CS 5786 Kaggle Competition 1 Final Report
Classifying Handwritten Digits

Team Name
ABC

Group Member
C***** Mok
R***** Sun
Q***** Yu
S***** Zhang

Final Kaggle Score
(Private Leaderboard)
0.77166

- It's a shame my code takes long to run, if it's bad when it's done.

1. Introduction & Notation

The objective of this project is to classify data points into 10 clusters, each of which corresponds to one handwritten digit from '0' - '9'. The total number of observations are 10000. 60 of them are correctly labeled and will be used as seed data. Some additional graph information are also provided for the first 6000 data points, including the 60 seeds. The task is to make predictions on the labels of the last 4000 data points. After taking into account the unlabeled nature of our dataset, mainly different unsupervised method will be experimented in order to cluster the first 6000 points as best as possible. Then some additional supervised learning methods will be used to label the remaining 4000 points.

Notation	Explanation
X	Original Dataset
X_{6000}	Dataset with Graph Information
X_c	Result matrix of first 6000 points after CCA
Y_c	Result matrix of Spectral Clustering after CCA
A_1	Adjacency Matrix with connected edge as 1, 0 otherwise.
A_2	Modified Adjacency Matrix with connected edge calculated using normalized Euclidean Distance, 0 otherwise.

2. Flow of Algorithm

- **Semi-supervised/Unsupervised Learning:**
  - The training data size will be too small and the variance of the model will be large if we only use 60 labeled points. Since we are given additional graph information about the first 6000 points, we will first need to perform unsupervised learning on the first 6000 points in order to obtain proper clusters, which will be useful for future supervised learning if we can label each cluster obtained as good as possible.
  - After obtained the clusters for the first 6000 points, we will need to decide a mapping assignment for each cluster and labels (hand-written digits). We would perform a majority vote algorithm to label the 10 clusters of first 6000 points.
  - As we obtained the partition of the first 6000 points above, we still did not know the correct label of each cluster. Therefore, we used majority vote on these clusters according to the well-labeled 60 points. We experimented two options to assign labels to the clusters obtained from the unsupervised learning methods.
    - **Option 1**
      - **Operation:** For each unlabeled cluster, we assigned the majority label of the points of the labeled 60 points to the current cluster.
      - **Problem:** Since we only 60 points, it is very likely that we will have tie while assigning, which makes the voting processing random.

    - **Option 2 :Optimal Permutation**
      - **Operation:** We generated all the possible 10! permutations of [0:9] and tried labeling clusters with the current permutation.

Record the permutation that can maximize the predicting accuracy of the well-labeled 60 points.

- **Result:** The optimal permutation had 83.3% (50 correct / 60 points) accuracy. We will be using this option as main method to label clusters.

- **Supervised Learning**
  - After we obtained labels for the first 6000 points, we will experimented different supervised learning methods to predict the label for the 4000 target points.
  - Ensemble method: we will ensemble the results from different supervised learning methods and generate the final prediction.

3. Semi-supervised/Unsupervised Learning

3.1 Principal Components Analysis (PCA)

- **Algorithm & Rationale**

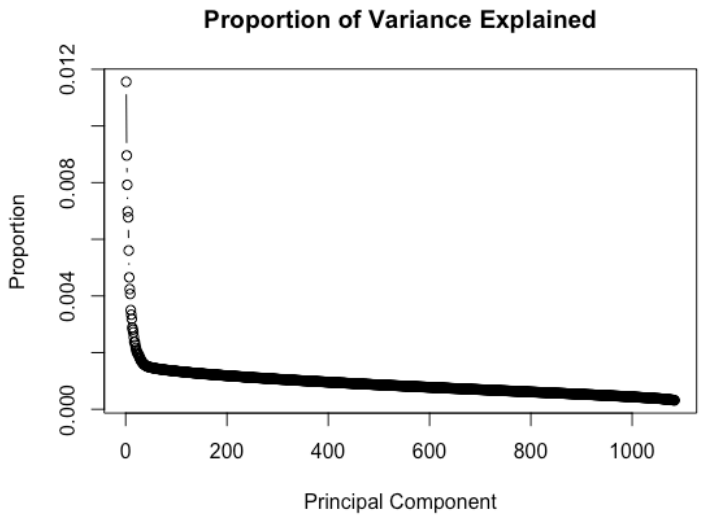
  PCA is used to reduce dimension of original dataset by realizing variance maximization. Since the original data points x_1, …, x_{10000} are given with each $x_t \in \mathbb{R}^d$, where $d = 1084$, for later application of K-Means algorithm, which is essentially a search strategy to minimize the squared Euclidean distance, all data points would appear to be equal-distance as dimension increases significantly, so dimension reduction is performed to avoid such curse of dimensionality and also, to reduce computational complexity. The projection Y to lower dimension of K is given by $Y = (X - \mu) \times W$, where W is obtained by taking the top K eigenvectors after eigen-decomposition of the covariance matrix of original dataset X.

- **Model Application & Interpretation**

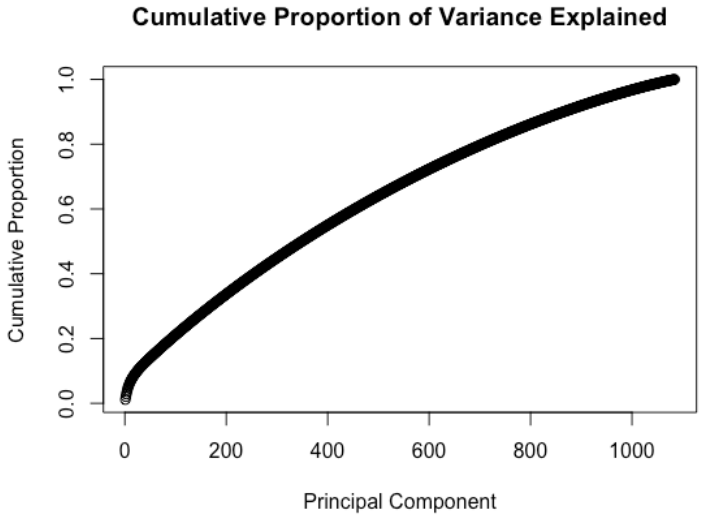
  The ***extracted_features*** dataset is of dimension 10000 x 1084, which is huge and hard to tackle in a whole. We have to apply feature selection methods to make it more approachable.

  We initially want to use PCA (Principal Component Analysis) to model this because PCA is a good method to reduce the dimension of points which are spread to different directions. In theory, PCA uses an orthogonal transformation to convert a set of observations of possibly correlated variables to a set of linearly uncorrelated components, with a ranking by how much they account for the variance for the original data. Applying principal component analysis with a certain amount of components will give the directions in decreasing order and 'elbow' approach will say that the amount of chosen components is right.

  However, after applying PCA to the 6000 data points, we observed that the data points in the ***extracted_features*** were not spread out that the variance among the data points were not big enough that we can do PCA to reduce dimensions. As a result, using K-Means or GMM after PCA will not improve the clustering. As a result, majority vote will not accurately predict the labels for the 4000 data points.



The graph above is the proportion of variance explained by the principal components after PCA ranking from priority high to low. The leading principal component accounts for only 0.012 percentage of the total variance of the ***extracted_features*** dataset. A majority of the components only counts for below 0.002. Then there is no way for us to extract some certain number of principal components because each explains so little about the total variance. If we adopt a large number of them to achieve a threshold, PCA makes no sense here in terms of dimension reduction because we have to take so many components. The other limitation of PCA is that it's a linear projection and thus any non-linear correlation in the original dataset would be lost after performing PCA. In the attempt to address this issue, Kernel PCA is performed but the result is not satisfactory.



The graph above shows there is no significant different between the explanatory power of each principal component. Therefore, if we conduct PCA on the original dataset, we will lose a great amount of information.

3.2 Random Projection (RP)

- **Algorithm & Rationale**
  RP works by multiplying the original dataset by a randomly generated K-dimensional weight matrix that only has 1 and -1. The projected matrix Y is computed by $Y = X^T W / \sqrt{K}$, where W is d x K dimension weight matrix which only contains random 1 and -1. Similar to the PCA algorithm we tried above, RP can significantly reduce the dimension of the original large dataset. Since PCA did not work well due to the similar explanatory power of each principal component, we tried RP since distance between all pairs of data-points in low dimensional projection is roughly the same as their distance in the original high dimensional space.
- **Model Application & Interpretation**
  We initially tried random projection since it preserve the distance between each vector in high dimensional dataset. However, performing Random Project did not save us a huge amount of time since if we simply use all the data to CCA, it only takes a 2-3 minutes. Therefore, we decided to go with the full data of the first 6000 points in CCA in order not lose information.

3.3 Spectral Clustering

- **Algorithm & Rationale**
  - Spectral Embedding/Clustering is useful to embed graph information into meaningful matrix representation, such that the points that are connected closely in the graph, the distance between the vector representation will also smaller, so the similarity information from the graph is preserved.
  - Normalized spectral clustering can find partitions that minimize number of cuts while ensuring that number of edges within each partition is large. By find the K smallest eigenvectors from the Normalized Laplacian Matrix instead of the unnormalized one, the embedded vector space will be less sensitive to outliers far away from their clustering centers, thus yielding more robust results.

- **Model Application & Interpretation**
  We have the ***graph*** information with the similar data points linked together. We will use Spectral Embedding to encode the graph similarity into a matrix representation. This method makes use of the eigenvalues of the similarity matrix of the data to perform dimensionality reduction before clustering in lower dimensions. In the project, our choice of K (# of eigenvectors of K smallest eigenvalue of the Normalized Laplacian Matrix) is 100. The choice is somewhat arbitrary, but since we will utilize some other dimension reduction techniques in subsequent steps, we would rather choose some large value of K to preserve as much information as possible.

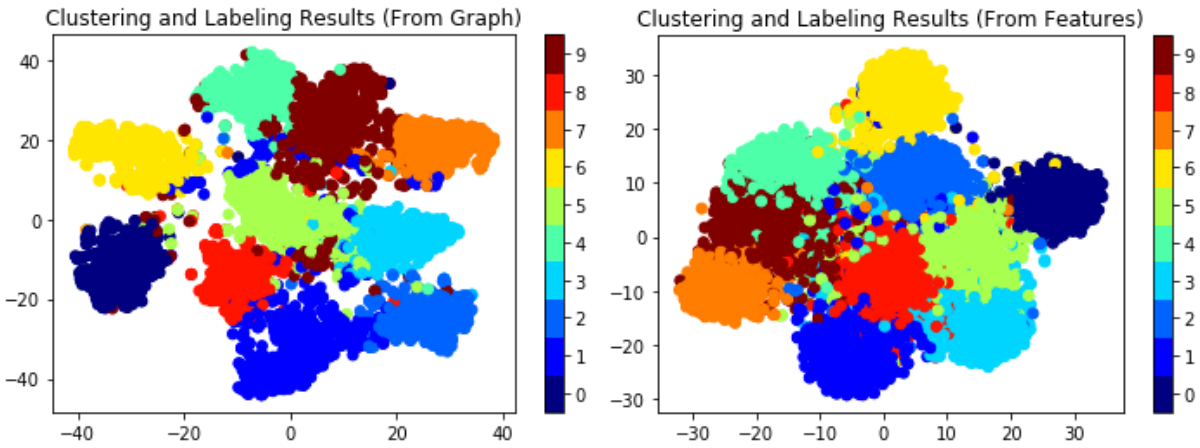
  The similarity matrix is provided as an input and consists of the relative similarity of each pair of points in the dataset. We did Spectral Clustering on the adjacency matrix based on the 6000 points.
  - **Option 1: Adjacency Matrix (Denoted as A_1)**
    We denote each connected edge in the adjacency as 1, 0 otherwise.
  - **Option 2: Adjacency Matrix + Euclidean Distance (Denoted as A_2)**
    In the modification of adjacency matrix, we assume that 2 points that are closer (shorter Euclidean Distance) will have stronger connection. Therefore, for 2 connected points, we assign their corresponding value in the adjacency matrix as $e^{\frac{-d(x_1,x_2)}{\gamma}}$, where we divide 6000 to normalize.

3.4 Canonical Correlation Analysis (CCA)

- **Algorithm & Rationale**
  - Canonical Correlation Analysis is used when data comes in pairs, and we want to retain redundant information between the two views. For example, when performing the task of speech recognition, CCA could be applied on audio data and video data to extract common information between them in order to reduce the noise.
  - One important property of CCA is “scale-free”, which means that data from two separate views doesn’t have to be on the same scale because the idea of CCA is to maximize not covariance but rather the covariance normalized by variance, $i.e.$ correlation coefficient between two views.
- **Model Application & Interpretation**
  - In order to reduce dimensionality and noise before performing any distance-based clustering method, we found Canonical Correlation Analysis very helpful in the project because for the first 6000 points, we had information from two points of view. One is from the ***extracted_features***, and the other is ***graph*** information that some of points are connected to each other. Since these two data were not necessarily from the same source, CCA’s scale-free property would still retain the common information between them and reduce dimensionality on both view.
  - Mathematically, we would try to find two projection views X_c, Y_c such that, the correlation coefficient is maximized.

$$\mathrm{Corr}(X_{features}, X_{graph}) = \frac{\frac{1}{n}\sum_{t=1}^{n}(x_t - \frac{1}{n}\sum_{s=1}^{n}x_s)(x_t' - \frac{1}{n}\sum_{s=1}^{n}x_s')}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t - \frac{1}{n}\sum_{s=1}^{n}x_s)^2}\sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t' - \frac{1}{n}\sum_{s=1}^{n}x_s')^2}}$$

  - After obtaining two projections views from CCA, we would perform clustering method on one of the two views, trying to find clusters such that
    - The cluster-label mapping assignment gives that best accuracy on X_{seed}
    - By visualization, 10 clusters are separated well and centers are as far as possible
  - After doing K-means clustering on the two views of CCA projections, we tried 10! permutations of label-cluster mappings and chose the one which yielded the best accuracy on 60 labeled points. It’s clear that the projection view from the graph information (plot on the left) has better clusters results and the clusters are well separated, thus we will use that view for future clustering and prediction purpose.



- **Method Limitation**

A limitation of CCA is that it is still a linear projection. It can only detect linear correlation between the two domains that is globally valid throughout both data sets. So the projection result for CCA might not be perfect if there are non-linear relationship between two views.

3.5 Clustering

- **Algorithm & Rationale**
  Two useful clustering algorithms we learned from lecture are K-Means and Gaussian Mixture Model (GMM).
  - **K-Means** works by randomly initializing K centroids, assigning points that are closest to each centroids (according to Euclidean Distance) in clusters and therefore updating each centroids to the the mean of each cluster. K-Means works better when the clustering of points are round-shaped.
  - **GMM** works better when the clustering of points are ellipse-shaped. GMM works by first randomly assign K centroids $r_j$where $j \in \{1, \ldots, K\}$ and ellipsoids $\sum^{m-1}$for each cluster, set $m = 1$. Then
    i. For each $t \in \{1, \ldots, n\}$, set cluster identity of the point as $c^m(x_t) = argmin_{j \in [k]}(x_t - r_j{}^{m-1})^T(\sum^{m-1})^{-1}(x_t - r_j{}^{m-1})$.
    ii. For each $j \in [K]$, update $r_j{}^m = \frac{1}{|c_j{}^m|}\sum_{x_t \in c_j{}^m} x_t, \sum^m = \frac{1}{|c_j|}\sum_{t \in c_j} (x_t - r_j{}^m)(x_t - r_j{}^m)^T, m = m + 1.$
  - **Model Application & Interpretation**
    i. After performance CCA, we obtain 2 matrices (X_c from 6000 points, and Y_c from the result of Spectral Clustering). We now tried performing Kmeans and GMM on either X_c or Y_c.
    ii. Since we observed that K-means has slightly better performance according to the result on validation set, we will choose K-Means as our clustering method.
  - **Use of 2 Views**
    i. According to our validation set approach, Y_c (prediction of CCA on result of Spectral Clustering) has a better performance in general.
    ii. According to the low dimensional cluster plot of $X_c$and Y_c generated by CCA, we observed that both the points and the clusters of Y_c were more separated. Therefore, we naturally selected Y_c to be used in clustering on the first 6000 points.

4. Supervised Learning

Validation-Set Approach: As we obtain the labeled 6000 points, we use 80% (4800 points) of the labeled points to train supervised models such as SVM, KNN and Neural Network, and then use 20% (1200 points) as validation data to assess model performance and choose the optimal hyper parameters.

1. **Support Vector Machine (SVM)**
   a. SVM works by finding a hyperplane that is closest to the a point of each class in order to make proper cuts among high dimensions. In this case, we obtained the hyperplane using the training points and labeled target points according to the cuts of the hyperplane.
   b. **Kernel:**
      i. **Linear kernel:** Use hyperplane or line to cut training points.

ii. **Radial Basis Function (rbf) kernel:** Gaussian kernel, clustering points nonlinearly.

2. **K Nearest Neighbor (KNN)**
   a. The KNN algorithm first uses K-means algorithm to find the centroids of 10 clusters by minimizing the total within cluster distance. After obtained the centroids from the training set, KNN assigns label to each unlabeled points according to the majority label of K nearest labeled points.
   b. In this case, we use K = 20.
3. **Neural Network (NN)**
   a. Neural Network is a widely used method to deal with classification problem.
   b. In this case, we constructed a fully connected neural network with 5 hidden layers. Numbers of neurons are 1084, 500, 500, 2000 and 30 accordingly and the output layer with 10 categories. In this case, as the weights between 2 connected layers are computed using training data, we plug in target points and use weights obtained above to get predicted label for the target points.
   c. **Problem**: NN works like a black box and computation is expensive. It is hard to precisely adjust the parameters in it.

5. Ensemble Method

After obtaining the 3 models, which all have great predicting power (>71%) on the test data, we improved our prediction by ensembling the results of the 3 models above since ensemble method can remove bad prediction in our base model. According to the results above, we observed that KNN obtained the highest scores on both our testing data and Kaggle. Therefore, we use the result of KNN as a base and perform ensemble method by:

1. Use the majority vote among 3 results as the new result.
2. If 3 results are all different, use the result of KNN (since KNN is the base model).

6. Attempts & Result

Description	Score		PCA	Spectral Clustering	CCA	Majority Vote	Supervised Learning		
	Local	Kaggle					SVM	KNN	NN
Baseline	N/A	0.1800	x	A_1	X_c	Option1			
SVM (linear, Cost = 0.01)	0.634	0.71437		A_1	Y_c	Option2	x		
KNN (K = 20)	0.636	0.74625		A_1	Y_c	Option2		x	
NN	0.54	0.63000		A_1	Y_c	Option2			x
Emsemble	0.624	0.76062		A_1	Y_c	Option2	x	x	x

(In the table above, “x” means this method was used in the current combination. If a cell is filled with value such as A_1or Y_c, it indicates the matrix or value used in the corresponding method.)

According to the table above, we observe that the baseline model has very low predicting power. This is caused by the randomness in our baseline majority vote. As we change our majority vote to a more advanced one, which finds the best permutation of clustering

assignment to maximize the accuracy of well-label 60 points, the Kaggle score was significantly increased to above 70%.

7. Possible Improvements to the model in the future

- Since the correctness of the labels of first 6000 points directly influences the supervised learning process, we could try different majority vote algorithms to label the first 6000 points and determine optimal majority vote according to the score from cross validation.
- Instead of clustering and labeling the first 6000 points all at once, EM algorithm could be utilized to label a small portion of them with the highest confidence level. And then treating them as seeded data and re-run the learning process. So ultimately we could iteratively make better predictions on the first 6000 points, and the correctness of their labels is crucial to later learning stages.

8. References

- Piazza forum
- Lecture notes and slides