issues of inconsistent algorithms of using eigenvectors as well as providing evidence of it resulting in a reasonable clustering. We implemented the algorithm using Python with considering both common and edge cases. We also optimized the Python codes by applying vectorization, Cython, and Just-In-Time compiling. Finally, we tested the algorithm on simulated and real datasets and compare the experimental results with those of k-means clustering to see if spectral clustering dramatically improves the results.

1 Background

Spectral clustering is inspired by the the idea of spectral graph partitioning, in which we use the first two eigenvectors to partition the graph into exactly two parts. According to Ng et al [1]., the basic idea of spectral clustering is utilising the k eigenvectors simultaneously to cluster points into k subsets. The detailed algorithm is as follows:

Suppose we have a set of n points $S = \{s_1, \ldots, s_n\}$ in $\mathbb{R}^m$, and we want to cluster them into k groups.

Step 1: Construct the affinity matrix $A \in \mathbb{R}^{n \times n}$, each element in A is defined as $A_{ij} = exp(-||s_i - s_j||^2/2\sigma^2)$, for $i, j = 1, \ldots, n$. We will introduce a method of choosing σ in later pages.

Step 2: Define D to be the diagonal matrix with $D_{ii} = \sum_{j=1}^{n} A_{ij}$, and form the normalized Laplacian matrix $L = D^{-1/2} A D^{-1/2}$.

Step 3: Capture the fist k largest eigenvectors of $L, e_1, e_2, \ldots, e_k, e_i \in \mathbb{R}^n$ and form the matrix $E = [e_1\ e_2\ \cdots\ e_k] \in \mathbb{R}^{n \times k}$.

Step 4: Create the new normalized matrix $U \in \mathbb{R}^{n \times k}$ from E, defined as $U_{ij} = E_{ij}/(\sum_{j=1}^{n} E_{ij}^2)^{1/2}$.

Step 5: Consider U to be a set of n points that need to be clustered now, apply K-means or any other algorithm that can minimize distortion to cluster U.

Step 6: Assign the original point s_i in to cluster j if and only if the ith-row of U was distributed to cluster j in previous step.

2 Implementation

2.1 Ideal Case

2.2 General Case

3 Testing

3.1 Unit Test for Common Case

3.2 Unit Test for Edge Case

4 Optimization

4.1 Vectorization

4.2 Cython

4.3 JIT (Just-In-Time compiling)

4.4 Improvement result

5 Application and Comparison

5.1 Application on Simulated Data

5.2 Application on Real Data

5.3 Comparison of Sepctral clustering and K-means

6 Conclusion

References

[1] A. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. *Advances in Neural Information Processing*. Vol. 14, No. 2. (2001), pp. 849-856