Photo Tag Recommendation System

Multimedia Systems and Applications 4/M Coursework Submission

Magda Kowalska

2037342K

1. *Introduction*

For this assignment I decided to use Java, because I’m the most familiar with I/O methods and data structures (especially Maps) in this language. When running the program, it creates two outputs. First is the coocurrencePhotoTags.csv file containing co-occurrence matrix, second is the standard output of Java Console containing top 5 recommendations with and without IDF for ‘water’, ‘people’ and ‘london’ tags.

* 1. *Tag Recommendation Strategy*  
     The most popular tag recommendation technique gives you tags that co-occur the most with other tags within a given collection. However, this technique tends to recommend unrelated popular tags. The reason for this is because popular tags highly co-exist with almost all tags in a given collection.
     1. *Pseudo-code (for my solution):*

From a collection of photos extract all used tags

For every photo map co-existing tags

* 1. *Tag Suggestion with popularity and significance*

This technique takes into account inverse document frequency (IDF) to get rid of unrelated popular tags. The syntax is the same as for Tag Recommendation Strategy however values of co-occurrence are multiplied by IDF factor. IDF is calculated as follows , where I is the number of images in the collection and I(X) is number of images tagged with tag X. In fact, in my solution I used the same Map from Task 1 to calculate values for Task 3, only updating values with the IDF.

* + 1. *Pseudo-code (for my solution):*

Blab la

Bla

1. *Code description*

My code is divided into 3 sections – Task 1, Task 2 and Task 3.

For the Task 1 getCoocurrence() is the most important one. It reads the photos\_tags.csv file, and create a following data structure Map<String, Map<String, Double>> that contains co-occurrence values of tags. Later this map is printed in a CSV file using coocurenceToFile(…) method.

For the Task 2 findFiveHighestValues(…) is a method that returns a list of top 5 entries (entries with highest values) from a specified map. Method topFive(…) prints out the results as standard output.

For the Task 3 tagsPopularityMap() reads the tags.csv file and returns a map of tags and their popularity value. Method calculateTopFiveWithIDF(…) updates values of tags co-occurrence using IDF value, and a list of top 5 highest values using the findFiveHighestValues(…) I created for Task 2. Helper function roundNumber(…) rounds up numbers to 3 decimal digits and getIDF(…) function calculates IDF value of a tag.

* 1. Code in the appendix

1. *Top 5 tags without IDF*
   * 1. water
        1. nature, 74.0
        2. blue, 71.0
        3. reflection, 63.0
        4. lake, 62.0
        5. landscape, 62.0
     2. people
        1. portrait, 28.0
        2. street, 27.0
        3. bw, 24.0
        4. 2007, 23.0
        5. explore, 21.0
     3. london
        1. explore, 32.0
        2. geotagged, 15.0
        3. graffiti, 15.0
        4. architecture, 14.0
        5. street, 13.0
2. *Top 5 tags with IDF*
   1. *Suggestions*
      1. water
         1. lake, 270.215
         2. reflection, 238.48
         3. nature, 236.189
         4. landscape, 224.4
         5. blue, 208.026
      2. people
         1. street, 99.171
         2. portrait, 87.143
         3. bw, 70.818
         4. 2007, 66.626
         5. man, 53.69
      3. london
         1. explore, 72.335
         2. graffiti, 56.715
         3. geotagged, 54.179
         4. architecture, 52.042
         5. street, 47.749
   2. Reflections