National University of Singapore
School of Computing
CS3244: Machine Learning
Solution to Tutorial 10

Unsupervised Learning

1. ***K*-means convergence:** We have learned the iterative algorithm for K-means. Let's look into the algorithm again.

Algorithm 1 K-means Clustering

```
1: for $c = 1$ to $k$ do
2:     $\mu_c \leftarrow$ some random location
3: while Not Converged do
4:     for $j = 1$ to $m$ do
5:         $y^{(j)} \leftarrow$ c $=$ arg min$_c \|x^{(j)} - \mu_c\|^2$        ▷ A. assign example
6:         $x^{(j)} \leftarrow S_c$
7:     for $c = 1$ to $k$ do
8:         $\mu_c \leftarrow \frac{1}{|S_c|} \sum_{x \in S_c} x$        ▷ B. re-estimate center
9: return $y$
```

The algorithm leads to convergence when the clustering quality L_{clust} is minimized (*i.e.*, when cluster centers or assignments stop changing).

$$L_{clust} = \sum_{c=1}^{k} \sum_{x \in S_c} \|x - \mu_c\|^2 \tag{1}$$

(a) Show that each data assignment step (Line 5 in Algorithm 1) minimizes L_{clust}, given fixed cluster centers.

$$\begin{aligned} L_{clust} &= \sum_{c=1}^{k} \sum_{x^{(j)} \in S_c} \|x^{(j)} - \mu_c\|^2 \\ &= \sum_{j=1}^{m} \|x^{(j)} - \mu_{assigned}\|^2 \\ &= \sum_{j=1}^{m} L_{x^{(j)}} \end{aligned} \tag{2}$$

To minimize L_{clust}, we minimize the error for each data ($L_{x^{(j)}}$) independently since cluster centers are fixed. For each data, error is minimized if we assign it to the nearest cluster. Hence Line 5 minimizes L_{clust}.

(b) Show that each cluster center update step (Line 8 in Algorithm 1) minimizes L_{clust}, given fixed data assignments.

$$\begin{aligned} L_{clust} &= \sum_{c=1}^{k} \sum_{x \in S_c} \|x - \mu_c\|^2 \\ &= \sum_{c=1}^{k} L_c \qquad (3) \end{aligned}$$

where $L_c = \sum_{x \in S_c} \|x - \mu_c\|^2$ represents the error for each cluster. Since assignments are fixed, we can optimize for each cluster independently. To minimize L_c, we update the cluster centers to the mean of all data assigned to the cluster. This is very intuitive but can be proven using simple math.

$$\begin{aligned} L_c &= \sum_{x \in S_c} \|x - \mu_c\|^2 \\ &= \sum_{x \in S_c} \sum_{d} (x_d - \mu_{c,d})^2 \qquad (4) \end{aligned}$$

In the above equation, $x = \{x_1, x_2, ..., x_n\}$ and $\mu_c = \{\mu_{c,1}, \mu_{c,2}, ..., \mu_{c,n}\}$. Hence, $\mu_{c,d}$ indicates d-th element of vector μ_c. L_c is minimized when derivatives with respect to $\mu_{c,d}$ equals zero. Let's calculate the derivative first.

$$\begin{aligned} \frac{\partial L_c}{\partial \mu_{c,d}} &= \sum_{x \in S_c} \frac{\partial}{\partial \mu_{c,d}} (x_d - \mu_{c,d})^2 \\ &= \sum_{x \in S_c} -2(x_d - \mu_{c,d}) \qquad (5) \end{aligned}$$

If L_c is minimized, this derivative equates to zero.

$$\begin{aligned} \frac{\partial L_c}{\partial \mu_{c,d}} &= 0 \\ \sum_{x \in S_c} (x_d - \mu_{c,d}) &= 0 \\ \left(\sum_{x \in S_c} x_d \right) - m_c \mu_{c,d} &= 0 \\ \mu_{c,d} &= \frac{1}{m_c} \sum_{x \in S_c} x_d \qquad (6) \end{aligned}$$

where m_c is the number of data being assigned to the cluster c. It can be further shown using second derivatives that this is a minimum point.

(c) Despite the guarantee of convergence, the clustering result varies depending on initialization of the centroids. Sub-optimal clustering may also occur when the clusters are of different sizes and densities. Suggest two solutions to overcome this issue.

- *Trying multiple initializations and choosing one with minimum L_{clust}.*
- *Increasing k, and then manually merging clusters after k-Means.*

- *The performance of k-Means is sensitive to the centroid initialization, which can be improved by k-Means++. k-Means++ is shown below (credit to Wikipedia, not required for examination).*

  i. Choose one centroid uniformly at random among the data points.

  ii. For each data point x not chosen yet, compute $D(x)$, the distance between x and the nearest existing centroid.

  iii. Choose one new data point at random as a new centroid, using a weighted probability distribution where a point x is chosen with probability proportional to $D(x)^2$.

  iv. Repeat Steps ii and iii until k centroids have been chosen.

  v. Now the initial centroids are chosen, proceed using standard k-means clustering.

2. **Auto-Encoder**: An auto-encoder is a neural network that is trained to attempt to reconstruct the input to the output. Autoencoder has an encoder E and a decoder D. The encoder $\mathbf{z} = E(\mathbf{x})$ consumes the input $\mathbf{x}$ and produce an intermediate representation $\mathbf{z}$ which is fed into the decoder $\hat{\mathbf{x}} = D(\mathbf{z})$ to reconstruct the input. Figure 1 shows this.

   (a) Given a large image dataset without label(dataset P) and a small image dataset(dataset Q) with label, both datasets come from the same domain. Propose a way to improve classifier's performance on the small dataset.

   Pre-train a deep convolutional autoencoder on the dataset, P with reconstruction loss. Figure 1 shows this. Discard the decoder of the autoencoder, attach a simple classifier after the encoder, and fine-tune the neural network on the dataset Q with classification loss as shown Figure 2

   Deep convolutional networks are the SOTA models for image classification. However, it is hard to train deep convolutional networks on small dataset because of the limited supervision. Thus, we train an convolutional autoencoder with reconstruction loss on the dataset P. In this way, the convolutional encoder can learn to extract important visual feature and eliminate the noise in images. With this pre-trained convolutional encoder, it would be easier to train a classifier with the dataset Q than training a classifier from scratch.

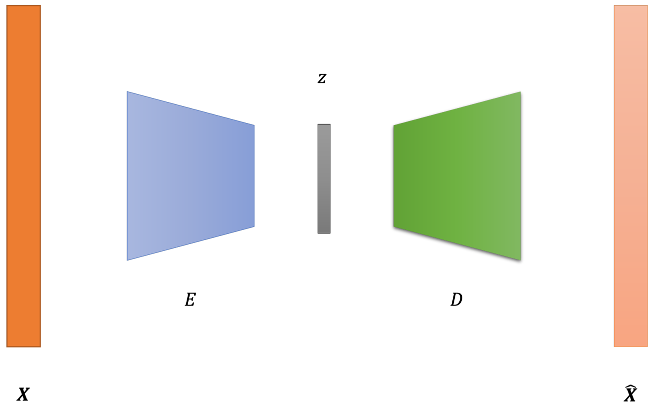

Figure 1: Autoencoder

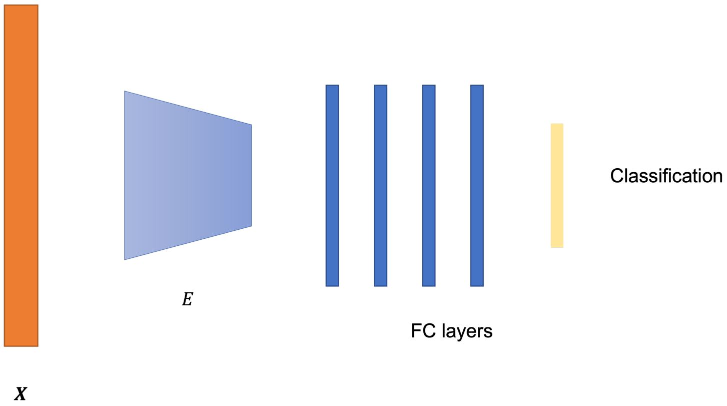

Figure 2: Classifier

(b) Autoencoder can compress the original input into a lower dimensional encoding. However, it is hardly used in practice for image compression. List the disadvantages of autoencoder when used for compression.

Lossy. Autoencoder loses pixel information because it aims to discard the unimportant features and retain only important features. On the contrary, modern image format like PNG can achieve lossless compression.

Data Dependent. Autoencoder works well only if the input comes from the same distribution of the training set. If the input is out of distribution, the recovered image quality will be very poor.

Despite the disadvantages, autoencoder is an important model for being unsupervised. Unsupervised learning is an important research field because it enables machine learning models to leverage the large amount of cheap unlabeled data on internet. The idea of reconstructing the input is further developed and inspired the SOTA unsupervised deep learning models, including GPT (language) and CLIP (vision + language).