COMP41450 Machine Learning

Assignment: Non-negative Matrix Factorization

Assignment Guidelines

- This is an individual assignment. Collaboration in groups is not permitted.
- Submissions should be made via the COMP41450 Moodle page for this assignment by Friday November 28th 2014.
- A submission should contain (1) code in a ZIP file, (2) a report with an account of the tasks below. The recommended report length is 4-5 pages and should be in PDF format.
- Recommended programming languages for the assignment include Java, Python and C/C++. Implementations should not be written in MATLAB, Octave, or R.
- Code should be commented to explain your implementation.

Assignment Description

The objective of this assignment is to write a new implementation of the Euclidean distance formulation of Non-negative Matrix Factorization (NMF) as proposed by Lee & Seung, which is outlined below (also see references). This implementation should be suitable for application to text data, in the form of a sparse term-document matrix.

1. Randomly initialise $\mathbf{W}$ and $\mathbf{H}$ with positive values.
2. Update factor $\mathbf{H}$ for $1 \leq j \leq n$, $1 \leq c \leq k$:

$$H_{cj} \leftarrow H_{cj} \frac{(\mathbf{W}^{\mathsf{T}}\mathbf{A})_{cj}}{(\mathbf{W}^{\mathsf{T}}\mathbf{W}\mathbf{H})_{cj}}$$

3. Update factor $\mathbf{W}$ for $1 \leq i \leq m$, $1 \leq c \leq k$:

$$W_{ic} \leftarrow W_{ic} \frac{(\mathbf{A}\mathbf{H}^{\mathsf{T}})_{ic}}{(\mathbf{W}\mathbf{H}\mathbf{H}^{\mathsf{T}})_{ic}}$$

4. Repeat from Step 2 until convergence or maximum number of iterations have elapsed.

The tasks for this assignment are as follows:

1. Read a sparse term-document matrix from a file, with terms (words) on rows and documents on columns. A sample matrix *bbcnews.mtx* is provided. The words corresponding to the rows of the matrix are provided in the file *bbcnews.terms*.
2. Apply TF-IDF normalization to the term-document matrix.
3. Randomly initialise factors for NMF.
4. Apply Euclidean NMF as described above to the normalized term-document matrix for a user-specified value of *k* (i.e. the number of clusters).
5. Report the top terms for each cluster.

The process above should be repeated for a number of different values of *k* (e.g. from 2 to 6 clusters).

Your report should explain how you implemented the tasks above, summarise the output that you produced on the sample files (*bbcnews.mtx* and *bbcnews.terms*), and include a discussion on the differences in the results when using different values of *k*.

Sample files and a description of their format is provided on the module Moodle page.

References and Useful Links

Learning the parts of objects by non-negative matrix factorization
Daniel D. Lee & H. Sebastian Seung (1999)
http://www.nature.com/nature/journal/v401/n6755/full/401788a0.html

Algorithms for Non-negative Matrix Factorization
Daniel D. Lee & H. Sebastian Seung (2000)
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.7566

Matrix-toolkits-java
A high-performance library for developing linear algebra applications in Java
https://github.com/fommil/matrix-toolkits-java

Jama
A basic linear algebra package for Java.
http://math.nist.gov/javanumerics/jama/

NumPy
A numerical computing package for Python
http://www.numpy.org/