AIML - CS 335

Lab 3: More Linear Algebra

18 August 2022

Lecturer: Abir De

Important: Please read the instructions mentioned in the questions carefully. We have provided boilerplate code for each question. Please ensure that you make changes in the areas marked with TODO. Please read the comments in the code carefully for detailed information regarding input and output format.

1 Discrete Cosine Transform

Discrete Cosine Transform (DCT) is a widely used transformation technique in signal processing and data compression. There are around 8 variations of DCT – depending on the normalization coefficients and boundary conditions.

For this lab, we will use the 2-Dimensional DCT-II. Consider an image X of size $N_1 \times N_2$. The DCT of this image is also a matrix Y of size $N_1 \times N_2$ given by the formula

$$Y_{k_1,k_2} = 4 \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} X_{n_1,n_2} \cos(\frac{\pi}{N_1}(n_1 + \frac{1}{2})k_1) \cos(\frac{\pi}{N_2}(n_2 + \frac{1}{2})k_2)$$

The computation of Y can be succinctly captured by a series of matrix multiplications, resulting in a much faster implementation when coded in python.

Implement the vectorized version of 2-D DCT by completing the function **vectorized_dct**

2 Document Ranking by Query

Information retrieval can be defined as the task of obtaining the most relevant resources according to some requirements from a collection of resources. Document ranking is an important problem in this domain. We are given a collection of documents and a query. We need to order the documents according to their relevance with their query i.e. the document most relevant to the query must be a the top and the document least relevant must be at the bottom.

In this problem, each query is represented as a list of vectors $v_i \in r^w$. The length of each list is m. Each vector in the list can be though of as the representation of a word in the query.

We are also given an collection of k documents where each document is a list of n vectors $a_j \in R^w$. Again, each vector in the list can be though of as the representation of a word in the document.

The document at the i^{th} position has docID = i where $i = 0, 1, 2, 3...k - 1$

Your task is to return a list of docIDs ordered according to relevance to the query.

The relevance R(d,q) between a document d and a query q is calculated as follows -

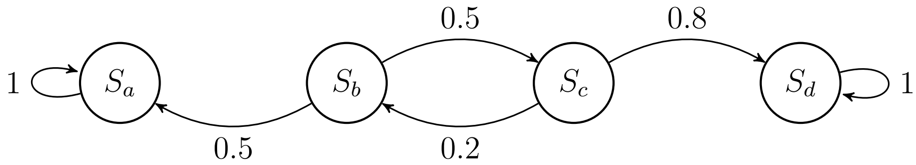
> For a vector v_i in the query, compute $max_{a_j} v_i^T a^j$ over the vectors a_j in the document. Let the value be s_i. This denotes the maximum similarity a word in the query has with a word in the document.
>
> Now, sum all these values of s_i for each vector v_i in the query. This gives the relevance score R(d,q).

Complete the function **get_document_ranking** to implement this.

3 Markov Chains

Markov Chains: It is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event. We can represent it using a digraph, where the nodes represent states & edges represent transition probabilities.

Example of Markov chain:



$$\forall i \sum_{j=1}^{n} M_{ij} = 1$$

In this problem you have to find an optimal way to compute reaching a particular state from a given state using transition probabilites matrix. Assume that each transition takes a unit time.

Given a markov chain matrix M_{nxn} where M_{ij} denotes transition probability from state S_i given S_j = $P(S_j|S_i)$. Find the probability of reaching state S_T at time t = T if we start from state S_a at time t = 0.

- Complete the function **compute_state_probability(M, S_T, T, S_a)** to return computed probability upto **4 decimal places**.

- Complete the function **compute_state_probability_vec(M, S_T, T, S_a)** to return vectorized computed probability upto **4 decimal places**.

Note: Assume n (number of states) is large.

Submission instructions

Complete the functions in **assignment.py**. Keep the file in a folder named **<ROLL_NUMBER>_L3** and compress it to a tar file named **<ROLL_NUMBER>_L3.tar.gz** using the command

```
tar -zcvf <ROLL_NUMBER>_L3.tar.gz <ROLL_NUMBER>_L3
```

Submit the tar file on Moodle.

The directory structure should be -

```
<ROLL_NUMBER>_L3
| - - - - assignment.py
```

Replace ROLL_NUMBER with your own roll number. If your Roll number has alphabets, they should be in "small" letters.