Evaluating Text Clustering Methods for Text Classification

Mehrbod Sharifi
Language Technology Institute
School of Computer Science
Carnegie Mellon University
msharifi@cs.cmu.edu

Abstract
In this project report, I will evaluate the several text clustering approaches and how they can be used for the purpose of text classification. The particular task is topic classification of 20 Newsgroup dataset and sentiment classification restaurant reviews dataset. Future direction for improving the results will also be discussed.

1 Introduction
Text classification is usually performed in a supervised setting [2]. However, given the large amount of unlabelled data available on the web, the interest for semi-supervised approached has been increased in the past decade [3][5]. There is tremendous amount of research is performed on various aspects of text clustering. For the purpose of this course project, I will limit the scope to evaluation to several more popular algorithms and two possible expansions of them (learning distance metrics and feature extraction).

More recently, a type of classification task has emerged which is typically regarded as sentiment analysis [6]. Classification of sentiment is more dependent on the linguistic phenomenon in text and therefore demands a more careful work on representation and distance metrics of the machine learning algorithm which is used. As a result current state-of-art is still much lower from that of topic classification. I will be evaluating the topic and sentiment classification in parallel in this project.

2 Algorithms
The clustering algorithms that I have selected are K-Means and EM from the partitioning types (they are most popular and generally produce good results). Also, from the hierarchical clustering algorithms, I have chosen Group Average Agglomerative Clustering (GAAC or also known as UPGMA). The implementation can easily include the other popular hierarchical clustering such as single link or complete link but since the time complexity of those algorithms is prohibiting running many experiments, even GAAC is only tested in a few experiments (which didn’t produce good results).

Spectral clustering has been shown [3] to improve the performance of text clustering and a simple version of this will also be implemented.
Since the goal is to introduce the supervision later, I used boosting and Naive Bayes in this project. The reason for choosing these two algorithms was that in both of these algorithms, we can extract the features that have been useful for classification task (namely, the weak hypothesis for boosting and class conditional distributions of words).

### 3 Implementation

The choice of implementation is extremely critical when dealing with text. The most common representation of text in the form of document-term generally produces a very large but sparse document term matrix (figure 1).

![Sparsity pattern of a sample document term matrix – Only 0.06% non-zero](image)

Since a lot of time is spent on this part of the project, I will briefly report the result: the comparison was between using Matlab or Java for this purpose and the final decision was to use Java for all text extraction and learning code and Matlab for all visualizations. The most important factor in choosing Java (despite the much longer development time) was memory management. I found Java around 3 times more efficient in terms memory management measured by the size of matrices (performance were very close). For example, a subset of data after stop work elimination and low frequency terms was 6706x19823 matrix which couldn’t be loaded in Matlab on a good hardware\(^1\) but could be loaded and processed in Java (The final data is still downsized to allow multiple experiments in the short time.)

Other benefit of programming in Java is possible integration with existing packages (such as MinorThird) or also offer the result on the web as applets.

Here is current status of implementations: a framework for extracting and representing text for clustering is designed. The learning algorithms Naive Bayes, K-Mean and GAAC are completed. EM implementation is still underway and Spectral is not started. Boosting was performed by using Boostexter [8].

### 4 Datasets

1. Restaurant Review dataset (RR): 52,077 user reviews for 5531 restaurants extracted from citysearch.com. Data has rating 1-5.
2. 20 Newsgroups dataset (20N) [Course website] The mini version of this data is used due limitation in processing (20 newsgroups, 100 message each).

### 5 Evaluation

In this completely unsupervised case, evaluating the clustering is different than classification task because we need to consider various labeling possibilities for result clusters. The most similar measure to accuracy which is frequently used in clustering evaluation is called *cluster purity* which assigned the cluster label as the majority of the various classes contain and consider all minorities in class as errors.

---

\(^1\) Pentium 32-bit Quad-core Processor, 4 GB RAM
A better measure of clustering labels is called *Rand Index* [7] which is analyzing the agreement between two partitioning. The agreement is when a pair in both partitioning is in the same cluster and in both partitioning it is in separate clusters.

\[
\text{Accuracy} = \sum_{i<j} \frac{\text{\#}(c_i = c_j \land c_i \neq c_j)}{\text{\#}(c_i \neq c_j)} = \frac{\text{\#}(c_i \land c_j)}{\text{\#}(c_i \neq c_j)}, \quad c_i \text{'s are clustering result}
\]

[1] discusses that this measure should be changed to Adjusted Rand Index so that random cluster has a contact expectation (in their case, zero) but I decided to use the original index. Both values of the measures range from [0,1].

Finally, both purity and rand index are only considering the labeling and are not concerned with the quality of the clustering in terms of compactness and separation. To measure that, I have used *silhouette value* [4] which evaluate each points tendency (based on distance measure) to be part of the cluster it is assigned to or any other cluster. Unfortunately, since this evaluation is time consuming, I could not run it for varying parameters but one example is provided. Perfect case is shown in figure 2.

![Silhouette value in perfect clustering](image)

6 Results

Table 1 and 2 summarizes the results for two dataset for K-means and GAAC. Note that the baseline (random assignment of clusters) for 20N is 0.05 and for RR is 0.2. In all cases, the distance measures are cosine and k-means run was without multiple restarts. Two kinds of document vectors are used: the Boolean vector (which is 1 when word is present and 0 when not) and TFIDF which is calculating the TFIDF scores [2] for the terms in the vector. Since the TFIDF result was better all other experiments are run with this method.

In the case of clustering, it is not clear that amount of data will help with the quality of the clusters because there are no labels. In fact, this seems to be dependent on the amount of noise or structure in the dataset (Figures 3 and 5).

Determining the correct number of clusters is another problem (in previous cases I just used the correct number based on prior knowledge). Figures 4 & 6 are showing how the evaluation metric is changing by changing the number of clusters. Applying cross validation in this case a bit different because no learning is performed on the dataset and I used the entire dataset. The training/testing split should be done if I need to choose a single K based on this result.

Visualizing the clustering in such high dimension is very difficult. I search for certain dimension with high variance to produce interesting 2D or 3D plot but the idea didn’t work. In lieu of this visualization, we resort to the silhouette value which is described before (Figures 7 and 8).
Figure 3: 20N varying amount of data

Figure 4: 20N varying number of clusters

Figure 5: RR varying amount of data

Figure 6: RR varying number of clusters

Table 1: 20N set result for k=20 (2000x9441)

<table>
<thead>
<tr>
<th>Method</th>
<th>Purity</th>
<th>Rand</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Means (TFIDF)</td>
<td>0.3800</td>
<td>0.9185</td>
</tr>
<tr>
<td>K-Means (bool)</td>
<td>0.2245</td>
<td>0.9082</td>
</tr>
<tr>
<td>GAAC</td>
<td>0.0855</td>
<td>0.2070</td>
</tr>
</tbody>
</table>

Table 2: RR set result for k=5 (2813x3049)

<table>
<thead>
<tr>
<th>Method</th>
<th>Purity</th>
<th>Rand</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-Means (TFIDF)</td>
<td>0.5517</td>
<td>0.6170</td>
</tr>
<tr>
<td>K-Means (bool)</td>
<td>0.5158</td>
<td>0.5921</td>
</tr>
<tr>
<td>GAAC</td>
<td>0.5187</td>
<td>0.3553</td>
</tr>
</tbody>
</table>

Figure 7 (left top) & Figure 8 (left bottom): Silhouette values for the 20N and RR sets respectively.
The results from Naïve bayes implementation on 20N were 86% and on RR 57% (Results in [6] is for binary case. The baseline is lower: 34% for majority class).

Finally, Boostexter [8] results (1000 rounds), training on 90% of the data. Testing accuracy for RR was 64.91% using unigrams and 65.66% using trigrams. For 20N, the accuracy of unigrams was 72.28%.

7 Proposed Work Final Project

I will complete my implementation of EM and spectral clustering and making comparison to GAAC and K-means. Furthermore, I will look into the two possibilities for improving the quality of any clustering algorithm:

The clustering we have discussed in this paper had been completely unsupervised; however it is possible to incorporate some supervision into the clustering (analogous to semi-supervised settings in classification):

7.1 Improving Representation

As it was explained before, I will look into how the clustering can improve by incorporating the features that can be extracted with boosting or Naïve Bayes (the result for these are above).

7.2 Learning Distance Metric

I will be implementing an algorithm for learning the distance metrics explained in [9].

References