CS 4786 Competition 1 Report

Rajesh Bollapragada, Xinyu Ma, Anilkumar Vadali, David Wang

November 5, 2017

Abstract

This document presents the intuition behind the unsupervised machine learning workflow that we followed to accurately classify the dataset of 10000 unlabelled handwritten digits. Our classification algorithm included a sequence of dimensionality reduction and clustering techniques that were selected to reduce the noise of the data set while, at the same time, accurately classify the handwritten digits. The techniques included in our final workflow include spectral clustering, k-means clustering, kernel PCA, and an EM based gaussian mixture model approach to both estimate parameters and cluster data points among other hyperparameter optimization methods. We describe how we utilized these techniques in further detail in the documentation below. Our clustering and classification methods were ultimately successful in that they achieved a clustering accuracy of 81.625% on the test dataset, corresponding to a position of 14th place on the Kaggle leaderboard out of 133 teams. Note that due to teams being allowed to make multiple submissions, our true standing was much higher.

Summary of Implementing the Model

Prior to implementing our model, we were provided with limited information about the nature of the dataset. We were given a subset of the data with 60 labelled points such that six points from each of the ten labels were provided. We were also provided with a similarity table about 6000 points within the dataset. This table was further analyzed after being converted into an adjacency matrix. Using this available information, we carefully selected and trained dimensionality reduction and clustering algorithms in order to understand and accurately cluster the remaining 4000 test data points. The model and assumptions about the dataset were updated several times over the course of the competition by means of a trial and error process, after studying the results provided by our previous models. This analysis led us to either update or replace the algorithms that we initially used with those that would better describe the nature of the dataset. Throughout the course of the competition, the accuracy percentage provided by Kaggle upon submission of test labels was used to measure the strength of the current classifier. The descriptions about our trial and error process are presented below in the report.

Outline of Final Workflow

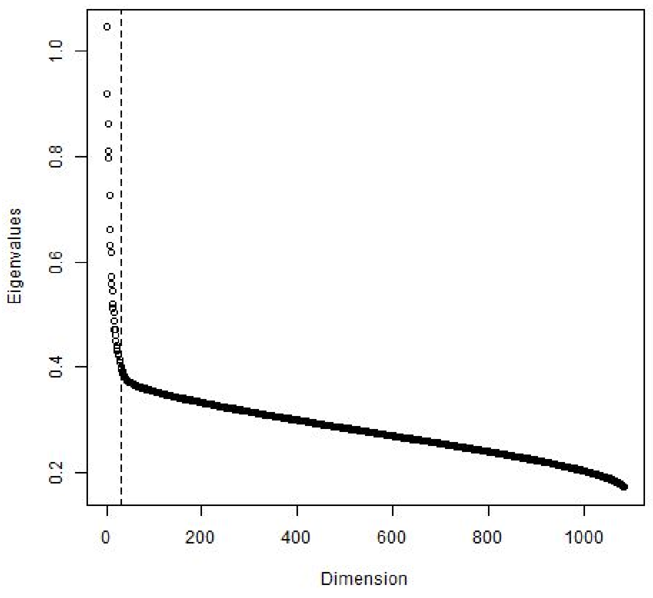
1. Plot the eigenvalues of the covariance matrix of the 6000 data points described by the similarity graph. Identify the optimal dimensionality which retains the most variance by using an elbow plot. The optimal dimensionality was found to be 30. Reduce the dimensionality of the data points using PCA to 30 dimensions.
2. Perform spectral clustering analysis using the 6000 data points by deriving an adjacency matrix from the similarity graph.
3. Cluster the spectral embedding results with the 60 labelled data points as centroids using the k-means algorithm and merge them into 10 clusters based on their original label where each cluster corresponds to a different type of handwritten digit.
4. Use Kernel principal component analysis (PCA) with the radial basis function (RBF) kernel to perform dimensionality reduction. Perform PCA to reduce the data dimensionality from 1084 to 40 dimensions. Optimize the hyperparameters of the RBF kernel to produce the best results.
5. Train a Gaussian mixture model (GMM) based on the data labels of the 6000 points obtained from spectral clustering. The GMM used a EM approach to predict parameters for the model.
6. Use a weakly supervised approach to iteratively train the GMM based on the previously generated labels for the 6000 training data points and use the improved model to reclassify the training data points. Iterate through this process approximately five times.
7. Use the fully trained GMM to predict label for the 4000 test data points.

Methods

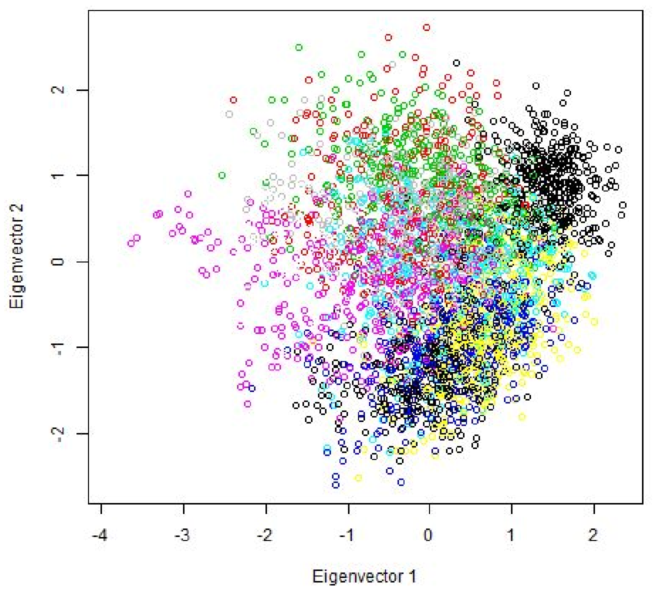
As mentioned earlier, we were provided with three different datasets to help with the completion of this task. The first dataset provided information in the form of a 1084 dimension vector for each of the 10,000 handwritten digits. The second dataset was a similarity graph for the first 6,000 handwritten digits illustrating which data points were similar to one another. Finally, the third dataset included 60 labeled data points where six labeled points were provided for each of the ten digits.

In order to develop a successful and accurate model, we initially attempted to select the best clustering approach by grouping the given labelled points into clusters of the same label. The clustering method we thought would work the best is the k-means algorithm. The k-means algorithm first initializes a centroid for each cluster which can be random or preset. Then each iteration of the algorithm assigns the data points to the closest centroid by Euclidian distance and recomputes the centroids of the new cluster. This is repeated until the centroids do not change after an iteration or a fixed number of iterations (we use the later). The main reasoning behind utilizing k-means clustering over other clustering method, such as single-link clustering, due to the fact that it would be able to handle outliers better.

First, we experimented with the k-means clustering algorithm without initial centroids on the 60 labelled data points. However, this technique produced very inaccurate results that also exhibited high variance in the clustering assignments. After abandoning this method, we then attempted to cluster these data points still using the k-means algorithm but instead by passing initial means as input into the algorithm. The ten initial means were computed by taking the average of the six data points corresponding to each label. The resulting clustering assignments were no longer random and slightly more accurate than without passing the initial means. However, using this version of the k-means algorithm on the 4000 test data points yielded a clustering accuracy of only 46%. We then decided to try using the 60 points as centroids as opposed to ten was primarily based on the fact that each handwritten digit could be displayed in many unique or illegible ways. As a result, if we were to restrict each digit to one representation, we may end up clustering several points incorrectly. For example, 1's may look like 7's in one case and 0's in another case and therefore will be incorrectly clustered in some cases. By using all 60 centroids in clustering, we end up considering 6 different representations for each label which increases the chance of each digit being put into the correct cluster. The 60 centroid k-means approach which involved clustering the data with 60 centroids and then merging the clusters into 10 based on their original labels achieved a score of 55% which was a significant improvement. Before moving forward, we decided to perform Principal Component Analysis, or PCA, on the test data and plotted the eigenvalue of each dimension shown below. Principal Component Analysis is a method that reduces the dimensionality of the data by projecting the data in the direction of maximum variance which preserves most of the critical information of each data point after the reduction. Accordingly, to find the direction of maximum variance, we pick the eigenvectors corresponding to the top k eigenvalues of the covariance matrix of the dataset to reduce the dimension of the data set to k dimensions. However, in order to find the optimal dimensionality, we must first plot an elbow plot to observe at what eigenvalue the change in the variance in the data starts to decrease.

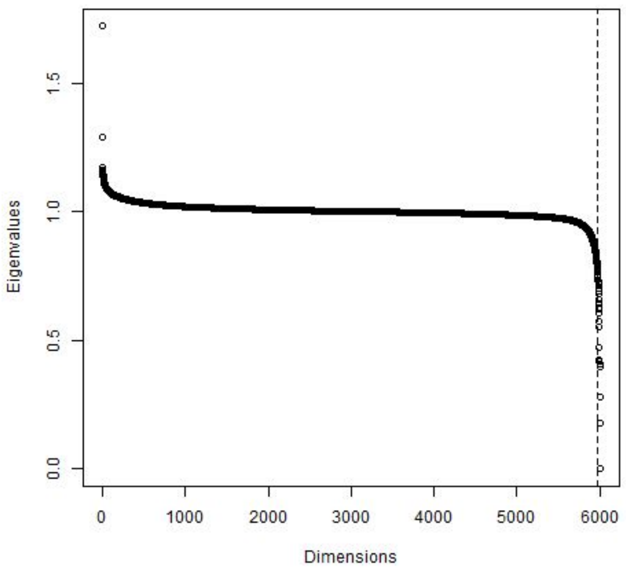


Based on the plot above, we realized that the first 30 dimensions explain most of the variance in the data and thus tried the 60 cluster k-means approach on the test data with its dimensions reduced to 30. This method achieved a score of 64%. The clustering can be visualized in the plot below. One of the main disadvantages of this method was that there appeared to be a lack of separation between points in PCA. As a result, we considered the Kernel PCA method instead, which is explained later on.

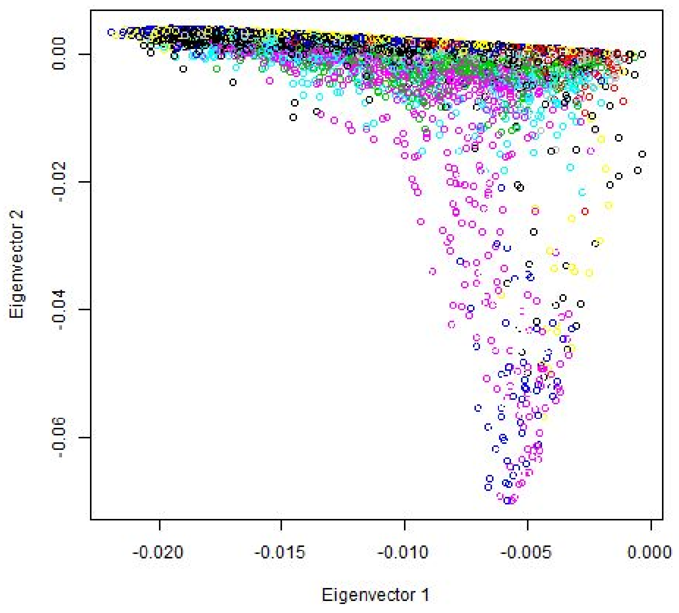


Through these preliminary experiments, we realized two important ideas. First, the 60 centroid clustering approach is superior to the 10 centroid approach simply because of the high variance in the way handwritten digits look even among the same digits. Second, dimensionality reduction is crucial to improving accuracy by removing unnecessary information. Furthermore, we realized that since we are using only the information from 60 data points to cluster the remaining 4000 points, our clustering algorithm was highly prone to overfitting. As a result, we then attempted incorporate information from more data points in order to accurately cluster the test points. Therefore, we turned our attention toward the information in the similarity graph.

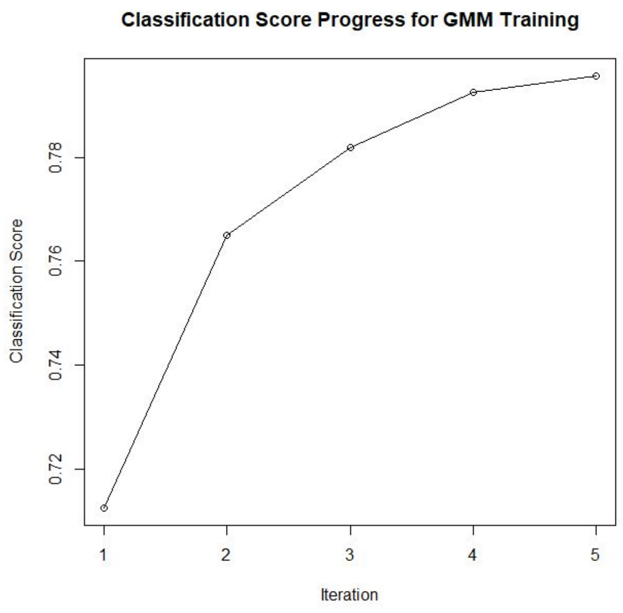
The similarity graph indicated pairs of points that were similar within the set of 6000 points of that were not the test points. Similarity was defined as the most likely assignment between points that were below a certain dissimilarity threshold. The information in the similarity graph was exploited by converting it to an adjacency matrix and then performing spectral clustering to cluster the graph according to the ten possible labels. Spectral clustering effectively creates clusters by exploiting the mutual adjacency of nodes or points in the graph network. We followed the spectral clustering algorithm by creating a Laplacian matrix using the adjacency matrix. We then decided to group the 6000 nodes of the graph into 60 clusters where each centroid was one of the labelled data points. As a result, we created a spectral embedding using the bottom 30 eigenvectors that were not the zero vector and performed k-means on the embedding. We observed that extra dimensions add minimal information after the 30th eigenvalue, which is located on the elbow of the graph. The graph of our elbow plot was displayed in the figure below.



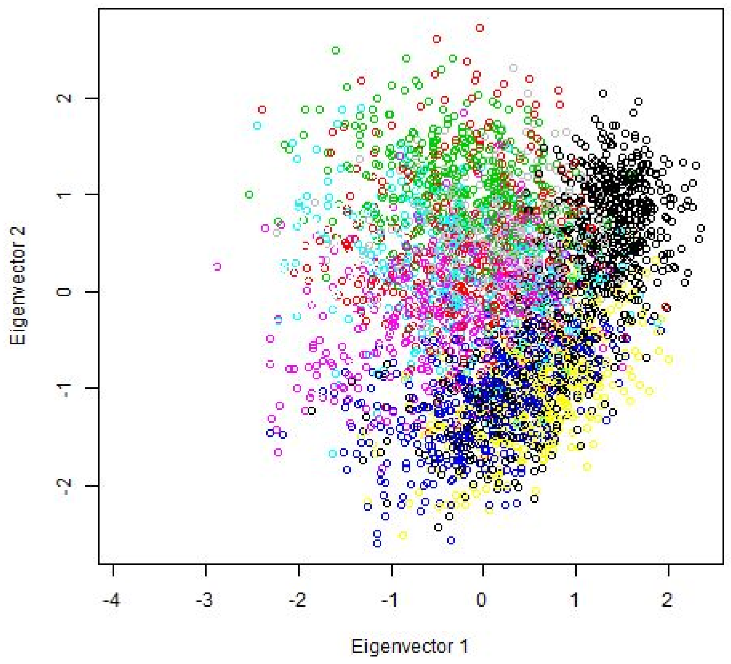
It is important to note that each row in the spectral embedding matrix corresponds to the data point in the same row in the data matrix being passed to the spectral clustering algorithm. As a result, we used the spectral embeddings corresponding to the 60 labelled data points as initial means into the k-means function in the spectral clustering algorithm. By grouping clusters that had the same label, we were able to generate a training set where each of the 6000 points that were not the test points had a label from zero through nine. The clustering can be visualized below.



This concludes the section on individual testing for analyzing the performance of k-means on the features and spectral clustering on the graph. The next step is to combine these result to improve clustering accuracy. Given these labels, we were able to train a GMM using an EM based approach to estimate parameters. This approach involves initializing random parameters for π (percent change of choosing a cluster based on its size), μ (the mean of the MVN on the cluster), and Σ (of the covariance of the MVN). In the E step, these parameters are used to estimate clustering assignments. Then in the M step, this result is used to update the parameters to the MLE value. Each iteration of the EM algorithm improve the MLE of the parameters. This was done using the MclustDA method in the R package mclust. The trained GMM model is now suitable to predict the labels for the test data. However, we only achieve a score of 72% with 1 iteration of training. We then devised a weakly supervised approach the involves retraining the model on the training data (6000 points) until its prediction error is minimized. This was done by using the model to predict the labels for the 6000 data points and using those labels to retrain to the model. This was repeated several times and the improvement after each iteration is shown in the graph below.

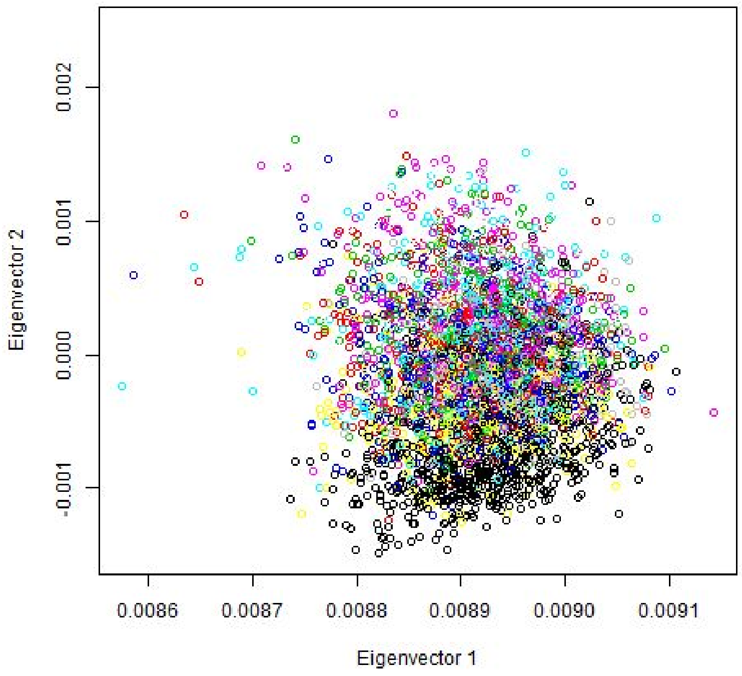


The max score we obtain is 79%. The clustering can be visualized below.



While this method improved our score significantly, it will eventually reach a cap on improvement. This is becauses the initial set of labels obtained from the spectral clustering process was not completely accurate. The graph linked points that had different true labels simply because they were similar (i.e. a thin 0 and a 1) thus some points that were originally misclassified would never be corrected by retraining the model.

While our clustering accuracy had improved significantly as a result of spectral clustering training and iterative gaussian mixture model clustering, we found that the dataset still exhibited significant noise. After doing careful research (Murphy, Kevin P. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2013.), we had found that Kernel PCA is an effective method of removing noise from a dataset, especially a dataset that contains image data. In order to implement the Kernel PCA we decided to use the radial basis function (RBF) kernel to describe the interactions features in the higher dimensional feature space. The RBF kernel takes the inner product of feature vectors that were projected into an infinite dimensional feature space. We therefore chose the RBF kernel for performing kernel PCA in order to utilize its impressive expressivity. In order to perform kernel PCA, a 6000x6000 kernel matrix was constructed where each element (i,j) was the kernel dot product of data vector i and data vector j. After performing a elbow plot similar analysis, the top 100 eigenvectors of the kernel matrix were utilized. We also optimized the hyperparameter of the RBF kernel (σ) through repeated submissions on Kaggle until the highest accuracy was achieved. The kernel matrix was multiplied by its eigenvector matrix in order to get the projected data. This projected data was then passed through the iterative gaussian mixture model workflow using the labels derived from spectral clustering, until the test point labels converged. Upon submission, we noticed that our accuracy jumped to 81.625%. The below graph shows clustering of data points projected onto two dimensions after kernel PCA. The graph shows relatively separated and distinct clusters similar to the graph above. The separation, would be more evident if it were possible to visualize the graph in a higher dimension.



Intermediate/Failed Models

In addition to the various clustering models and the different PCA failures and improvements that were considered and described above during the development of our model, we also considered two different and unique methods that branch away from our traditional approach, but still provided some information in developing our the best model for this project.

<u>CCA</u>

Since given features extracted based on images of the handwritten digits and an unweighted similarity matrix, an intuition is that we can try to use CCA to only extract the redundant information in both views and filter out those noises. We performed CCA on 6,000 training data of dimensions of 1084 with spectral embeddings which have dimensions of 30, thus reducing the dimensionality of both views to 5. Then we applied dimension reduction learned on the training data to 4,000 test data. Next, we performed K-Means clustering on all the 10,000 projected data points with the 60 labeled data points after projection as centroids and merged them into 10 clusters. The clustering result got a score of 56.9% on Kaggle, which was below 60%. Since K-Means approach is very sensitive to the initial centroids, we decided to use means of seeds as centroids in the next experiment to see if there is an improvement of performance.

We obtained ten means by taking the average of the six projected data points corresponding to each label. Then we performed K-Means clustering on all the 10,000 projected data points with the ten means obtained above as input. The clustering result got a score of 57.9% on Kaggle, which went up slightly but still had accuracy below 60%. It pointed to the possibility that applying CCA on features view and spectral embeddings was not an efficient approach to reduce the noise of our data set. Since some noise such as the background of images can also belong to the redundancy of two views, we cannot eliminate such noise using CCA. A more likely cause is that the test data is very different from the training data but we are only given a similarity matrix of 6,000 data. Thus we decided to experiment with other reduction techniques.

Implemented Bonus Methods (Bootstrap Clustering Model)

Ensemble classifiers in supervised learning are well known for their low training error as well as low variance and bias in classification respectively through bagging and boosting. As a result, we attempts to use bootstrap techniques to build a bootstrap ensemble clustering model. This model was based on the principle of bagging, which generates an ensemble classifier where each individual classifier in the ensemble is trained on a subset of the training points. The label produced by the ensemble classifier is an average or mode of the set of labels produced by the individual classifiers. In this manner, we generated an ensemble of GMMs where each GMM is the ensemble was trained on the 6000 training points each projected to a different dimensionality. The ensemble was then tested on the 4000 test points and the label for each test point was defined as the mode of the individual GMMs in the ensemble. This classifier worked relatively well and resulted in approximately 70% classification accuracy. However, while this technique had much potential for improvement, we ultimately abandoned this technique due to the development of more immediate classifiers that exhibited higher classification accuracy. The source code for this classifier is available in the code portion of the documentation.