COSC 522 Machine Learning (Fall 2024)

Project 3 – Dimensionality Reduction and Unsupervised Learning (Due 10/12)

Objectives:

The objective of this project is two-fold. First, to practice the usage of dimensionality reduction as one of the preprocessing steps and study its impact to the classification performance. Second, to get an in-depth understanding of unsupervised learning algorithms and how to formulate a problem in the context of unsupervised learning, e.g., the image compression problem.

Data Sets:

MNIST (used in Project 2) and a beautiful flower image.

Tasks:

- **Task 1: The impact of dimensionality reduction.**

  In this task, you'll get to experience a few very useful applications of dimensionality reduction, including visualization and reduction of computation. MNIST will be used in this task. A bit **heads-up** that you'll be tasked to learn and practice on t-SNE in the final project. Like PCA, t-SNE is an unsupervised dimensionality reduction method; but unlike PCA, t-SNE is nonlinear and has been popularly used for visualization purpose. You are welcome to start reading the t-SNE paper by Maaten and Hinton [1].

  - Task 1.1: Implement the supervised dimensionality reduction method, FLD.
    Description: You should write a module with input being the training data WITH the label information and output being the projection matrix, W. Note that since MNIST has c=10 classes, the number of dimensions it will be reduced to should be m = c - 1 = 9 dimensions – but you still cannot visualize a 9-D dataset, right? So what do you do? Continue on Task 1.2.
    Deliverable:
    - 1) The module that implements FLD
    - 2) Denote the original MNIST dataset as X. Apply standardization on X such that each feature has a zero mean and standard deviation of 1. Denote this dataset as **nX**.

- 3) Apply W on nX. Denote the resulting dataset of reduced dimension (i.e., 9-D) as **fX.** Pick one image from each category and display them as 3x3 images; show both the original image and the 3x3 reduced image.

- Task 1.2: Implement the unsupervised dimensionality reduction method, PCA.
  Description: You should write a module with three input parameters: the MNIST dataset WITHOUT the label information, the desired error rate or the desired dimension. Note that in practice, sometimes, you have to reduce the dimension to a certain number regardless of the error rate. For instance, this is the case in visualization applications where the reduced dimension has to be between [1, 3]. You are suggested to implement the module such that if the second parameter is less than 1, then it refers to the error rate (assuming an error rate of 100% is not acceptable); if it is greater than or equal to 1, then it is the desired dimension. The outputs are the projection matrix, P, and the resulting error rate. Again, the second output helps you understand when you need to achieve the desired dimension, what the resulting error rate would be.
  Deliverable:
  - 4) The module that implements PCA
  - 5) Apply PCA on fX with the desired dimension being 2. Denote this dataset as **p2fX**. Report the resulting error rate. Display p2fX.
  - 6) Apply PCA on nX with the desired dimension being 2. Denote this dataset as **p2nX**. Report the resulting error rate. Display p2nX.
  - 7) Comment on if the visual illustrations of MNIST on the reduced 2-D space from FLD+PCA and PCA are different and why.
  - 8) Apply PCA on nX with the desired error rate being 10%. Denote the dataset as **pX**. Report the resulting dimension.
- Task 1.3: Impact of dimensionality reduction in classification.
  Description: In Project 2, you have compared the classification performance using both Baysian-based (parametric or non-parametric) and neural network-based methods on the original dataset, i.e., nX. In this task, you are to extend that comparison to the reduced dataset (fX and pX). The comparison should be reflected from two metrics, the classification accuracy and the computation time. Note that the computation time should include both training and testing time, separately reported if you can.
  Deliverables:
  - 9) Generate a plot of overall classification accuracy (using minimum Euclidean distance classifier) vs. error rate (from 5% to 30% with 5% increment using PCA)
  - 10) Generate a table (or bar chart or both) of 3x3 rows and 3 columns, with each set of 3 rows being a different classification method and each column being a

performance metric (i.e., classification accuracy, train time, test time). The 3 classification methods compared are minimum Euclidean distance classifier (Baysian parametric), kNN (k=15, Baysian non-parametric), BPNN (3 layers with the hidden layer containing 15 hidden nodes). For each of the classifiers, apply it on nX, fX, and pX. Note that this is probably the most time-consuming deliverable.

Note 1: You are strongly suggested to run the full version AFTER testing everything using a much smaller train/test set and BEFORE you go to bed and do this EARLY.

Note 2: You can use the published notebooks from previous projects.

Note 3: For kNN, you can only report one time which is training+testing.

- **Task 2: The magic of unsupervised learning.**

In this task, you will gain hands-on experience on unsupervised learning through an interesting application: image compression! Each pixel of a full-color (or true-color) image is composed of three channels, red, green, and blue. Each channel is represented as an 8-bit unsigned char. That is, each pixel uses 24 bits to display a color, a total of 2^{24} possible colors. Suppose the size of your image is 1024 x 1024, then the true-color image will take up about 24Mb storage space. One of the solutions to the image compression problem is use less number of bits to represent each pixel. This also carries practical value. For example, not all display devices can accommodate all 2^{24} different colors at one time. Hence one technique often used is to "group" the 2^{24} different colors into clusters, say 8 clusters; then you only need 3 bits to represent each pixel that amounts to 3Mb of storage per image. The color image not showing its full-color potential is referred to as the pseudo-color image. Think about exactly how the image compression problem can be solved using unsupervised learning and what are the features. Another bit of **heads-up** is that you'll be asked to learn and practice "auto-encoder" as a neural network-based unsupervised learning approach in the final project. We often use auto-encoder for estimating the missing information in the given dataset – the science of data imputation.

- Task 2.1: Implement kmeans (batch learning) and wta (online learning) with a given "k" value.

  Description: You need to implement kmeans and wta with the input parameters being the dataset and the assumed/given "k" value. The output is the set of cluster centers and assignment of each sample in the dataset. The performance metric we use to evaluate the unsupervised learning algorithm is usually the reconstruction

error [or root mean square error (RMSE)] between the reconstructed (i.e., pseudo-color image) and the original (i.e., the true-color image).

Deliverables:

- 11) The implemented kmeans module
- 12) The implemented wta module
- 13) Generate a scatter plot of the original color image in a 3-D RGB space.
- 14) Assume k=4, 16, 64, and 256, generate a table of 4 columns and 3 rows, with each of the 4 columns showing results by using a different k value. The first two rows in the table should display the resulting pseudo-color image, using kmeans and wta, respectively, along with the corresponding RMSE and run-time.
- 15) Draw the convergence curve (# of samples changing membership vs. # of epochs) for each of the two clustering algorithms. Use k=16.

- Task 2.2: Learn to use hierarchical clustering.

  Description: Use the hierarchical aggolomerative clustering routine provided in sklearn to solve the same image compression problem. Try different linkage options.

  Deliverables:

  - 15) Try to use the same 4 cluster numbers as in deliverable 14) and add the results to the last row of the table. Use d_{min}.
  - 16) Comment on the pros and cons among kmeans, wta, and the hierarchical method from the perspectives of image quality, run-time, and RMSE.
- Task 2.3 (Bonus): Learn to use hierarchical clustering for dimensionality reduction purpose.

  Description: Did you realize clustering can be used to group features?! … and that is essentially dimensionality reduction! You need to use MNIST in this case.

  Deliverables:

  - 17) Suppose with 10% error rate, PCA would reduce the dimension from 784 to "m". Apply SLINK to group features into m clusters. Show the dendrogram.
  - 18) Use the same network structure as in pX and compare the classification performance.

Reference:

[1] L. Maaten, G. Hinton, "Visualizing data using t-SNE," Journal of Machine Learning Research 9 (2008) 2579-2605.