Assignment 3 Report

Peng Li ( peng24 )

1. **Briefly explain what algorithms you use in Step4~Step6.**

Step 4: I am using Apriori Algorithm to mine the frequent patterns for each topic. The Apriori Algorithm use bottom up approach, where frequent subsets are extended one item at a time. The minimum support is 0.5%. Each topic generates around 180 patterns that satisfies minimum support.

Step 5: The program reads frequent pattern files that are generated in Step 4 and then filter out unqualified patterns. One interesting observation is that the size of the closed-pattern file doesn’t reduce compared with frequent-pattern files. This is because the minimum support (1%) is relative big for this dataset. I re-ran step 4 and 5 with minsup = 0.1% and minsup = 0.01%. Both max-pattern and closed-pattern file are reduced a lot compared with the frequent pattern file.

Step 6: purity(p,t)=log [ f(t,p) / | D(t) | ] - log (max [ ( f(t,p) + f(t',p) ) / | D(t,t') | ] )

Where f(t,p) / |D(t)| is the support value that is generated in Step 4. The only confusing part of this algorithm is D(t, t’). D(t, t’) is the union of titles that include pattern p. I recycled the code from Step 3, which read *word-assignments.dat* and save the title and an auto-generated unique id for each title in a hashtable according to each topic. Then based on the key word and unique id, we can find the union easily.

1. **Answer all the questions in Question to ponder.**

**Question to ponder A:** How you choose min\_sup for this task? Note that we prefer min\_sup to be the consistent percentage (e.g. 0.05 / 5%) w.r.t. number of lines in topic files to cope with various-length topic files. Explain how you choose the min\_sup in your report. Any reasonable choice will be fine.

I choose 1% as minimum support for this task. Using 1% generates around 70 to 80 frequent patterns. It is a reasonable number for us to find some useful information from the output. Additionally, I tried 0.1% and 0.01% as well, which generates 2,000 patterns and 9,000 patterns for each topic accordingly. These two minsup values are two small. Especially, when mining the completeness, it takes a huge amount of time to run.

**Question to ponder B:** Can you figure out which topic corresponds to which domain based on patterns you mine?

This is my first semester to take computer science courses (my major was Electrical Engineering). So I am not quite sure which one is which. But based on the Combined Ranking, I get the following conclusion.

0: Machine Learning

1: Database

2: Data Mining

3: Information Retrieval

4: Theory

**Question to ponder C:** Compare the result of frequent patterns, maximal patterns and closed patterns, is the result satisfying? Write down your analysis.

Based on minsup = 1%, frequent pattern result is good. From the .phrase file we can get some useful information. Also, the lossy compression max-pattern is good. It reduced the size of frequent pattern. However, the lossess compression closed-pattern is the same as the frequent pattern. Thus, the algorithm doesn’t reduce the size of the frequent pattern. So it is not satisfying. But when I changed minsup = 0.1% or minsup = 0.01%, the closed-pattern file size reduced a lot, which have good result.

1. **List your source file names and their corresponding Steps.**

**Step 2:**

*Preprocessing.java*

**Step 3:**

*Partitioning.java*

**Step 4:**

*MiningFP.java*

**Step 5:**

*MiningMaxCloP.java*

**Step 6:**

*MiningPurityP.java*

**Step 7:**

*MiningPhrasenessP.java*

*MiningCompletenessP.java*

*CombinedRankingFunc.java*

**Step Mapping Number to Terms:**

*MapNumTerm.java*

1. **Bonus:**

Based on KERT: Automatic Extraction and Ranking of Topical Keyphrases from Content-Representative Document Titles, I implemented the combined ranking. It generated very good result and the output file is in folder combined Ranking folder.