Project 1B: Semi-Supervised Fuzzy K-Means Clustering

CS 145, Prof. Wei Wang

Team: Mining Your Data

Christian Cam - 204 446 732 - c4740674@gmail.com

Daniel Kho - 204 315 203 - daniel.j.kho@gmail.com

June Lee - 604 500 920 - june87lee@ucla.edu

Albert Wang - 504 457 700 - wang.l.albert@gmail.com

Alvin Vuong - 604 337 482 - alvin.t.vuong@ucla.edu

I. Instructions

For our project, we ran it on Windows 10 using Python 3.5 and the latest Scikit-Learn as of December 1, 2016 - version 0.18.1. The file directory structure is as follows:

```
/cs145Project
|
|-- fkm.py
|-- preproc.py
|-- README.txt
|
|-- reuters21578/
    |
    |-- reut2-000.sgm
    |-- reut2-001.sgm
    |-- …
    |-- reut2-021.sgm
```

Here, `fkm.py` is our main script that handles the semi-supervised fuzzy k-means algorithm; it uses `preproc.py` to parse and extract the relevant data from the `.sgm` files. It also manages a data structure of the articles with the specified labels and also selects the seeds for the fuzzy k-means algorithm. The `README.txt` is just a copy of these instructions.

The data in the `reuters21578` folder contains the dataset downloaded from: http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html as specified in the project spec. The `.sgm` files contain the dataset articles in an XML-format.

In order to run our procedure, open up a command-line interface at the root folder of our directory, such that the current working directory contains the `.py` files.

Then the following can be run:

```
$python fkm.py seedStrat seedFrac
```

where `seedStrat = {1 2 3 4}` is the seeding strategy as described in the project spec and `0 < seedFrac < 1` is the proportion of data that will be chosen as seeds.

This should output the pairwise precision, pairwise recall, F1 score, and the number of iterations required for the algorithm to converge for the given seed strategy and the given seed fraction.

Some sample output:

```
money-fx
      Pairwise Precision: 0.5177797051170858
      Pairwise Recall: 0.8728070175438597
      F1 Score: 0.6499727817093085
```

```
trade
      Pairwise Precision: 0.3564614050303556
      Pairwise Recall: 0.7996108949416343
      F1 Score: 0.4931013797240552
interest
      Pairwise Precision: 0.3313096270598439
      Pairwise Recall: 0.9009433962264151
      F1 Score: 0.4844641724793913
Iterations: 15
```

Sample error message:

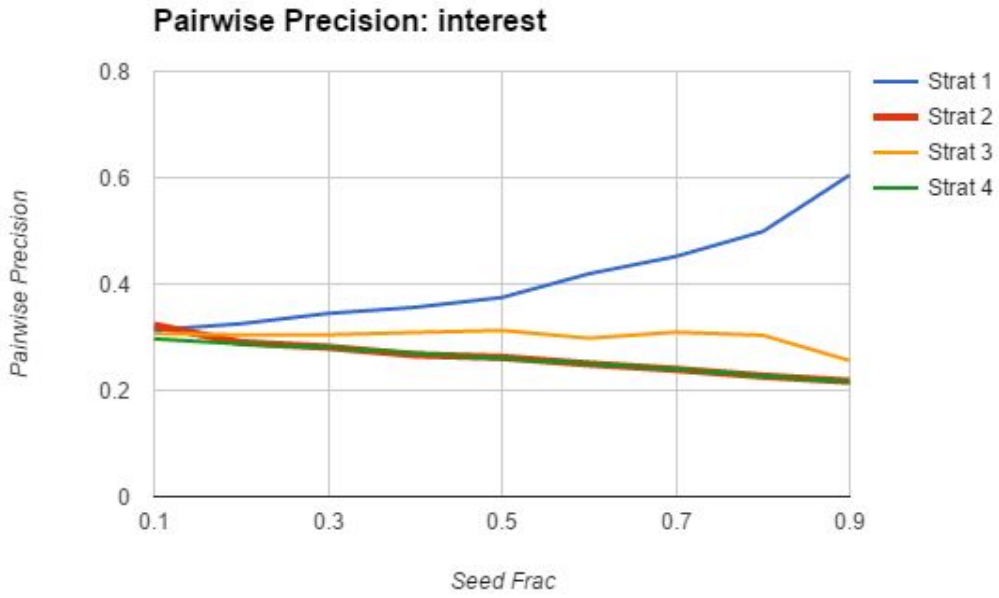
```
Error: too few arguments entered
Usage: python fkm.py seedStrat seedFrac
      seedStrat - a number from 1 to 4, see README.txt for
details
      seedFrac - a number from 0 to 1 (non inclusive),  fraction
of data to use as seeding data
```

Also outputted is a text file `clusters.txt` containing each label's assigned article IDs; these are the clusters that our algorithm produced.

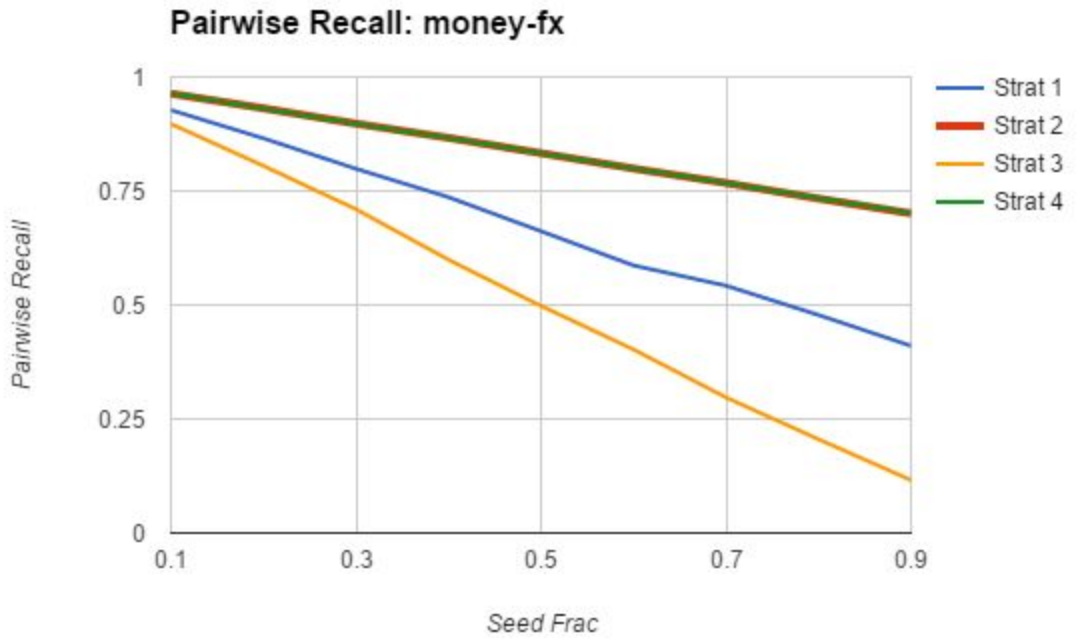
II. Results

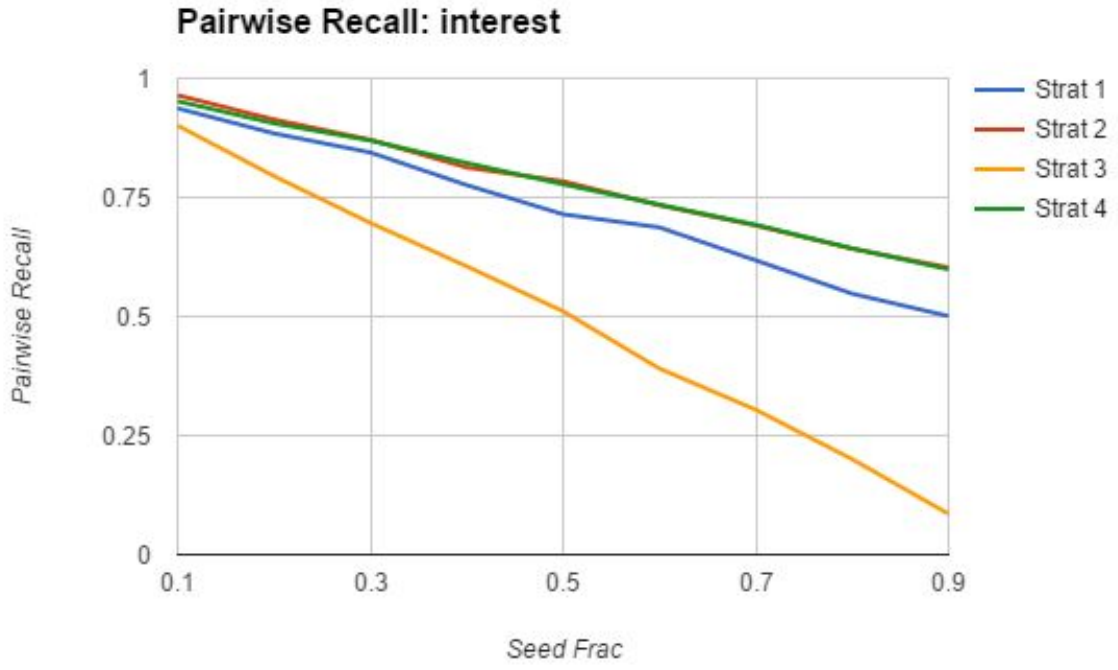
The following are plots for each of the types of comparisons that the project spec required. For each comparison, we have graphs for each of the labels, and on each graph there is a plot for each seeding strategy.

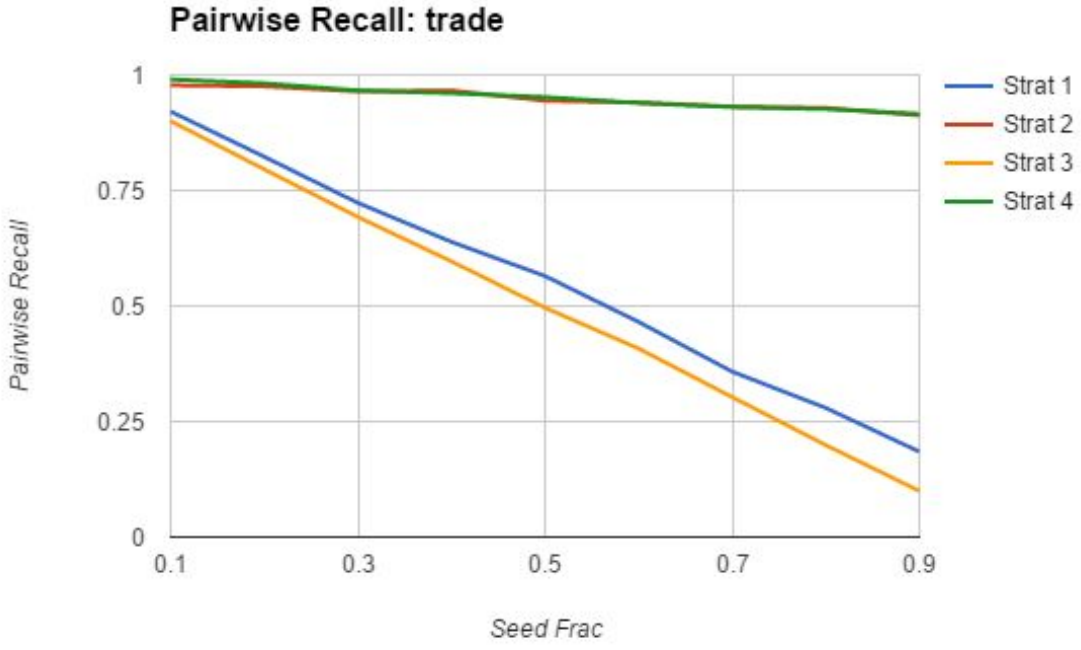
Pairwise precision vs. increasing seed fraction



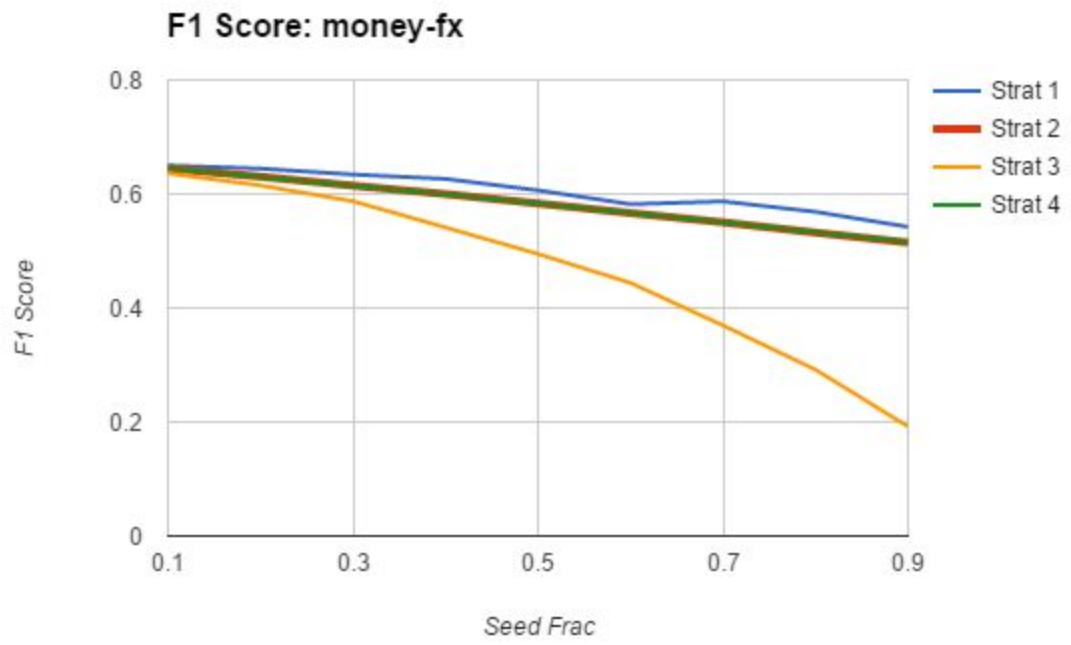
Pairwise recall vs. increasing seed fraction

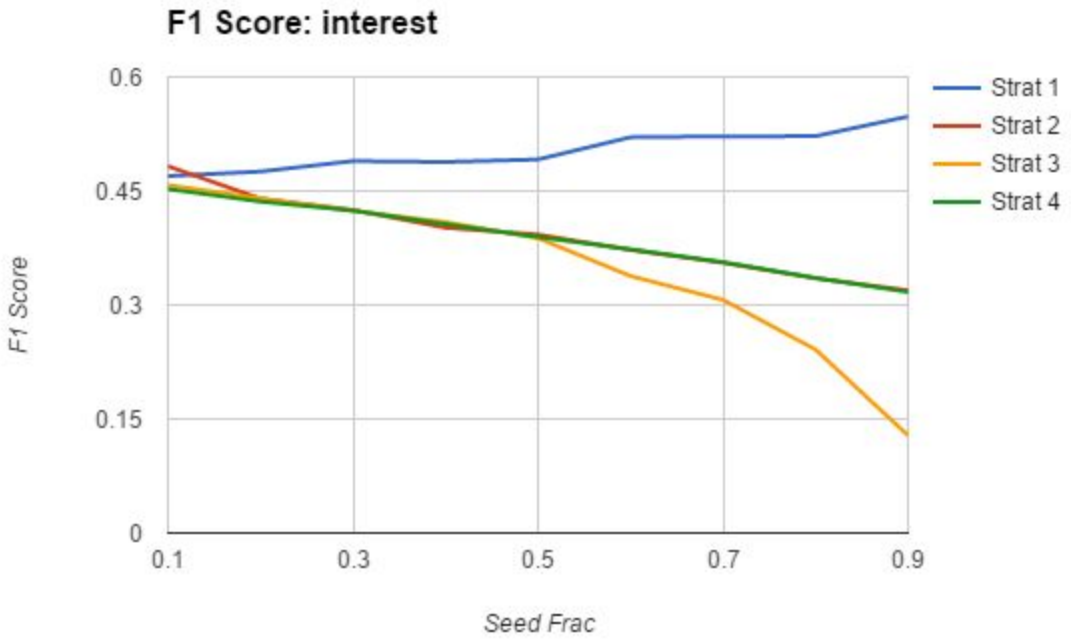


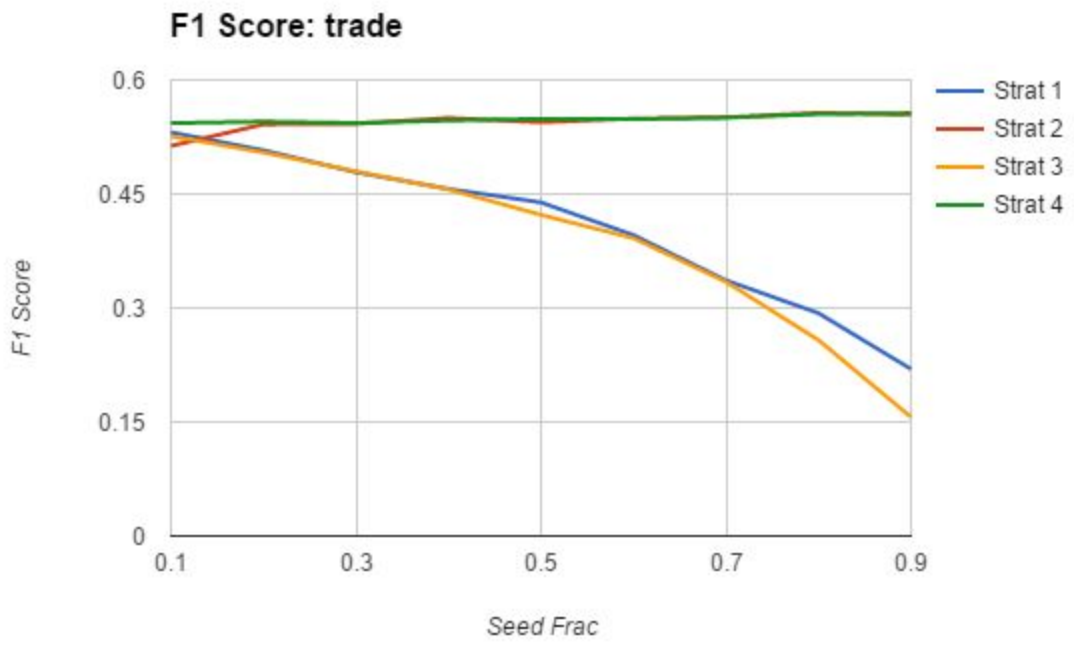

Pairwise Recall: interest
Pairwise Recall
1
0.75
0.5
0.25
0
0.1
0.3
0.5
0.7
0.9
Seed Frac
Strat 1
Strat 2
Strat 3
Strat 4


Pairwise Recall: trade
Pairwise Recall
1
0.75
0.5
0.25
0
0.1
0.3
0.5
0.7
0.9
Seed Frac
Strat 1
Strat 2
Strat 3
Strat 4

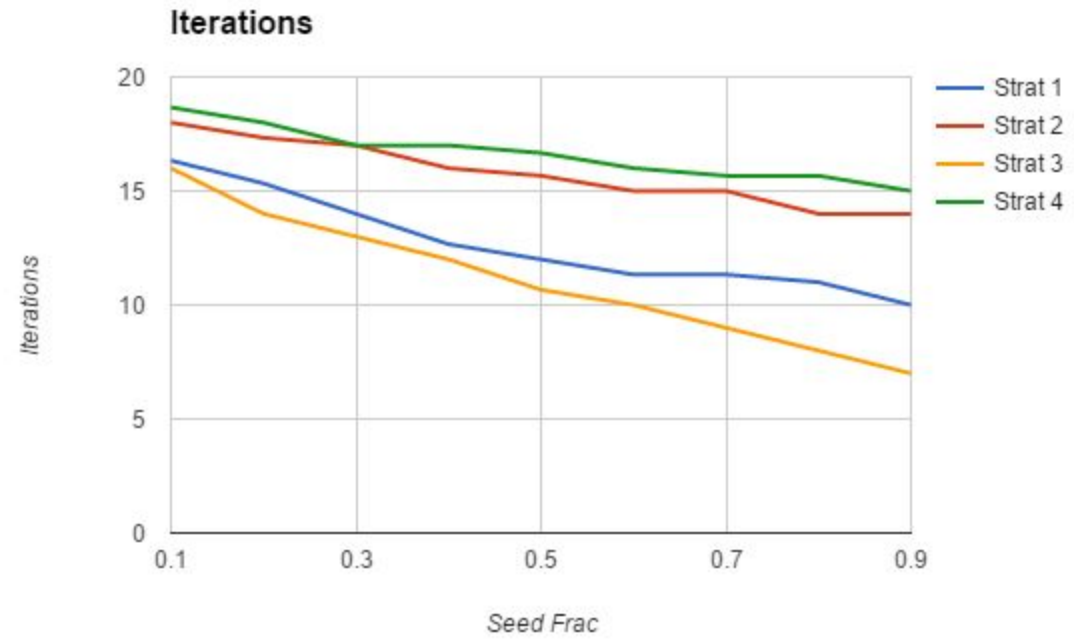
Pairwise F1 score vs. increasing seed fraction







Convergence time vs. increasing seed fraction



III. Analysis

We are primarily looking at four numbers from each of our cluster iterations. The numbers considered are *pairwise precision, pairwise recall, F1 score,* and *convergence time* (which we would be using the number of iteration till convergence). Using these numbers we compare it to the increasing seed fraction.

Strat. 1: uni-label
Strat. 2: multi-label (for all corresponding labels)
Strat. 3: uni-label and multi-label (for all corresponding labels)
Strat. 4: uni-label and multi-label (randomly for some of its labels)

Pairwise Precision Analysis

money-fx
Strat. 1 and Strat. 2 both show an upward trend (with Strat. 1 increasing at a faster rate). However, Strat. 3 and Strat. 4 both show a downward trend.

interest
Strat. 1 clearly has an upward trend while the rest has a steady downward to trend. Important to note this is another example of Strat. 2 and 4 appears to be similar.

trade
Strat. 2 and 4 are again very similar (very sight upward trend). Strat. 1 has a severe downward trend and Strat. 1 appears to maintain a steady trend.

Pairwise Recall Analysis

money-fx
All strategies appears to have a downward trend (especially Strat. 3 that is trending downwards most aggressively).

interest
All strategies are on a downward trend, however Strat. 3 has a severe downward trend.

trade
Strat. 2 and 4 are again very similar (very sight downward trend). However, both Strat. 1 and 3 have severe downward trend.

F1 Score Analysis

money-fx
Overall all strategies goes in a downward trend, however, Strat. 1, 2, and 4 appears to be steady (linear). Strat. 3 falls sharply a downward trend.

interest
Strat. 2 and 4 is very similar (both slight downward trend), while Strat. 1 is upward and Strat. 3 is downward (severely).

trade
Strat. 2 and 4 are again very similar, maintaining a steady trend. However, both Strat. 1 and 3 have a severe downward trend.

Convergence Time Analysis

The above graph shows that convergence time (in terms of number of loops the algorithm took) decreases as the seed fraction increased for all seed strategies. This is expected, with strategy 3 requiring the least amount of

Overall Analysis

A few things to note in general:

Interestingly, seeding strategies 2 and 4 were very similar as they both used multi-labeled data for seeding and also there were only about a fourth of the number of seeding points for multi-label. Meaning for seeding purposes, they mostly ended up using very similar seeding points due to the fact that there was a relatively low number of articles with multiple labels.

Overall, our group decided that strategies 2 and 4 seems like the best strategies out of the four if considering consistency is the most valued quality to clustering. Throughout the different iteration of labels both strategy maintained the same trend throughout different labels which implies the stability of the strategy. However, one negative side to both of these strategies is that they both took the longest to converge compared to the other strategies.

On the other hand, strategies 1 and 3 appear to take more drastic values across the board. The values for these two strategies may at times outperform strategies 2 and 4, but as a result they are a bit more unreliable. We do not know for certain how they will before if we were to extrapolate beyond this range of data, so it is better to go with strategies 2 and 4 because they had consistent trends and may be better for predicting patterns.

The reason for pairwise recall decreasing overall as seeding fraction increased was due to a calculation error in our implementation of the `calc_metrics` function, which found the

pairwise precision, pairwise recall, and F1 scores of the algorithm. Instead of only counting data points within the same cluster, it counted the number of occurrences of the label across the entire data set. The more seeds we have, the more total number of pairs we have in the entire database, so as we increased seeding fraction, we saw that pairwise recall dropped. Whereas, if we looked only at the data points within a same cluster, perhaps the trend might have been different.

Our fuzzy K-Means classifier has some points where we believe it could be improved. Although we chose labels (`money-fx, interest, trade`) such that we would have a good number of uni-labelled points and a good but lesser number of multi-labelled points, these articles all concerned a similar topic. As a result, we believe that these articles would have similar words across the different labels, and would not necessarily be easy to differentiate between when using a fuzzy K-Means classifier. In addition to this, currently, our algorithm uses a threshold on the article-cluster weight in order to determine whether or not it can be considered as part of a cluster. A different potential approach would be to compare the weights an article has for each cluster and choose the largest and most similar ones to determine the multiple clusters.