Stats 141C High Performance Statistical Computing Spring 2017

Homework 2

Lecturer: Cho-Jui Hsieh Date Due: May 11, 13:30pm (in class), 2017

Keywords: *Numerical Linear Algebra, PageRank, Word2vec*

Problem 1. PageRank, Hubs and Authorities [35pt]

We will compute the PageRank and Hubs & Authorities scores for the Wikipedia hyperlink network (`https://snap.stanford.edu/data/enwiki-2013.html`). Download two files from this website and write a program to read "enwiki-2013.txt" into a sparse CSR matrix.

1. Implement the power method for computing the **top singular value** and the corresponding **right singular vector** for a given CSR matrix (don't use eigs or svds). Input of this function is a sparse CSR matrix and number of iterations (an integer). Output of this function is the leading right singular vector (array) and singular value (float). The function will look like:

```
def power_method( sparse_matrix_in_csr_format, number_of_iterations ):
    ...
    ...
    return v, s ## (top eigenvector and eigenvalue)
```

   What is the time complexity for power method (per iteration)?

2. Explain why the quality of your solution can be measured by $\boldsymbol{v}^T A^T A \boldsymbol{v}$, where $\boldsymbol{v}$ is the eigenvector computed by your program and A is the input sparse matrix. What's $\boldsymbol{v}^T A^T A \boldsymbol{v}$ when $\boldsymbol{v}$ is the leading right singular vector of A? What's the range of $\boldsymbol{v}^T A^T A \boldsymbol{v}$ for other $\boldsymbol{v}$?

3. Test your program with number of iterations = 1, 3, 5, 10, 20. For each trial report the run time of power method and the quality of solution (measured by $\boldsymbol{v}^T A \boldsymbol{v} / \|A\|_F$). Also, report the run time and quality of solution when computing $\boldsymbol{v}$ using "svds" in scipy. Which one is better?

4. As we learned in the class, the leading right singular vector corresponds to the "authority" score of a web page. List the names of the top-5 authoritative pages and their scores for this Wikipedia hyperlink network.

5. Describe how to compute the "hub" scores $\boldsymbol{h}$ in $O(\text{nnz}(A))$ time using $\boldsymbol{v}$ (authority score) and A without doing one more SVD. List the names of the top-5 hub pages and their hub scores.

6. Computing the pagerank for the same dataset using the transition matrix

$$0.9P + \frac{0.1}{n} \boldsymbol{e}\boldsymbol{e}^T,$$

   where P is the transition matrix, n is number of nodes, and $\boldsymbol{e} = [1, 1, \cdots, 1]$. List the pages with top-5 agerank scores, and compare them with the hubs&authorities results.

Problem 2. Computing the PPMI matrix [35pt]

In this problem, you will write your own code for learning low-dimensional embeddings of each word in the Kaggle Quora-question-pairs data we used in homework 1.

1. Write a function in python to compute the PPMI matrix given a list of sentences. The PPMI matrix is defined by the following:

   - $\#(w, c)$: number of times two words w, c appear in the same sentence within distance L. Here we set $L = 3$. For example, in the following sentence

     The quick brown fox jumps over the lazy dog.

     We say "quick" and "jump" are within distance 3, but "quick" and "over" are not (they have distance 4).
   - $\#(w)$: number of times a word w appeared in the dataset
   - $|D|$: total number of words in the dataset
   - n: number of distinct words in the dataset
   - The PPMI value of two words w, c is defined by

     $$\text{PPMI}(w, c) = \max(0, \log(\frac{\#(w, c)|D|}{\#(w) \cdot \#(c)}))$$

   - The PPMI matrix is a n-by-n matrix, each element is the PPMI value between two distinct words ($M_{w,c} = \text{PPMI}(w, c)$).

   Since there are too many distinct words in a dataset, the PPMI matrix has to be stored in python sparse matrix format. Please use the CSR format in this homework. The function will look like:

   ```
   def compute_ppmi( list_of_sentences ):
       ...
       ...
       return ppmi_matrix_in_csr_format
   ```

2. Briefly describe your algorithm for forming the PPMI matrix. What is the time complexity of your algorithm?

3. Test your code by inputting all the questions in the Quora-question-pairs data after the pre-processing steps in Problem 1 Homework 1. Report the shape of the PPMI matrix, number of nonzero elements in this matrix, and the Frobenius norm of this matrix.

Problem 3. Word2vec Computation [30pt]

After getting the PPMI matrix, compute the top-k eigenvectors $V_k \in \mathbb{R}^{n \times k}$ and eigenvalues $\Sigma_k \in \mathbb{R}^{k \times k}$ (diagonal matrix). Form the word embedding matrix $F = V_k \Sigma_k$ where each row is a k-dimensional embedding feature vector for a word.

Now we use these features to classify Quora question pairs. For each sentence q, compute the feature vector for the sentence by averaging all the word embeddings features:

$$\text{feature vector for question } q := \boldsymbol{x}_q := \frac{1}{|q|} \sum_{w: w \in q} \boldsymbol{f}_w,$$

where $\boldsymbol{f}_w$ is the word embedding for word w, q is the set of words in the sentence, and $|q|$ is number of words in sentence q. The cosine similarity of two sentences can be computed by

$$\text{cosine similarity between } q_1, q_2 = \frac{\boldsymbol{x}_{q_1}^T \boldsymbol{x}_{q_2}}{\|\boldsymbol{x}_{q_1}\| \|\boldsymbol{x}_{q_2}\|}$$

We can predict the label for a question pairs (q_1, q_2) by

$$\text{sign}(\text{cosine\_similarity}(q_1, q_2) - \text{thr}),$$

where thr is a positive real number for thresholding.

1. Given the PPMI matrix from Problem 2, write a program to compute the accuracy for a given dataset and threshold.
2. Report the accuracy on training.csv with thr $= 0.8, 0.82, \cdots, 1.0$. Which threshold gives you the best result?
3. Use the best threshold, and run the code on validation.csv. What is the validation accuracy?
4. Briefly discuss your findings.