Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 4

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 17, 2018

QUESTION 1

We are given that matrix X has size $N * D$ where $D > N$. We are told to consider a case when we are given an eigenvector v of the matrix $\frac{1}{N}XX^T$ and we need to compute the eigenvector u for $S = \frac{1}{N}X^TX$

$$\frac{1}{N}XX^Tv = \lambda v$$

Left multiply both sides by X^T

$$\frac{1}{N}X^TX(X^Tv) = \lambda(X^Tv)$$

Observing the above equation, we can see that eignvector u of S is X^Tv. So, we can use the eigenvector v to get the eigenvector u for S.

Advantage :
Here, $\frac{1}{N}XX^T$ is an $N * N$ matrix. On the other hand, $\frac{1}{N}X^TX$ is a $D * D$ matrix. The cost of eigendecomposition for an $N * N$ matrix is $\mathcal{O}(N^3)$. Therefore, the cost of doing eigendecomposition for S is $\mathcal{O}(D^3)$. But if we use v to compute the eigenvectors for S, then cost of eigendecomposition for $\frac{1}{N}XX^T$ is $\mathcal{O}(N^3)$ plus we need to add a cost equal to $\mathcal{O}(DN^2)$ for getting N eigenvectors u from v using X^Tv. Therefore, total cost in this case is $\mathcal{O}(N^3 + DN^2)$ which is lesser in comparison to $\mathcal{O}(D^3)$ given that $D > N$.

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 4

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 17, 2018

QUESTION 2

We are given an activation function

$$h(x) = x\sigma(\beta x)$$

where σ denotes the sigmoid function

$$\sigma(z) = \frac{1}{1 + exp(-z)}$$

Case 1: Linear Activation Function

$h(x)$ has the following form

$$h(x) = \frac{x}{1 + exp(-\beta x)}$$

For positive values of x, consider β to have a large positive value (tending to ∞) and for negative values of x, consider β to have a large negative value (tending to $-\infty$). If we choose the value of β as per the above constraints, then the function $h(x)$ can approximate the linear activation function.

Case 2: ReLu Activation Function

$h(x)$ has the following form

$$h(x) = \frac{x}{1 + exp(-\beta x)}$$

In this case, consider β to have a large positive value (tending to ∞). For positive value of x, the function $h(x)$ approximates to x and for negative value of x, $h(x)$ goes to zero. Therefore, if we choose the value of β as per the above constraint, then the function $h(x)$ can approximate the ReLu activation function.

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 4

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 17, 2018

QUESTION 3

Marginal probability of the label $y_n = 1$, given x_n i.e. $p(y_n = 1|x_n)$ is as follows:

$$p(y_n = 1|x_n) = \sum_{k=1}^{K} \pi_k \sigma(w_{z_{nk}}^T x_n)$$

This quantity can be thought of as the output of a neural network with the following specifications :

- Input layer will consist of D nodes representing D features of each input instance x_n
- The network will consist of a single hidden layer consisting of K nodes, each node representing z_i, $i \in 1, .., K$
- The input layer and the hidden layer are fully connected with each input instance x_n connected to all the nodes in the hidden layer, i.e. x_n is connected to each z_i, $i \in 1, .., K$.
- The weights between the input layer and the hidden layer will have the form w_{z_n} where it is the weight matrix corresponding to the connection between x_n and z_n
- Hidden nodes have sigmoid activation applied over them.
- The connection weights between the hidden layer and the output layer are the $\pi_k' s$

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 4

QUESTION 4

Student Name: Sahil Bansal
Roll Number: 150614
Date: November 17, 2018

We are supposed to consider an $N * M$ matrix X, where the rows represents the N users and the columns represent the M items.
The parameters that we are supposed to estimate are as follows:

$$\Theta = \{u_n, \theta_n\}_{n=1}^{N}, \{v_m, \phi_m\}_{m=1}^{M}, W_u, \text{and } W_v$$

The loss function for the model will be as follows :

$$\mathcal{L}(\Theta) = \left(\prod_{n,m \in \Omega} P(X_{nm} | u_n, v_m, \theta_n, \phi_m) \right) \left(\prod_{n=1}^{N} P(u_n) \right) \left(\prod_{m=1}^{M} P(v_m) \right)$$

Taking the negative log-likelihood,

$$NLL = -\log(\mathcal{L}(\Theta)) = -\sum_{n,m \in \Omega} \log(P(X_{nm} | u_n, v_m, \theta_n, \phi_m)) - \sum_{n=1}^{N} \log(P(u_n)) - \sum_{m=1}^{M} \log(P(v_m))$$

$$= \frac{1}{2} \sum_{n,m \in \Omega} \lambda_x (X_{nm} - (\theta_n + \phi_m + u_n^T v_m))^2 + \frac{1}{2} \sum_{n=1}^{N} \lambda_u ||u_n - W_u a_n||^2 + \frac{1}{2} \sum_{m=1}^{M} \lambda_v ||v_m - W_v b_m||^2 + K$$

ALT-OPT Algorithm :
1. Initialize Θ except $\{\phi_m\}_{m=1}^{M}$, i.e. $\{u_n, \theta_n\}_{n=1}^{N}, \{v_m\}_{m=1}^{M}, W_u, \text{and } W_v$ as $\Theta^{(0)}$ and set $t = 1$
2. Update equations for the parameters can be obtained by taking the derivative of NLL w.r.t them. First, finding the update equation for ϕ_m, $m = 1, .., M$ by taking the derivative of NLL w.r.t ϕ_m.

$$\phi_m = \frac{1}{|\Omega_{c_m}|} \sum_{n \in \Omega_{c_m}} \lambda_x (X_{nm} - \theta_n - u_n^T v_m)$$

3. Similarly, the update equation for θ_n, $n = 1, .., N$ can be obtained by taking the derivative of NLL w.r.t θ_n.

$$\theta_n = \frac{1}{|\Omega_{r_n}|} \sum_{m \in \Omega_{r_n}} \lambda_x (X_{nm} - \phi_m - u_n^T v_m)$$

4. Update equation for W_u by taking partial derivative of NLL w.r.t W_u is as follows :

$$\sum_{n=1}^{N} 2 W_u a_n^T a_n - \sum_{n=1}^{N} 2 u_n a_n^T = 0$$

$$W_u = \left(\sum_{n=1}^{N} u_n a_n^T \right) \left(\sum_{n=1}^{N} a_n a_n^T \right)^{-1}$$

5. Similarly, update equation for W_v

$$W_v = \left(\sum_{m=1}^{M} v_m b_m^T \right) \left(\sum_{m=1}^{M} b_m b_m^T \right)^{-1}$$

6. For the update equation of u_n, take the derivative of NLL w.r.t u_n.

$$-\sum_{m\in\Omega_{r_n}} \lambda_x v_m(X_{nm} - (\theta_n + \phi_m + u_n^T v_m)) + \lambda_u(u_n - W_u a_n) = 0$$

Therefore, u_n, $n = 1, .., N$ is as follows :

$$u_n = \left(\sum_{m\in\Omega_{r_n}} \lambda_x v_m v_m^T + \lambda_u \mathbf{I}\right)^{-1} \left(\sum_{m\in\Omega_{r_n}} \lambda_x (X_{nm} - \theta_n - \phi_m) v_m + \lambda_u W_u a_n\right)$$

7. Similarly, update equations for v_m, $m = 1, .., M$ is as follows :

$$v_m = \left(\sum_{n\in\Omega_{c_m}} \lambda_x u_n u_n^T + \lambda_v \mathbf{I}\right)^{-1} \left(\sum_{n\in\Omega_{c_m}} \lambda_x (X_{nm} - \theta_n - \phi_m) u_n + \lambda_v W_v b_m\right)$$

8. Set t = t + 1 and go to step 2 if not yet converged

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 4

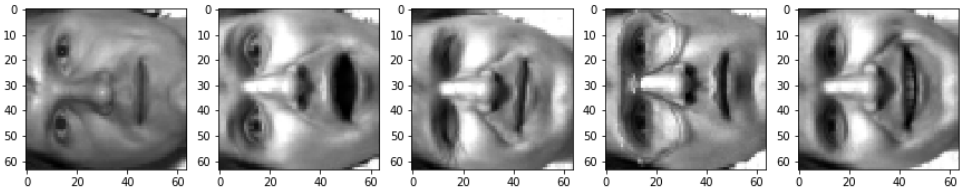
Student Name: Sahil Bansal
Roll Number: 150614
Date: November 17, 2018

QUESTION

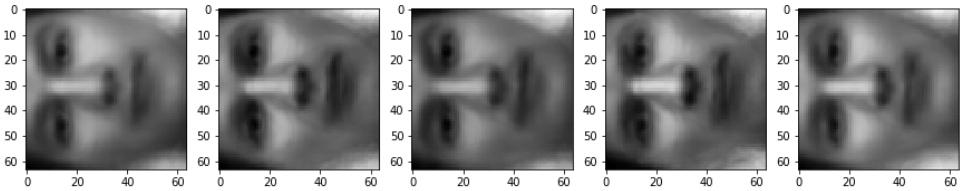
5

Programming Problem 1:
For k = 10,

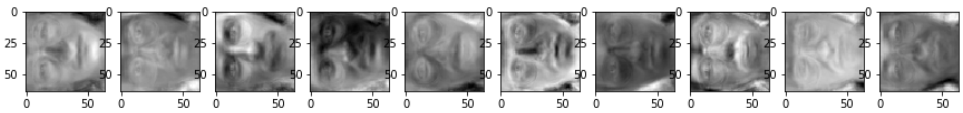
- Original Images



- Recreated Images

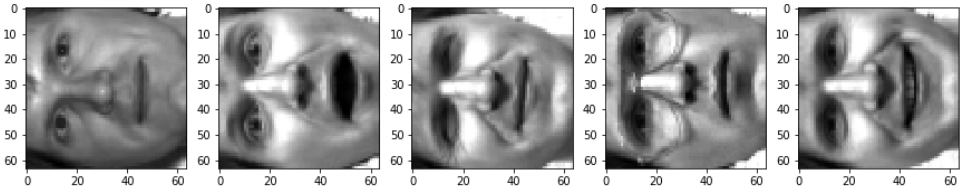


- Basis Images

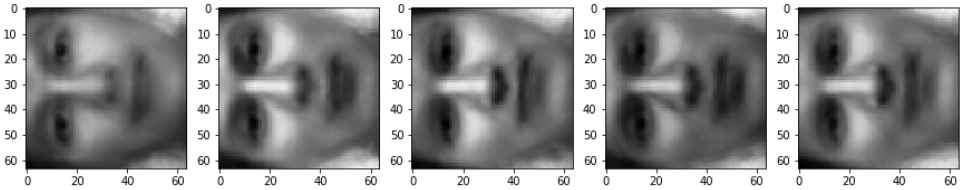


For k = 20,

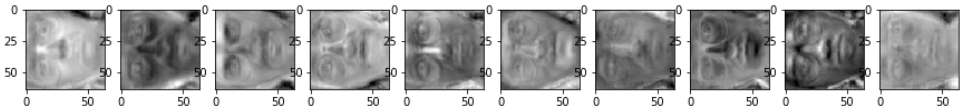
- Original Images



- Recreated Images

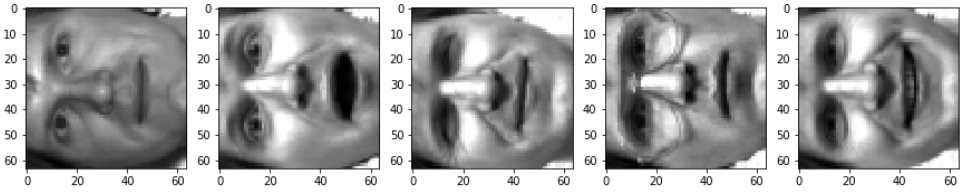


- Basis Images

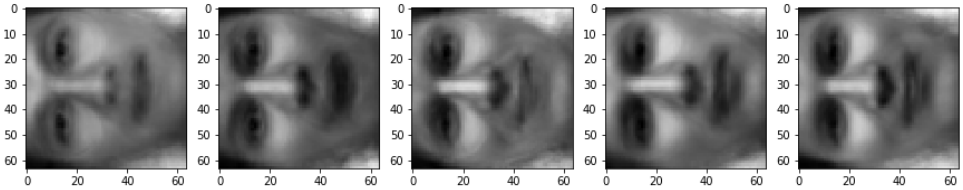


For k = 30,

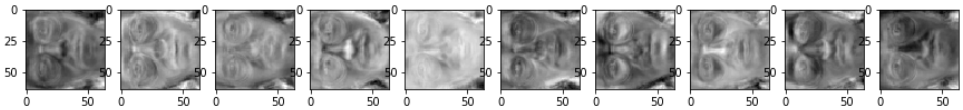
- Original Images



- Recreated Images

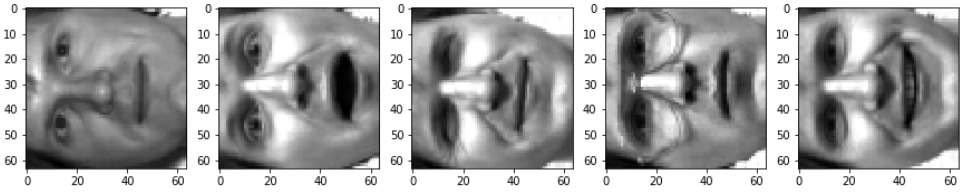


- Basis Images

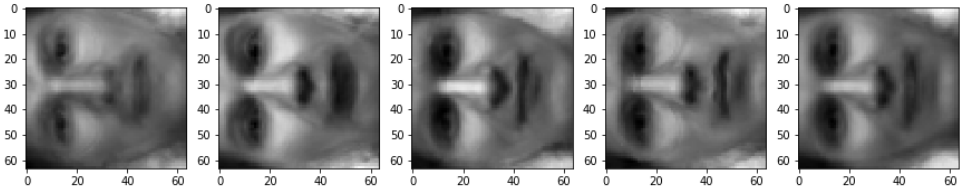


For k = 40,

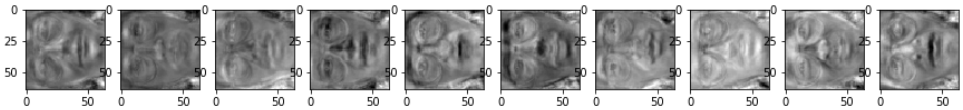
- Original Images



- Recreated Images

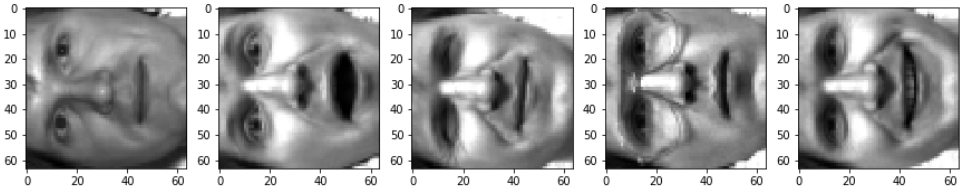


- Basis Images

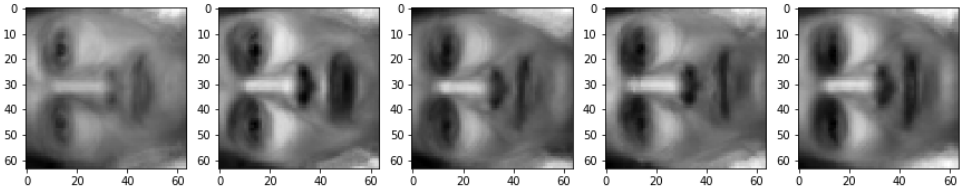


For k = 50,

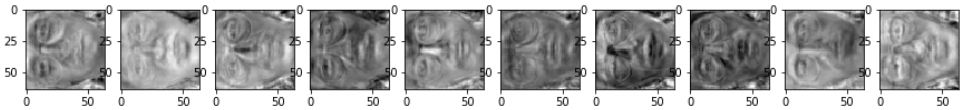
- Original Images



- Recreated Images

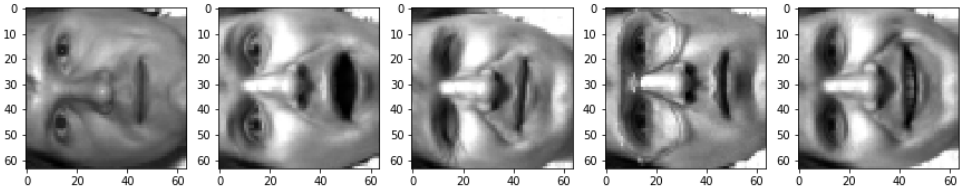


- Basis Images

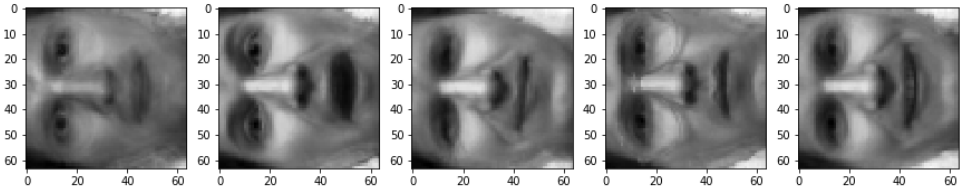


For k = 100,

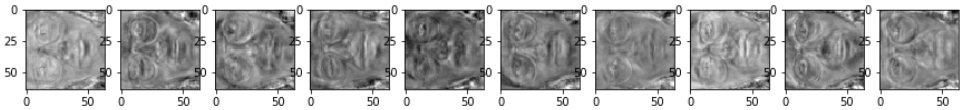
- Original Images



- Recreated Images



- Basis Images



Observations :

As the value of K increases, the reconstruction gets better. The latent feature z_n that is used for the reconstruction of x_n is K-dimensional. Therefore, as the value of K increases, the dimensionality of the latent feature also increases. In dimensionality reduction as the value of K gets smaller and smaller, we are trying to represent x_n using latent feature of smaller dimensionality. So, the loss of information increases as the value of K gets smaller and smaller or we can say this in the other manner, that as the value of K increases, the loss of information gets smaller and smaller and z_n can better approximate x_n. This is exactly what is also observed visually that as the value of K increases, reconstruction gets better and better.

Programming Problem 2:

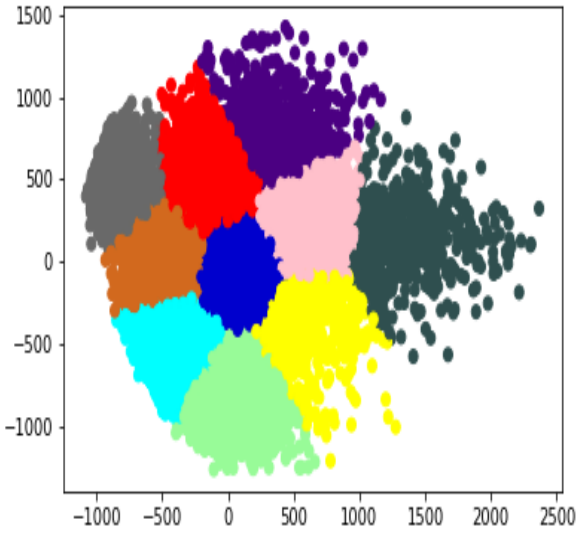


Figure 1: PCA-1

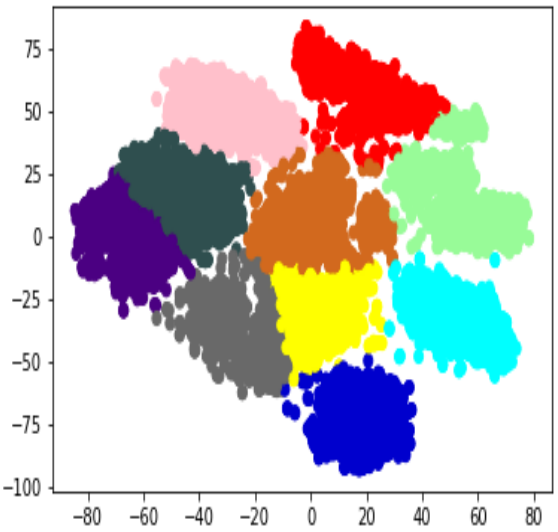


Figure 2: tSNE-1

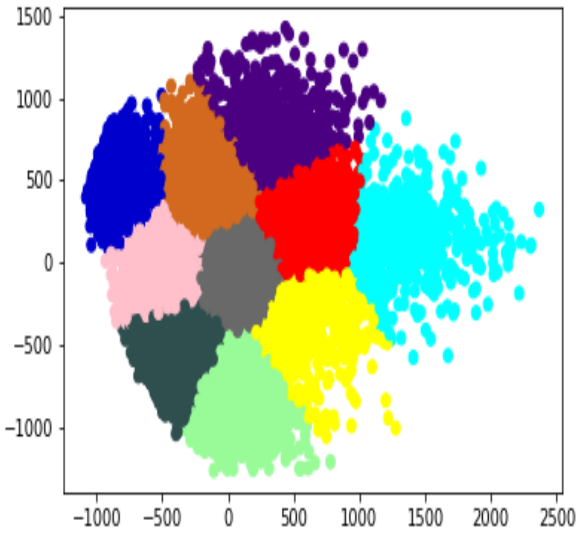


Figure 3: PCA-2

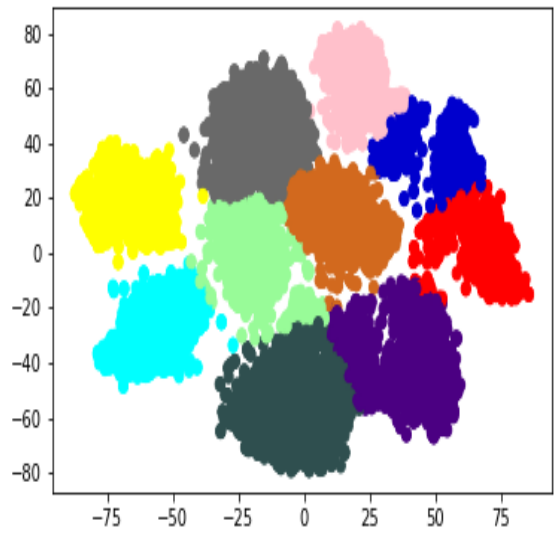


Figure 4: tSNE-2

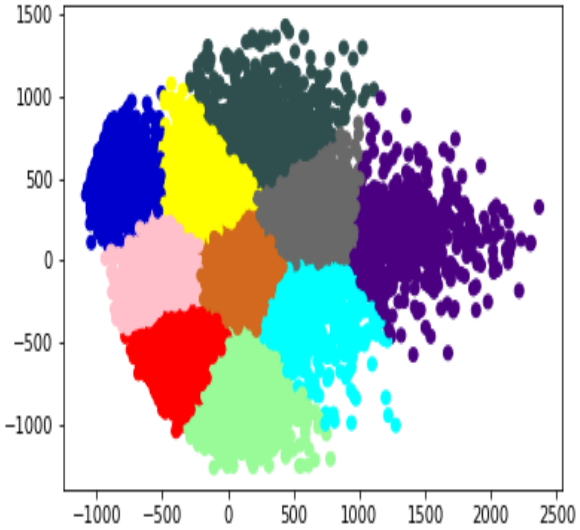


Figure 5: PCA-3

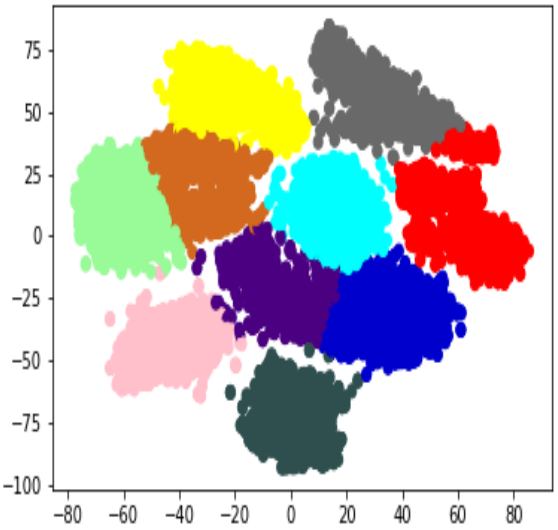


Figure 6: tSNE-3

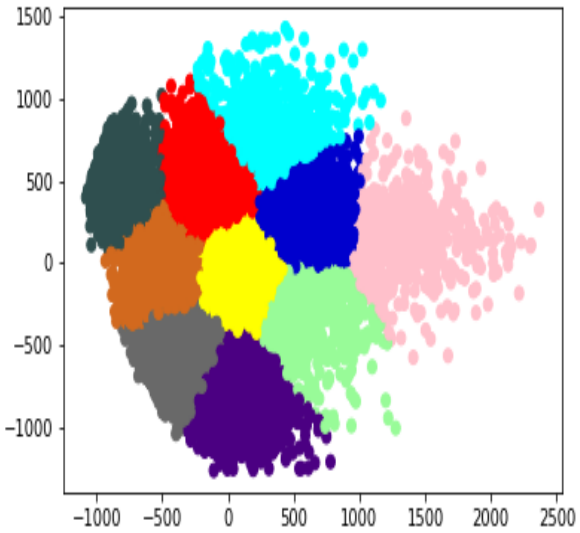


Figure 7: PCA-4

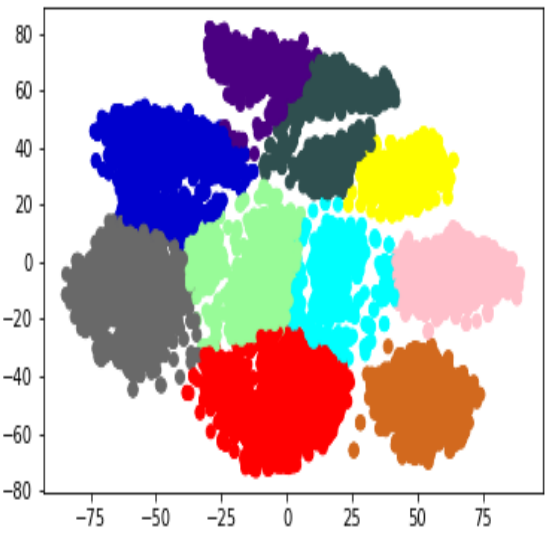


Figure 8: tSNE-4

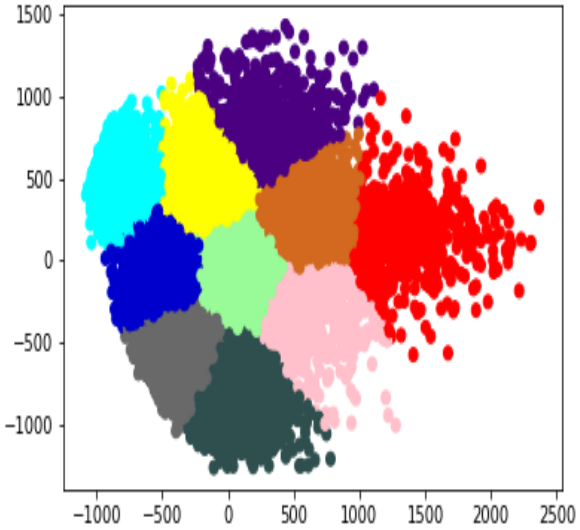


Figure 9: PCA-5

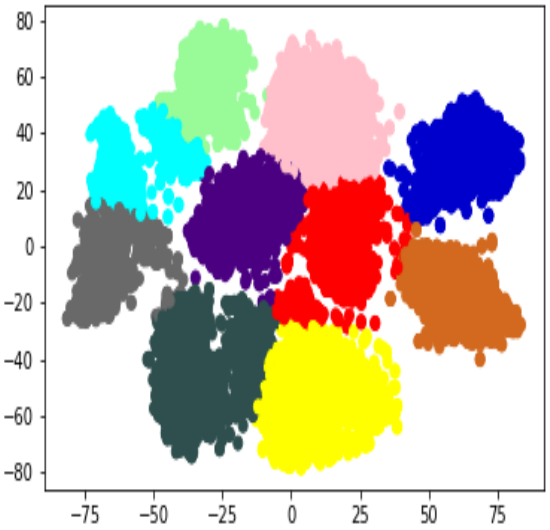


Figure 10: tSNE-5

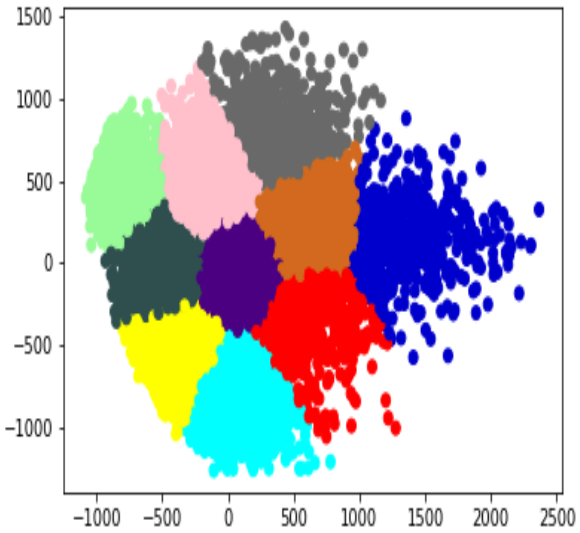


Figure 11: PCA-6

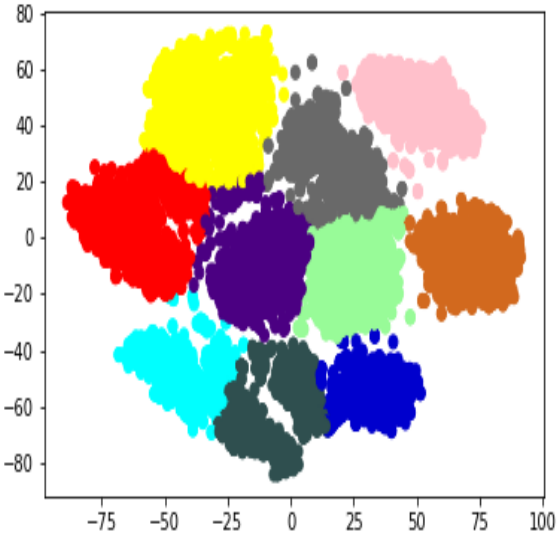


Figure 12: tSNE-6

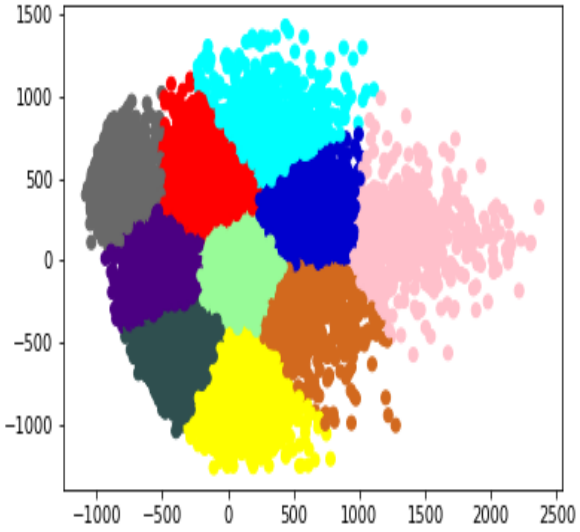


Figure 13: PCA-7

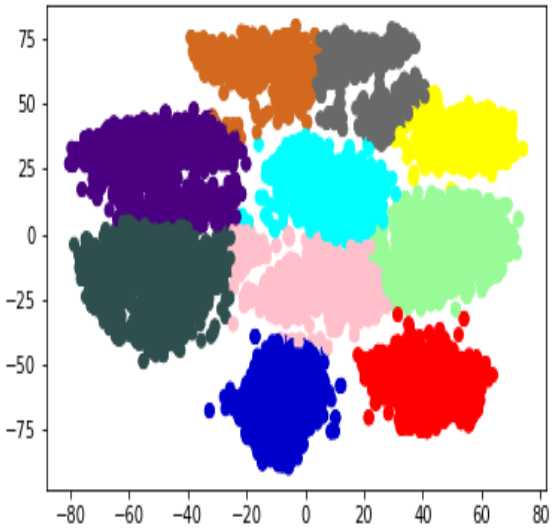


Figure 14: tSNE-7

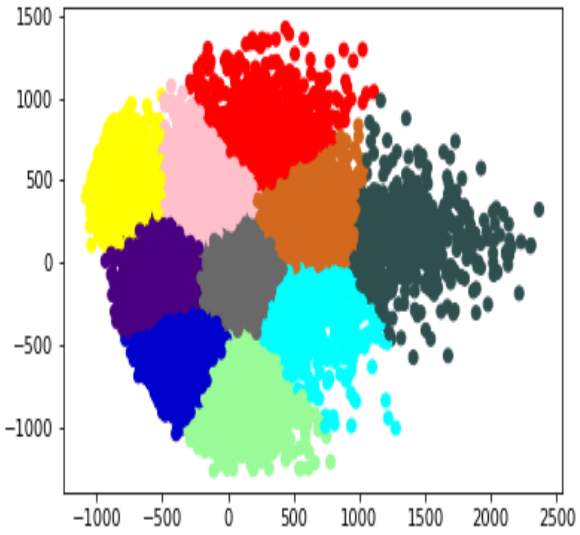


Figure 15: PCA-8

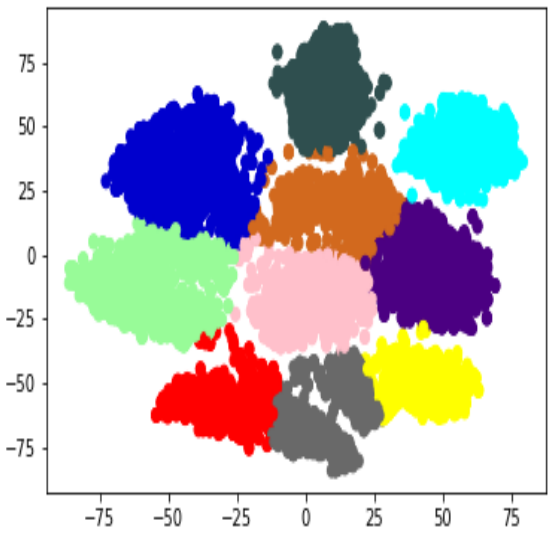


Figure 16: tSNE-8

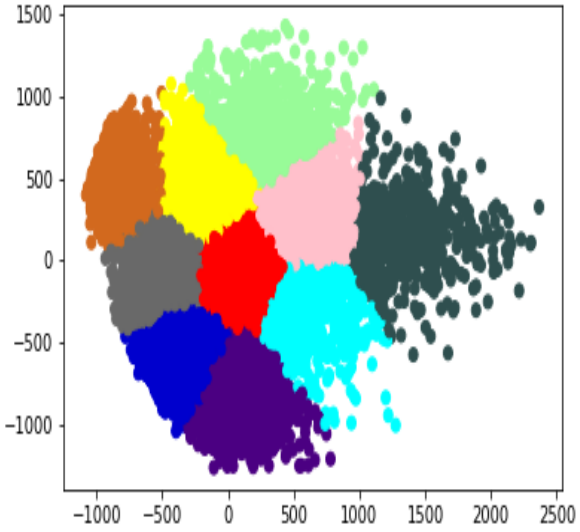


Figure 17: PCA-9

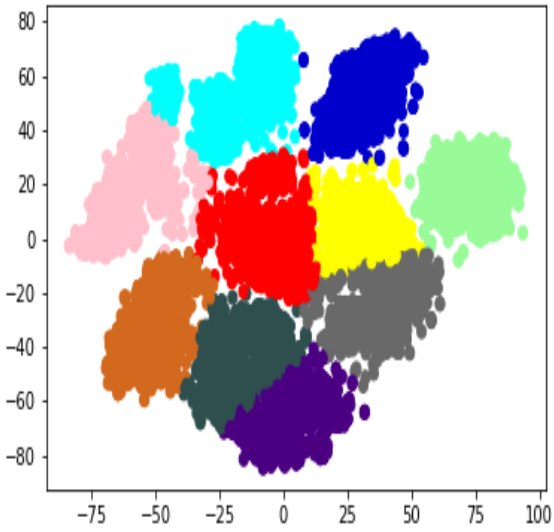


Figure 18: tSNE-9

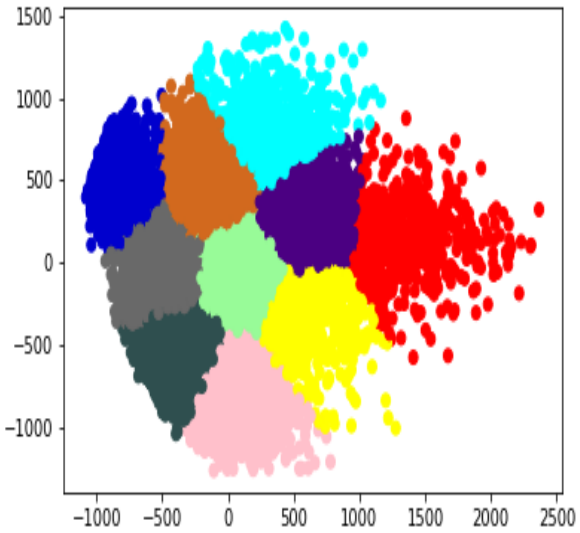


Figure 19: PCA-10

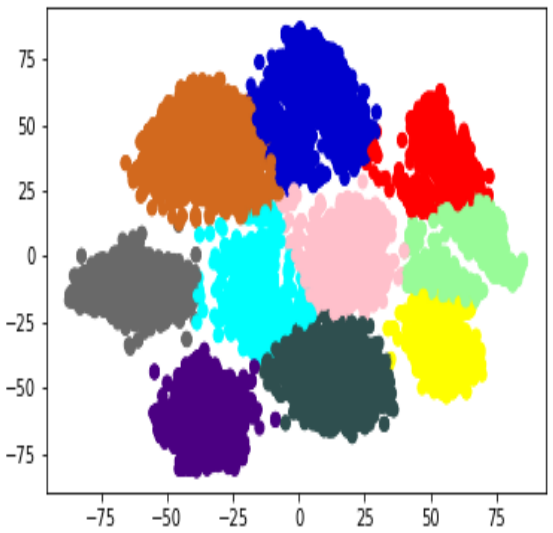


Figure 20: tSNE-10

Observation :

tSNE seems to work better in comparison to PCA as we can see that the clusters are well separated out in case of tSNE whereas this is not the case with PCA based 2D embeddings.