CSE 474/574: Introduction to Machine Learning (Fall 2019)

Sargur N. Srihari
University at Buffalo, The State University of New York
Buffalo, New York 14260
Contact: 716-645-6162 (O), srihari@buffalo.edu

October 21, 2019

1 Task

The task of this project is to perform cluster analysis on fashion MNIST dataset using unsupervised learning. Cluster analysis is one of the unsupervised machine learning technique which doesn't require labeled data.

Your task will be that of clustering images and identify it as one of many clusters. You are required to train your unsupervised model using Fashion-MNIST clothing images. Following are the three tasks to be performed:

1. Use KMeans algorithm to cluster original data space of Fashion-MNIST dataset using Sklearns library.
2. Build an Auto-Encoder based K-Means clustering model to cluster the condensed representation of the unlabeled fashion MNIST dataset using Keras and Sklearns library.
3. Build an Auto-Encoder based Gaussian Mixture Model clustering model to cluster the condensed representation of the unlabeled fashion MNIST dataset using Keras and Sklearns library.

Report the clustering accuracy for each of the task.

2 Dataset

For training and testing of our clustering models, we will use the Fashion-MNIST dataset. The Fashion-MNIST is a dataset of Zalando's article images, consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image.

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 784 columns. You can import the Fashion MNIST dataset using keras library.

Each training and test example is assigned to one of the labels as shown in table 1. (The labels are only used during testing and not used during training)

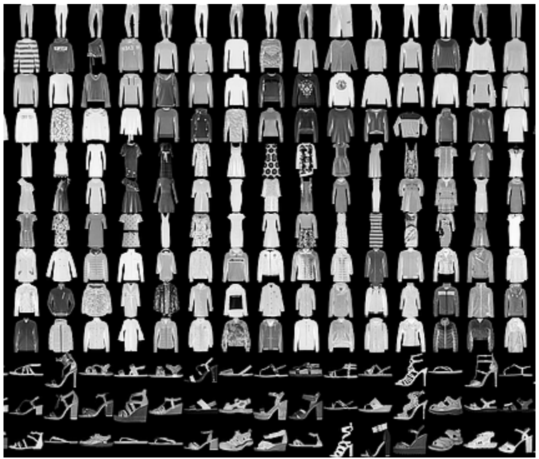

Figure 1: Example of how the data looks like.

1	T-shirt/top
2	Trouser
3	Pullover
4	Dress
5	Coat
6	Sandal
7	Shirt
8	Sneaker
9	Bag
10	Ankle Boot

Table 1: Labels for Fashion-MNIST dataset

.

3 K-Means

The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares (see below). This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields.

The k-means algorithm divides a set of samples into disjoint clusters, each described by the mean of the samples in the cluster. The means are commonly called the cluster centroids; note that they are not, in general, points from set of samples, although they live in the same space.

4 Auto Encoder

"Autoencoding" is a data compression algorithm where the compression and decompression functions are data-specific, lossy, and learned automatically from examples rather than engineered by a human. Autoencoder uses compression and decompression functions which are implemented with neural networks as shown in Figure 2.

To build an autoencoder, you need three things: an encoding function, a decoding function, and a distance function between the amount of information loss between the compressed representation of

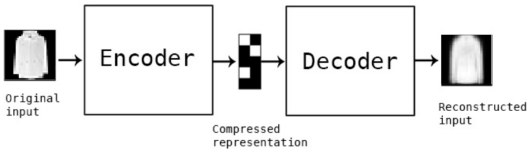

Figure 2: AutoEncoder

your data and the decompressed representation (i.e. a ”loss” function). The encoder and decoder will be chosen to be parametric functions (typically neural networks), and to be differentiable with respect to the distance function, so the parameters of the encoding/decoding functions can be optimize to minimize the reconstruction loss, using Stochastic Gradient Descent.

4.1 Auto-Encoder with K-Means Clustering

The K-means algorithm aims to choose centroids that minimise the inertia, or within-cluster sum-of-squares criterion:

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2) \tag{1}$$

Inertia can be recognized as a measure of how internally coherent clusters are. Inertia is not a normalized metric: we just know that lower values are better and zero is optimal. But in very high-dimensional spaces, Euclidean distances tend to become inflated (this is an instance of the so-called curse of dimensionality). Running a dimensionality reduction algorithm such as Principal component analysis (PCA) or Auto-encoder prior to k-means clustering can alleviate this problem and speed up the computations.

4.2 Auto-Encoder with GMM Clustering

A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. One can think of mixture models as generalizing k-means clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians.

The GaussianMixture object implements the expectation-maximization (EM) algorithm for fitting mixture-of-Gaussian models. It can also draw confidence ellipsoids for multivariate models, and compute the Bayesian Information Criterion to assess the number of clusters in the data. A GaussianMixture.fit method is provided that learns a Gaussian Mixture Model from train data. Given test data, it can assign to each sample the Gaussian it mostly probably belong to using the GaussianMixture.predict method.

5 Plan of Work

1. **Extract feature values:** Fashion MNIST dataset is downloaded and processed into a Numpy array that contains the feature vectors and a Numpy array that contains the labels.

2. **K-Means Clustering:** Use Sklearns library to cluster Fashion MNIST dataset into 10 clusters. Report the clustering accuracy.

3. **Train using Auto-Encoder Network:** Use Keras library to build a Auto-Encoder Network.

4. **Create K-Means clustering layer:** By training the auto-encoder, we have the encoder learned to compress each image into latent floating point values. We are going to use K-Means to generate the cluster centroids, which is the 10 cluster centers in the latent feature space. We then cluster compressed latent codes produced by AutoEncoder using K-Means.

5. **Create GMM clustering layer:** Similar to Step 4, we are going to use Gaussian Mixture Model to generate the cluster centroids, which are the 10 cluster centers in the latent feature space (Latent space provided by pre-trained encoder network)

6. **Test your machine learning scheme:** Create a confusion matrix and report the clustering accuracy by matching the clustering assignment for step 2, 4 and 5.

6 Evaluation

1. Report the clustering accuracy with K-Means as the baseline model.

2. Plot graph of training loss and validation loss vs number of epochs while training for auto-encoder.

3. Construct a confusion matrix for Auto-Encoder based K-Means clustering prediction and report the clustering accuracy.

4. Construct a confusion matrix for Auto-Encoder based GMM clusteing prediction and report the clustering accuracy.

7 Deliverables

There are two deliverables: report and code. After finishing the project, you may be asked to demonstrate it to the TAs, particularly if your results and reasoning in your report are not clear enough.

1. Report (30 points)

   The report should be in NIPS format. The report should have: Abstract, Introduction, Dataset, Architecture, Results and Conclusion. Submit the PDF on a CSE student server with the following script:

   `submit_cse474 proj3.pdf` for undergraduates

   `submit_cse574 proj3.pdf` for graduates

2. Code (70 points)

   The code for your implementation should be in Python only. You can submit multiple files, but the name of the entrance file should be main.ipynb. Please provide necessary comments in the code. Python code and data files should be packed in a ZIP file named `proj3code.zip`. Submit the Python code on a CSE student server with the following script:

   `submit_cse474 proj3code.zip` for undergraduates

   `submit_cse574 proj3code.zip` for graduates