ML HW#4

Wei-Rou Lin, R05922128

1 Feature extraction

The bag-of-word model is used to extract the feature of the sentences. The terms are decided after stopword removal, stemming analysis and discarding the terms of which document frequency larger than 1/20 amount of the document set or smaller than 2. These processes result in 3588 terms.

TF-IDF method is then taken to generate the sparse document-term matrix. After all of the above processes, two different decomposition methods are applied. Finally, the decomposed vectors are normalized for better clustering.

1.1 Decomposition Methods

1.1.1 LSA

LSA is used to decompose the sparse document-term matrix. Fixing the cluster number to 20, I select an output dimension of 100 by minimize the coefficient of variation calculated on the cluster size.

1.1.2 Autoencoder

The autoencoder is a three-layered NN activated by sigmoid function. 20% of the datas are splitted for validation. Binary-cross-entropy is chosen to be the loss function, and NADAM is applied to minimize the loss.

To avoid overfitting, the validation loss is monitored during the training, which would be stopped after 30 epochs with no improvement. The best weigts of the NN leading to a minimum validation loss during the training is saved.

Finally, the training stopped after 81 epochs. The minimum validation loss is 2.129e-3 and the training data loss at the same epoch is 1.248e-3.

Feature	Cluster#	Score
Autoencoder	20	0.67149
LSA-100	20	0.48543
LSA-100	40	0.56591
LSA-100	100	0.37335
LSA-200	40	0.50552
BM25F	-	0.41666

Table 1: The resulting Kaggle score of different feature extraction and clustering implementations.

1.2 Discussion

The Kaggle score of each decomposition methods are presented in Table 1. The score of autoencoder method is significantly higher than that of the LSA method with the same cluster number. This implies the autoencoder learned better semantic informations than LSA for the purpose of reconstrcuting the bag-of-word vectors.

I've tried another method to decide whether or not the two given titles are in the same category without K-means clustering. The method is to search the titles with BM25F scoring using or group of terms in a title as a query. And then the first 10000 results are taken as being in the same cluster as the query. The result is as well shown in Table 1, which is not satisfactory but expectable because of using a sparse vector to cluster the data directly without a decomposition method to extract the requisite sematic information in the sentences and without an advanced algorithm for clustering.

Also, as I trying out different cluster numbers, the minimization of coefficient of variance calculated on cluster size becomes nonsense. Thus, I tuned the dimensions of decomposed vectors, but the difference of resulting score seems not as notable as changing the method or cluster number.

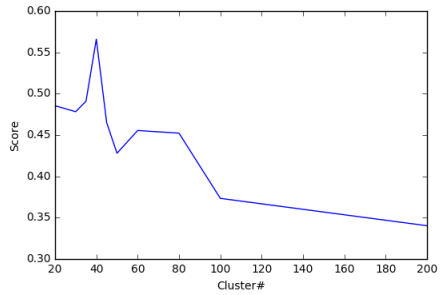

Figure 1: The cluster#-score plot with LSA-100, with cluster# 20, 30, 35, 40, 45, 50, 60, 80, 100 and 200, some scores of which are shown in Table 1.

2 Cluster numbers

Drastic different results is provided by different cluster numbers using K-means clustering. The cluster#-score relation is plotted in Figure 1. The highest score is presented as setting the cluster number to 40, two times as large as the real cluster number.

The K-means clustering will result in totally different output each time even if we provide the same input data, so the highest score at 40 cluster number may just be a lucky try. However, Figure 1 still shows a tendency of accuracy as the cluster number increased.

3 Cluster word analysis

Let's first define the "keyterm" of a cluster as the term maximize the sum of the TF-IDF for all sentences in the cluster. Then the keyterms of each clustering experiment are shown in Table 3.

Each keyterm extracted from true labels can be matched one of the given tags, with the differences mostly come from the stemming process. The only one match of different terms even after stemming is osx/mac. Surprisingly, the keyterms extracted from autoencoder clusters and from true label clusters are **exactly** the same.

In LSA-20, 3 terms are replaced to other terms, two of which (it, what) seems not to provide information in that cluster. I wondered if other terms producing relatively large summation would provide a better representation of the cluster, but neither are the top-5 terms of the two clusters (it, when, why, typ, doe; what, way, best, ar, doe) more informative.

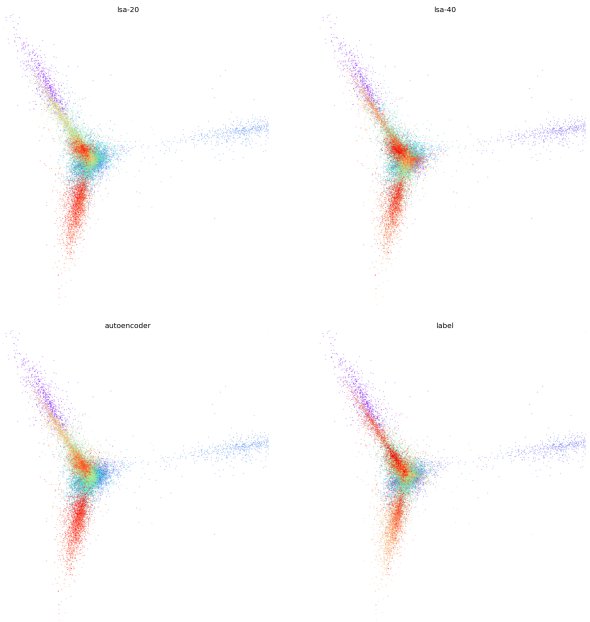

Figure 2: The 2D decomposition of Sentence Vectors colored by clusters. The implementations from left to right, from top to bottom are LSA-20cluseters, LSA-40clusters and autoencoder-20clusters in turn, and the rightmost bottom one plots the true labels of the vectors.

As the cluster number becoming larger in LSA-40, the keyterms include all of those extracted in the true label clusters. The extra terms are colored in Table 3. The sentences in the clusters with matched keyterms are counted to 11755, which is a relatively large part of the data.

4 Visualization

To visualize the clustering results, PCA decomposition is applied to the sparse TF-IDF vectors, resulting in 2-dimensional vectors which then taken to plot Figure 2. There seems to be three "branches" and a "core," and each method can easily distinguish the branches, but the difference in each results could lay in the core.

We can take a closer look at the core via Figure 3. The centers of each cluster are scattered in the figure. "Explicit" terms such as **matlab**, **excel** are relatively close on the plane, and the feature resulting in better accuracy are closer to the true label generally.

Feature	Keyterms
True Label	ajax, apach, bash, coco, drup, excel, haskel, hibern, linq, mac, magento, matlab, orac, qt, scal, sharepoint, spring, svn, vis, wordpress
Autoencoder	ajax, apach, bash, coco, drup, excel, haskel, hibern, linq, mac, magento, matlab, orac, qt, scal, sharepoint, spring, svn, vis, wordpress
LSA-20	ajax, bash, coco, drup, excel, haskel, hibern, it, linq, magento, matlab, orac, qt, scal, sharepoint, subvert, svn, vis, what, wordpress
LSA-40	ajax, ap, apach, bash, coco, command, cre, custom, dat, diff, direct, drup, er, excel, get, haskel, hibern, it, linq, mac, magento, matlab, multipl, net, not, or, orac, qt, scal, set, sharepoint, spring, subvert, svn, url, vis, what, when, wordpress, work
Tags	ajax, apache, bash, cocoa, drupal, excel, haskell, hibernate, linq, magento, matlab, oracle, osx, qt, scala, sharepoint, spring, svn, visual-studio, wordpress

Table 2: The keyterms extracted from different features. Teal colored terms are terms not found on the clusters of true label, whereas brown colored terms are terms found on true label but not found on LSA-20.

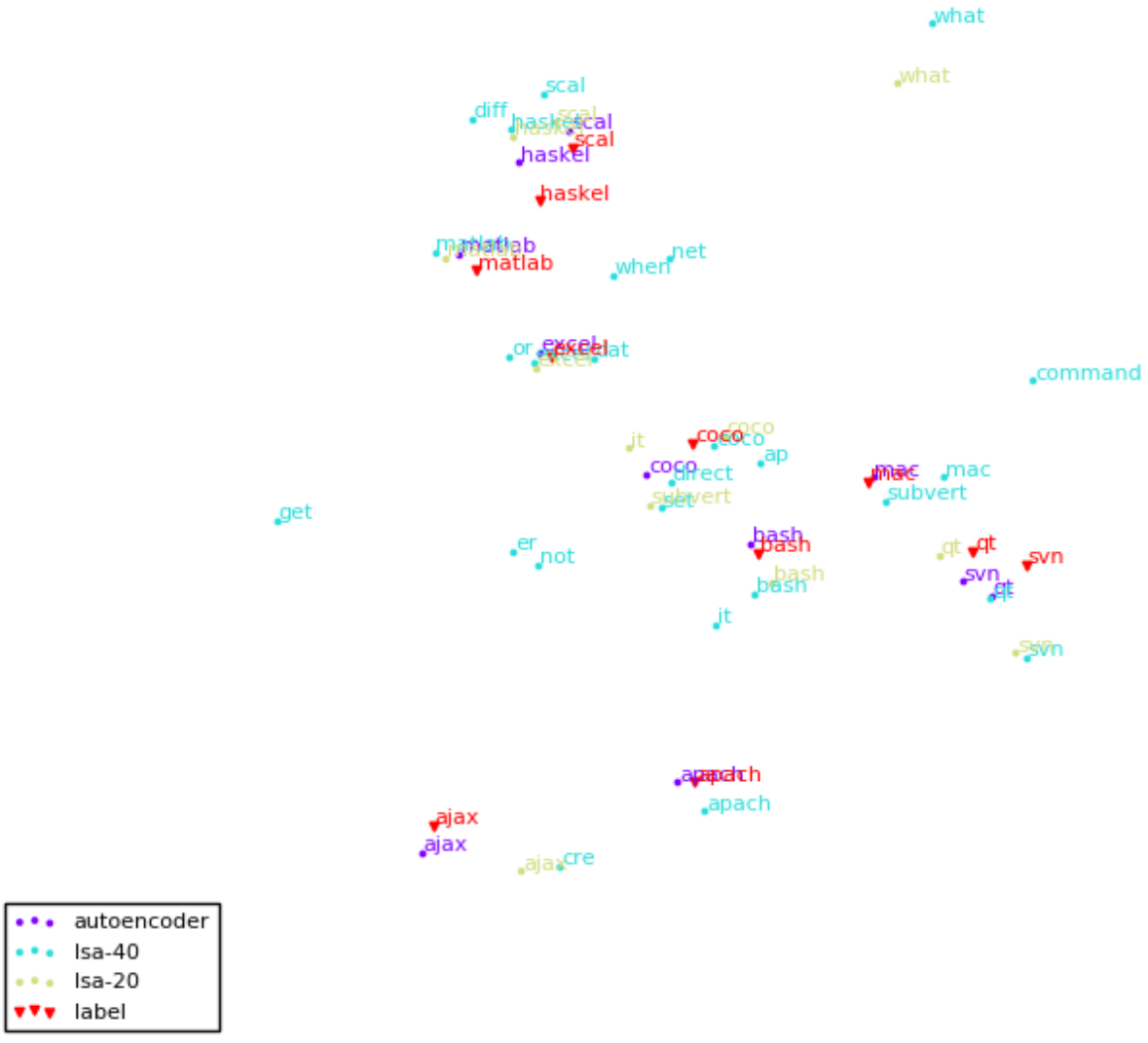

Figure 3: The centers of each label colored by clustering results and marked by its keyterm. The "label" in the legend represent the true label of the data.