Prob 1

let's assume we have a eigenvector $\underline{v}_i$ of

Matrix $\underline{S} = \frac{1}{N} \underline{X}\underline{X}^T$

$\therefore$ It will satisfy - $\underline{S}\underline{v} = \lambda \underline{v}$

λ ↳ eigenvalue.

$$\Rightarrow \frac{1}{N}(\underline{X}\underline{X}^T)\underline{v} = \lambda \underline{v}$$

$$\Rightarrow \frac{1}{N}(\underline{X}^T\underline{X})(\underline{X}^T\underline{v}) = \lambda(\underline{X}^T\underline{v})$$ {pre multiply X^T both side}

Substitute $\underline{u} = X^T\underline{v}$

$$\Rightarrow \frac{1}{N}(\underline{X}^T\underline{X})\underline{u} = \lambda \underline{u}$$

$\therefore$ $\underline{u} = X^T\underline{v}$ is nothing else but an eigenvector for $\frac{1}{N}(X^TX)$

In normal case to compute K eigenvectors for $\frac{1}{N}(X^TX)$ complexity will be $O(KD^2)$

but in this case order will be $O(KN^2) + O(KND)$

↓ for decomp. of $\frac{1}{N}(XX^T)$

↓ for multi matrix

$\therefore$ overall complexity $O(KND)$

↓ which is less than $O(KD^2)$

as $\boxed{N < D}$

Prob 2

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

if we choose $\beta = 0$

$\Rightarrow \sigma(0) = 1$ ~~$\Rightarrow$ a~~ a linear activation function.

$\Rightarrow h(x) = \frac{x}{2}$ $\Rightarrow$

if we chose $\beta \to \infty$

$\Rightarrow$ for $x < 0 \Rightarrow \sigma(\beta x) = \sigma(-\infty) = 0$

& for $x > 0 \Rightarrow \sigma(\beta x) = \sigma(\infty) = 1$

$$\Rightarrow h(x) = \begin{cases} x & ; x > 0 \\ 0 & ; x < 0 \end{cases}$$

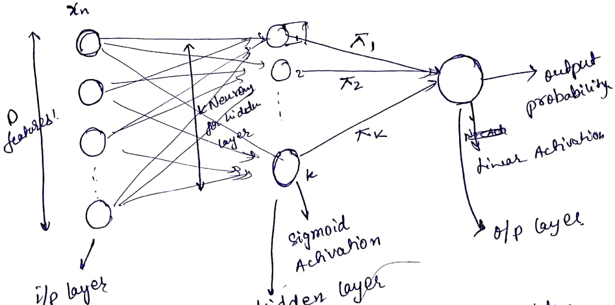
$\rightarrow$ RELU $\downarrow$ Activation f.n

Prob 3

$$P(y_n=1 \mid x_n) = \prod_{k=1}^{K} P(y_n=1 \mid z_n = k, x_n)\, P(z_n=k \mid x_n)$$

∴ Above equation can be modelled as a multiplication of two probabilistic fn(s).

⇒ 2 layer NN with one hidden layer.



∴ basically $\sigma(w_{zn}^T x_n)$ → acts as an hidden neurons.

Activation is sigmoids.

∴ Weights for first layer will be $(w_{z1}, w_{z2} \dots w_{zk})$ for k^{th} neurons.

∴ Weight for outputs will be $[\pi_1, \pi_2 \dots \pi_K]^T$

multiplying the probability of $P(z_n \mid x_n)$

Prob 4

$$P(u_n, v_m, \theta_n, \phi_m \mid X)$$

$$= \prod_{n=1, m=1}^{M,N} p(x_{nm} \mid u_n, v_m, \theta_n, \phi_m)\, p(u_n)\, p(v_m)$$

$\Rightarrow 0$

$$= \prod_{n,m} \sqrt{\frac{\lambda_x}{2\pi}} \exp\left\{-\frac{\lambda_x}{2}\left(x_{nm} - \theta_n - \phi_m - u_n^T v_m\right)^2\right\}$$

$$\times \sqrt{\frac{\lambda_u}{2\pi}} \exp\left(-\frac{\lambda_u}{2}\|u_n - W_u a_n\|^2\right)$$

$$\times \sqrt{\frac{\lambda_v}{2\pi}} \exp\left(-\frac{\lambda_v}{2}\|v_m - W_v a_m\|^2\right)$$

$$\Rightarrow NLL = \sum_{n,m}\left\{\frac{\lambda_x}{2}\left(x_{nm} - \theta_n - \phi_m - u_n^T v_m\right)^2\right\}$$

$$+ \sum_n \left\{\frac{\lambda_u}{2}\|u_n - W_u a_n\|^2\right\}$$

$$+ \sum_m \left\{\frac{\lambda_v}{2}\|v_m - W_v a_m\|^2\right\}$$

NLL → cost function

Now, for W_u, W_v → minimization will be similar to linear regression model.

which will give :-

$$W_u^T = (A^T A)^{-1} A^T U .$$

where; $A \rightarrow N \times D$ matrix of all a_n's

$U \rightarrow N \times K$ matrix of all u_n's

$$\Rightarrow \quad W_u = U^T A (A^T A)^{-1}$$

Similarly, $W_v = V^T B (B^T B)^{-1}$

for θ_n & ϕ_m | taking the derivatives yield.

$$\sum_m -\lambda_x (x_{nm} - \theta_n - \phi_m - u_n^T v_m) = 0$$

$$\Rightarrow \quad \theta_n = \frac{1}{M} \sum_m (x_{nm} - \phi_m - u_n^T v_m)$$

Similarly $\phi_m = \frac{1}{N} \sum_n (x_{nm} - \theta_n - u_n^T v_m)$

Now taking the derivative w.r.t. v_m we get.

$$\sum_n -\lambda_x \left(x_{nm} - \theta_n - \phi_m - u_n^T v_m \right) u_n$$
$$+ \lambda_v \left(v_m - W_v b_m \right) = 0$$

$$\Rightarrow \left(\sum \lambda_x u_n u_n^T + \lambda_v I_K \right) v_m = \lambda_v W_v b_m + \sum_n \lambda_x \left(x_{nm} - \theta_n - \phi_m \right) u_n$$

we can write $\sum u_n u_n^T = U^T U$

$$\Rightarrow \boxed{v_m = \left(\lambda_x U^T U + \lambda_v I_K \right)^{-1} \left[\sum_n \lambda_x \left(x_{nm} - \theta_n - \phi_m \right) u_n + \lambda_v W_v b_m \right]}$$

Similarly, for u_m

$$\boxed{u_m = \left(\lambda_x V^T V + \lambda_u I_K \right)^{-1} \left[\sum_m \lambda_x \left(x_{nm} - \theta_n - \phi_m \right) v_m + \lambda_u W_u a_n \right]}$$

∴ Using the equation show above. ALT-OPT algoo will be :—

• Initial

PTO

ALT – OPT

→ ① Initialise latent variables $u_n = \hat{u}_n$; $v_m = \hat{v}_m$

→ ② Solve for $\hat{w}_u$, $\hat{w}_v$, $\hat{\theta}_n$, $\hat{\theta}_m$ using the eq^n.

→ ③ Solve $\hat{u}_n$, $\hat{v}_m$ using the eq^n

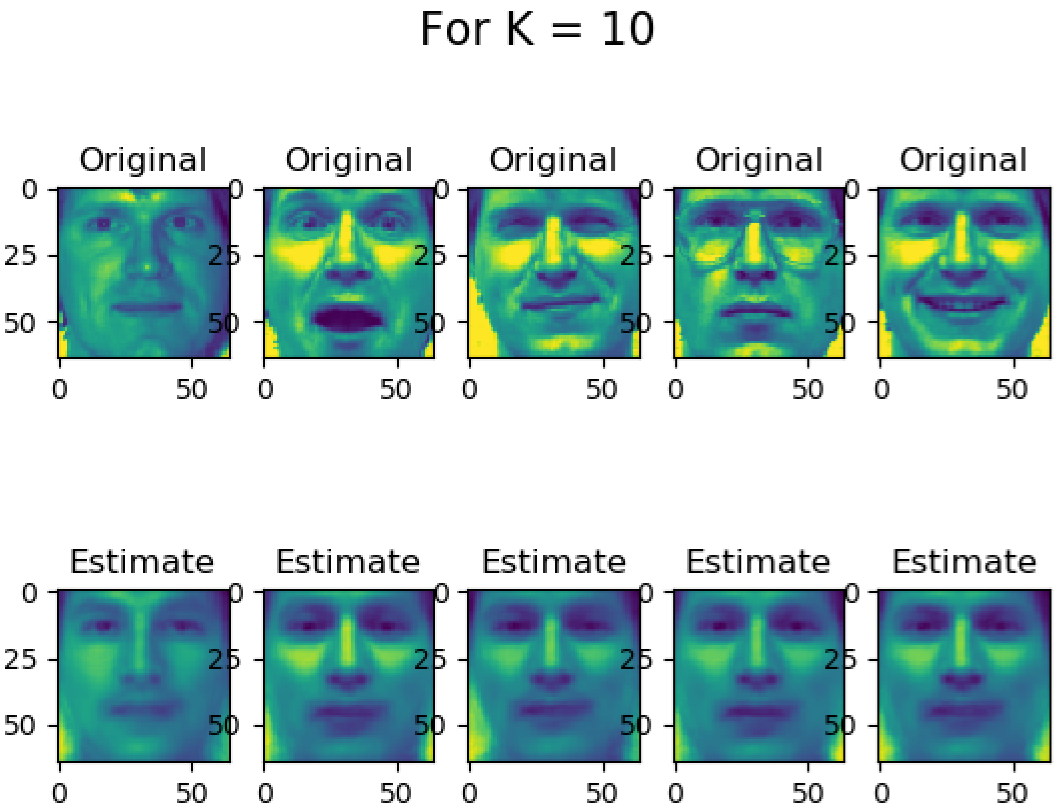
→ ④ Go to step ② if not converge.

Introduction to ML (CS771), Autumn 2018
Indian Institute of Technology Kanpur
Homework Assignment Number 5

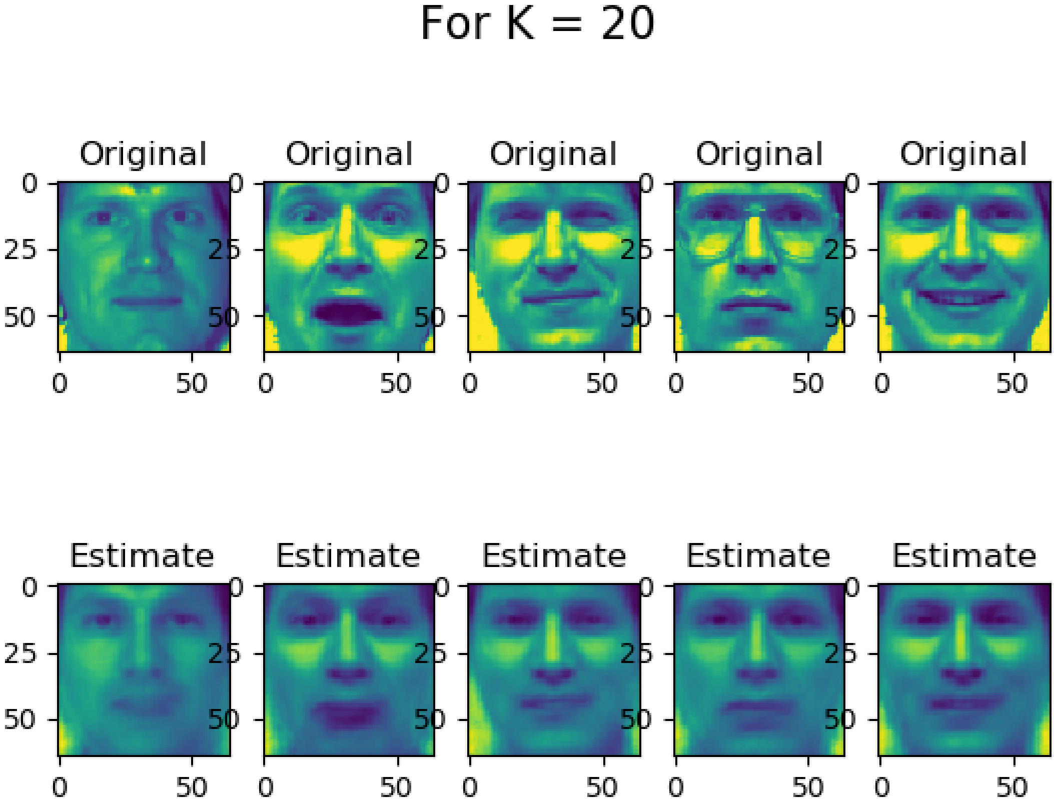
QUESTION 5

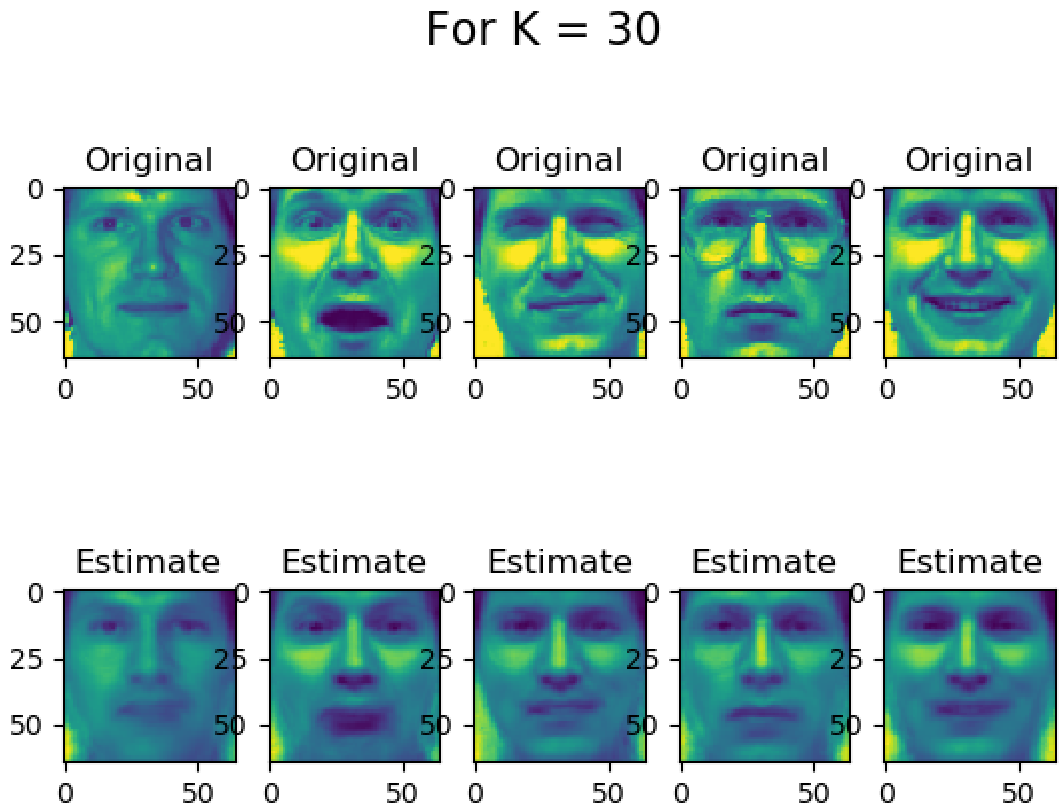
Student Name: Shashi Kant Gupta
Roll Number: 160645
Date: November 17, 2018

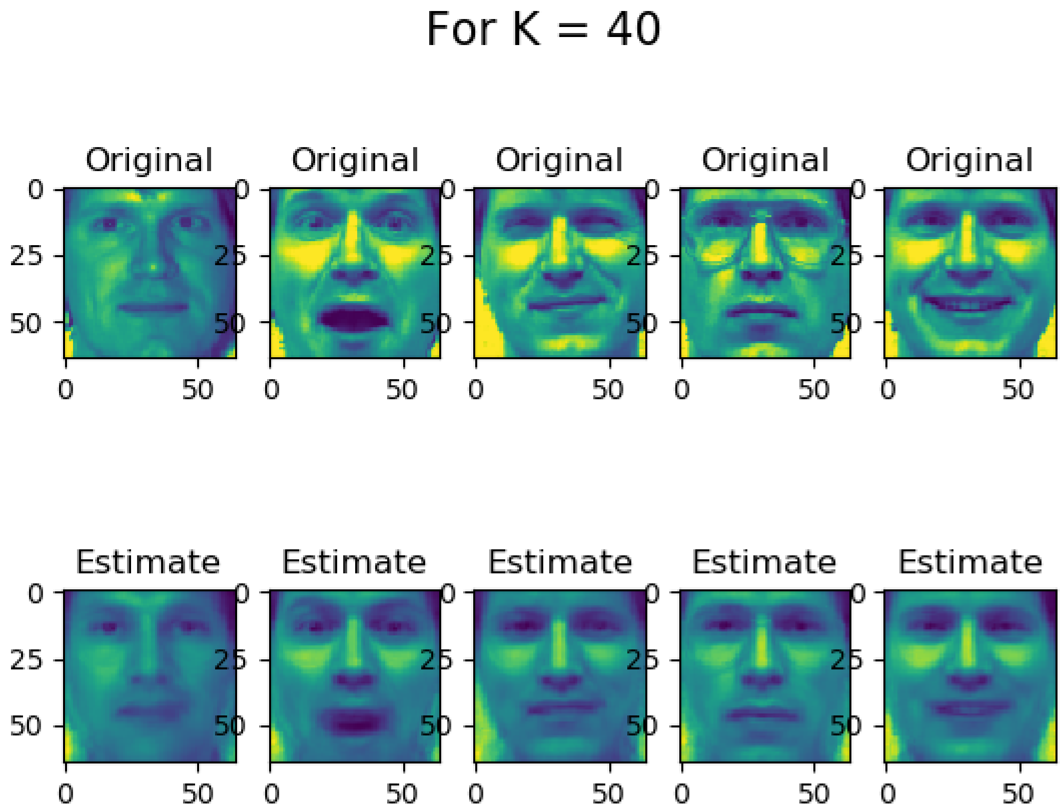
Programming Part 1:
On increasing k there is improvement in the reconstruction of images! Morevover the images appear to be more clearer. Explanation is simple higher the value of k more the no of features retained in z.
Please find the plots from next page.

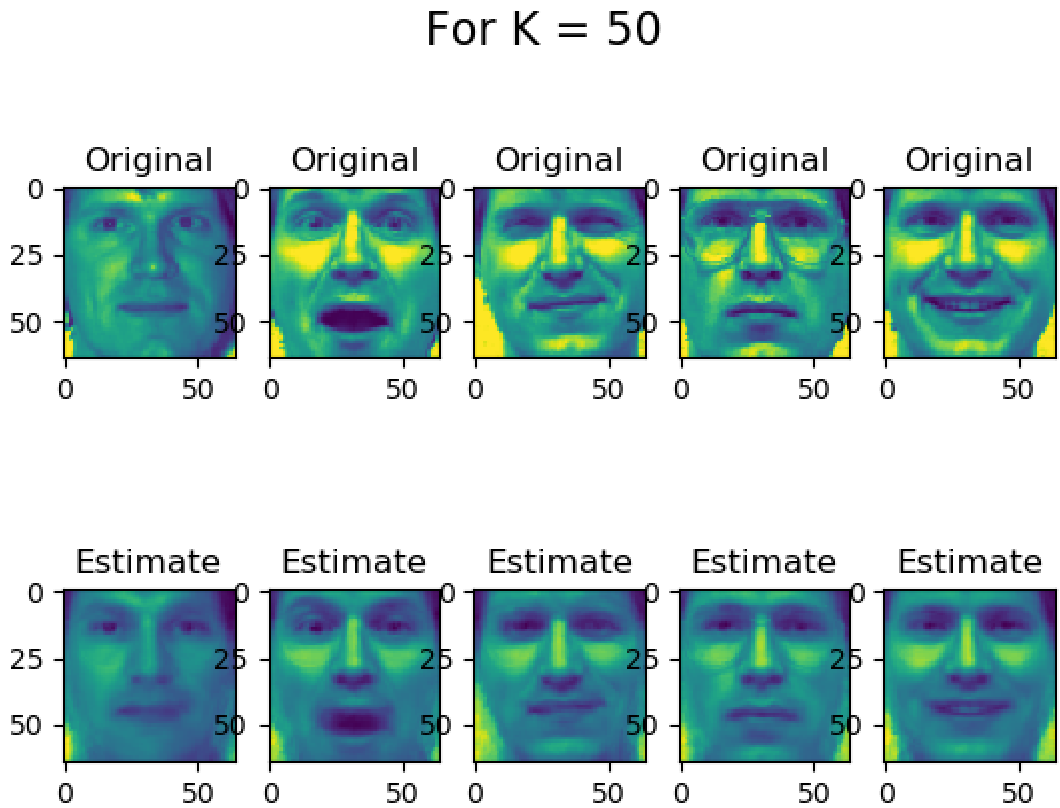


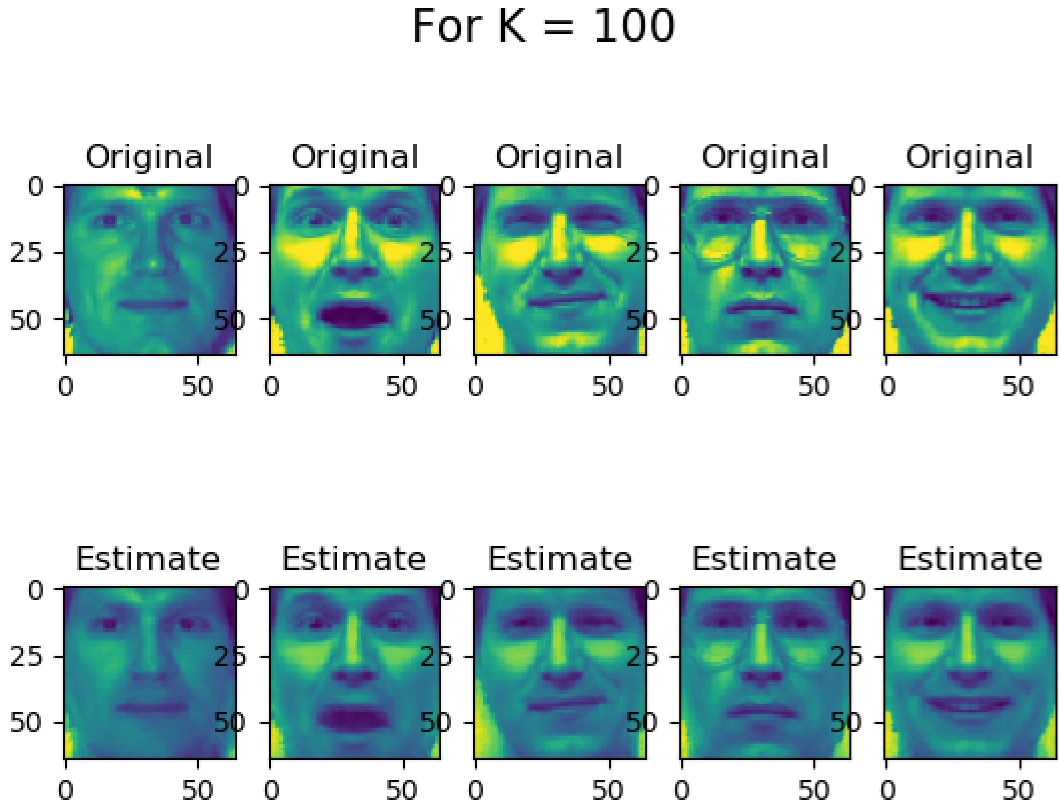
For K = 10
Original
Original
Original
Original
Original
Estimate
Estimate
Estimate
Estimate
Estimate

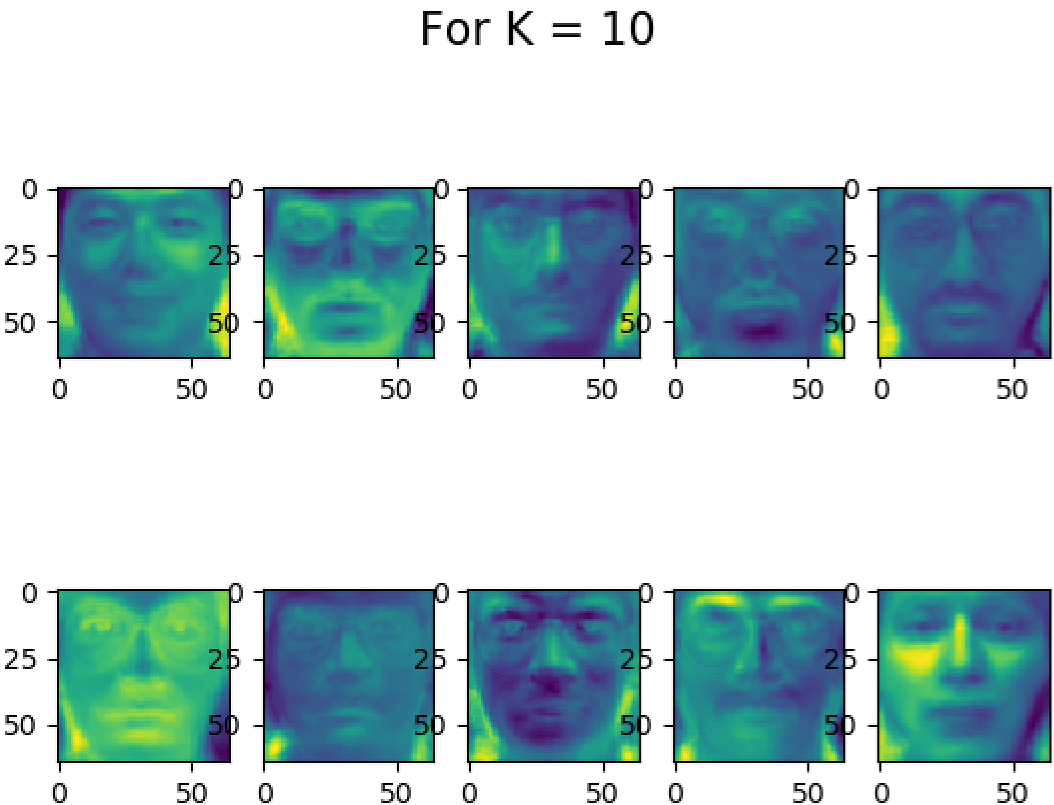

For K = 20
Original
Estimate
0
25
50


For K = 30
Original
Original
Original
Original
Original
Estimate
Estimate
Estimate
Estimate
Estimate
0
25
50
0
50

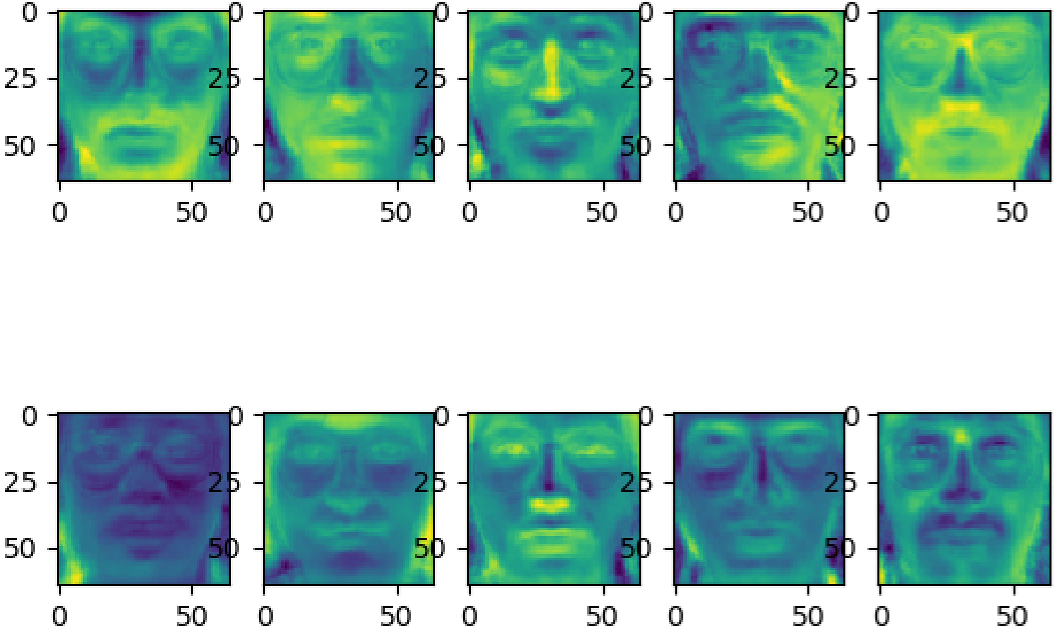

For K = 40
Original
Original
Original
Original
Original
Estimate
Estimate
Estimate
Estimate
Estimate
0
25
50
0
50


For K = 50
Original
Original
Original
Original
Original
Estimate
Estimate
Estimate
Estimate
Estimate
0
25
50
0
50

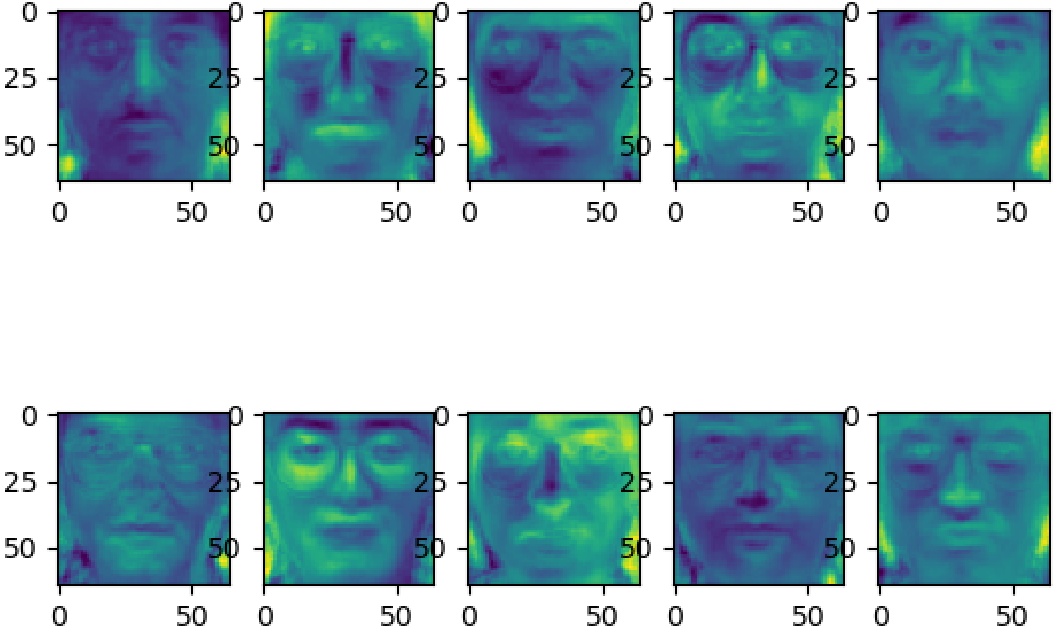

For K = 100
Original
Original
Original
Original
Original
Estimate
Estimate
Estimate
Estimate
Estimate
0
25
50
0
50


For K = 10
0
25
50
0
50

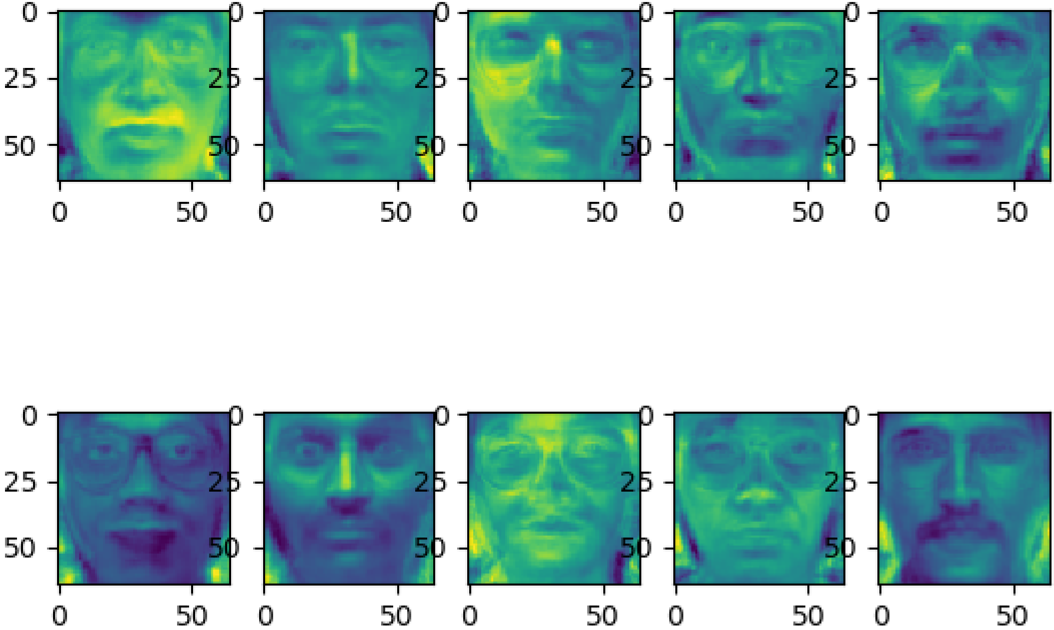
For K = 20



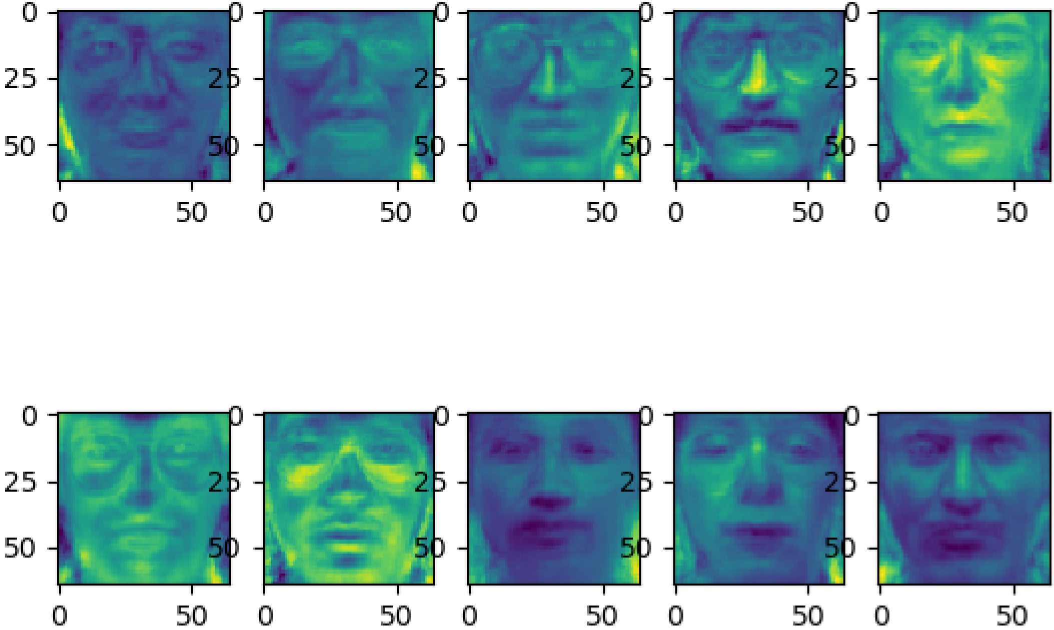
For K = 30

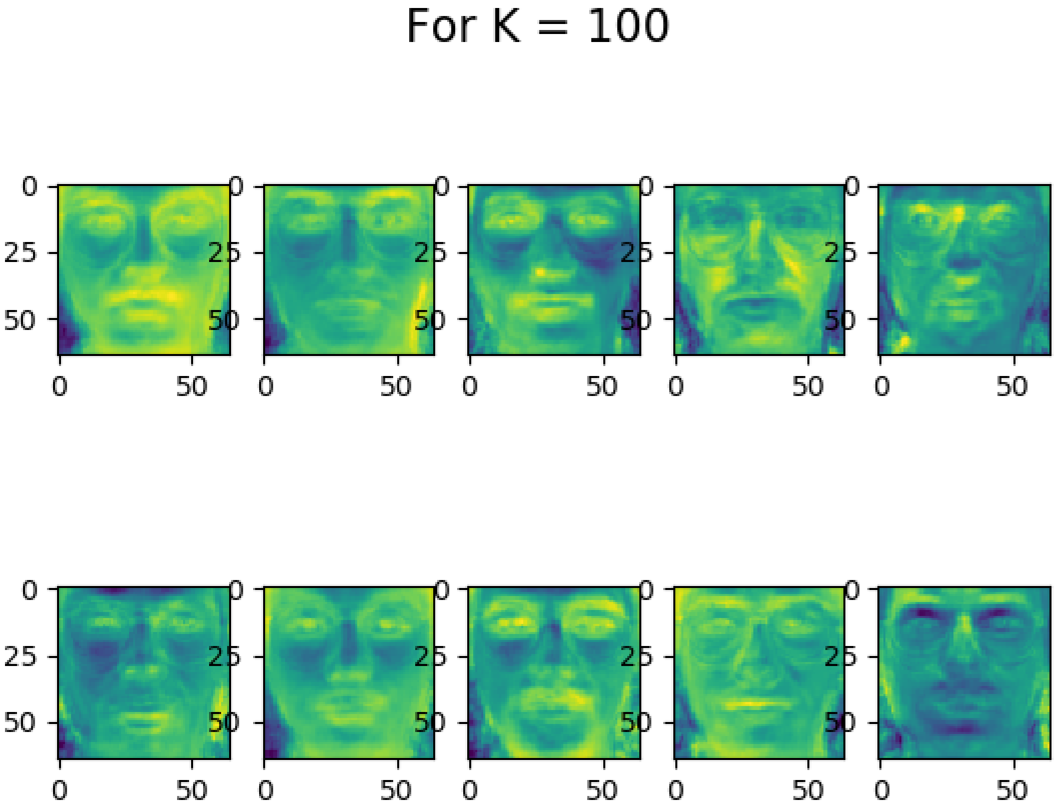


For K = 40

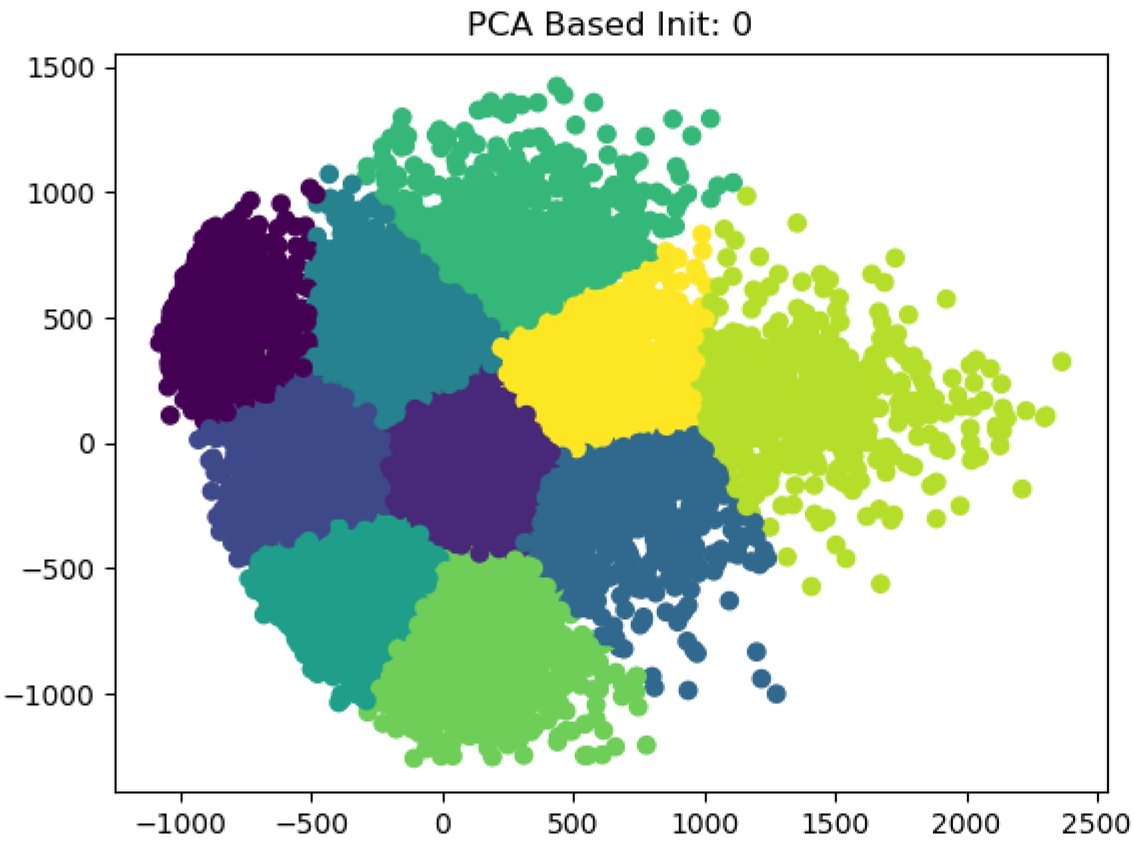


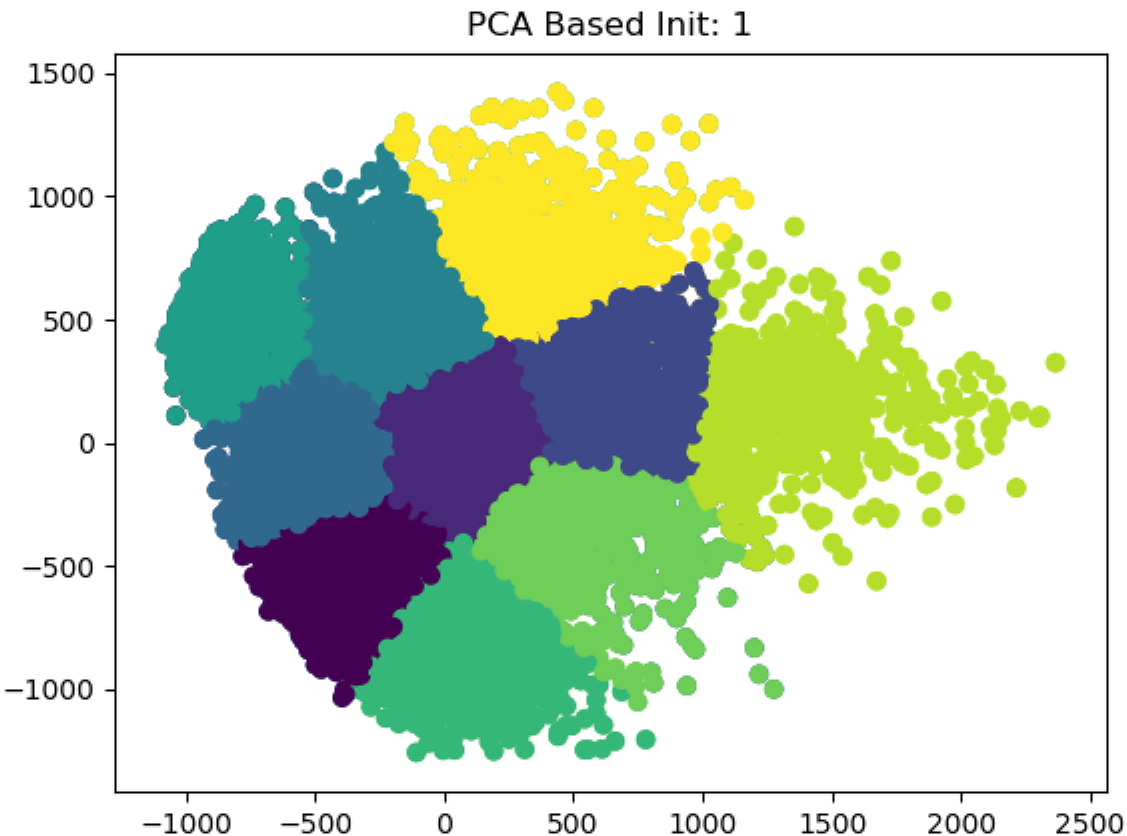
For K = 50

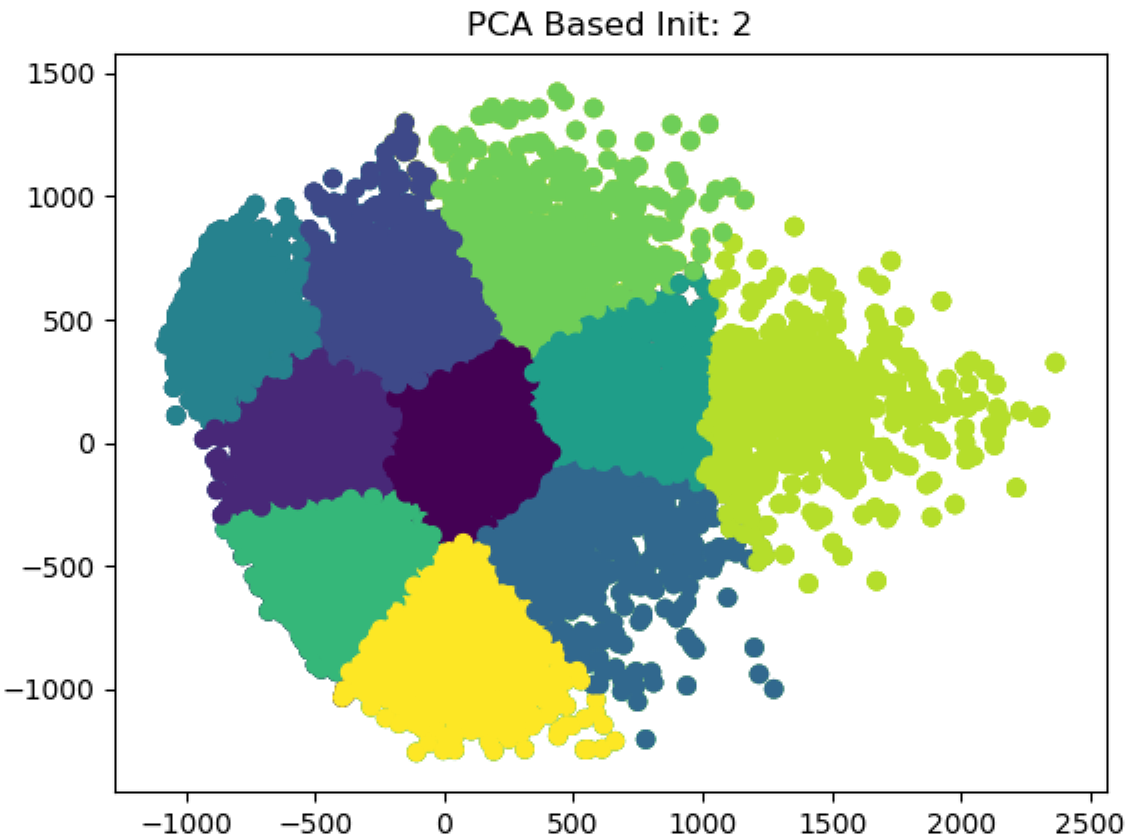


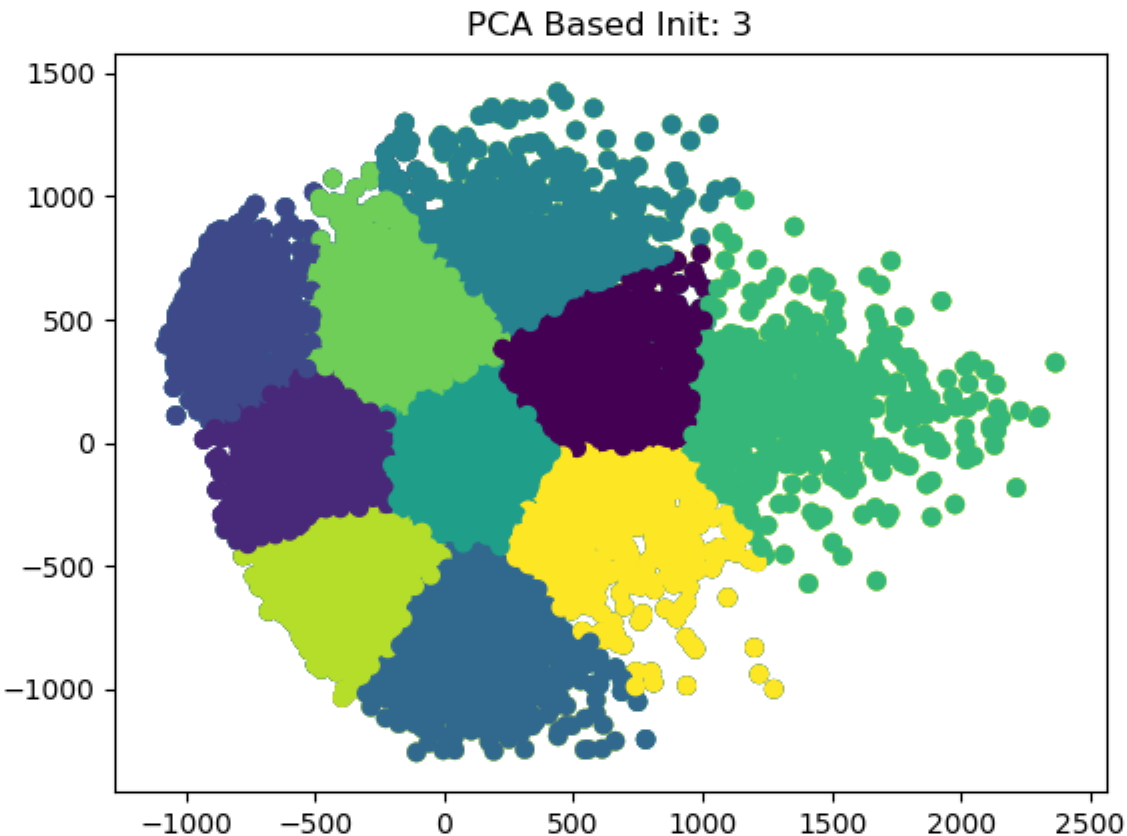

For K = 100
0
25
50
0
50

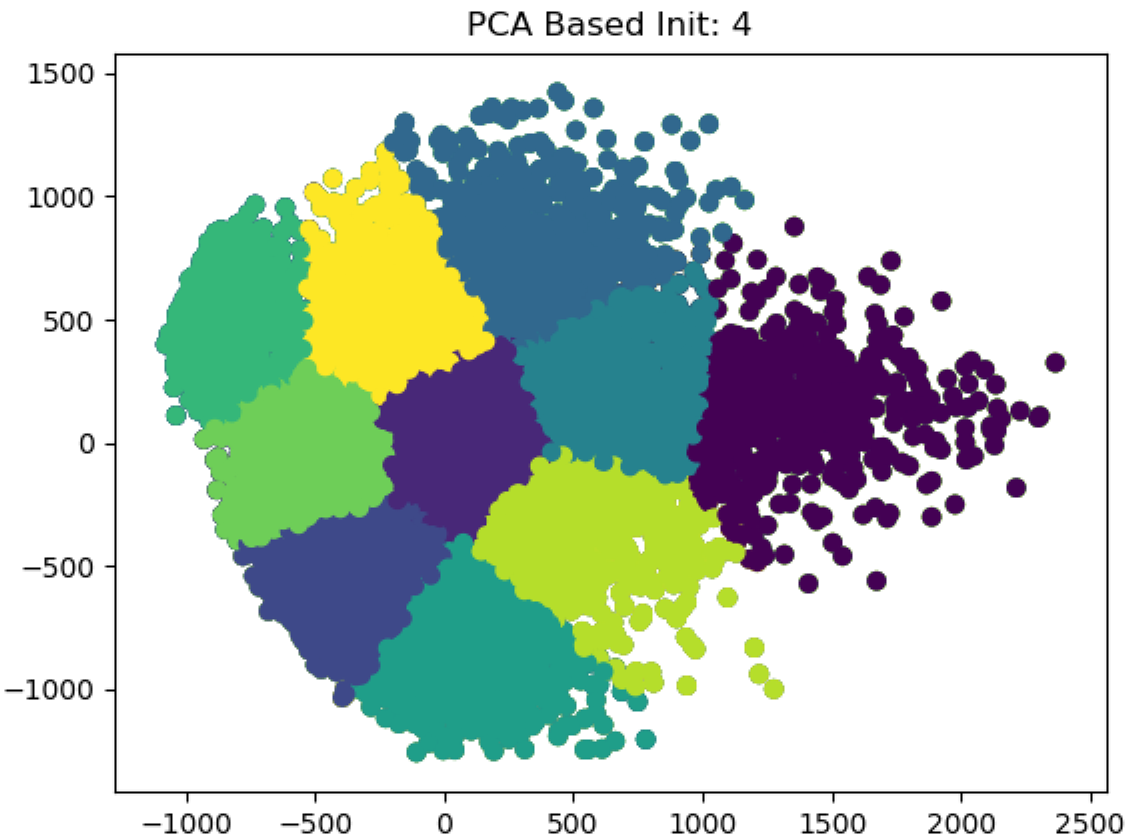
Programming Part 2:
Visually tSNE appears to be better, clusters are more seperated as well as on running 10 different initiallisations errors in tSNE seems to be less than that of PCA!
Please find the plots from next page.

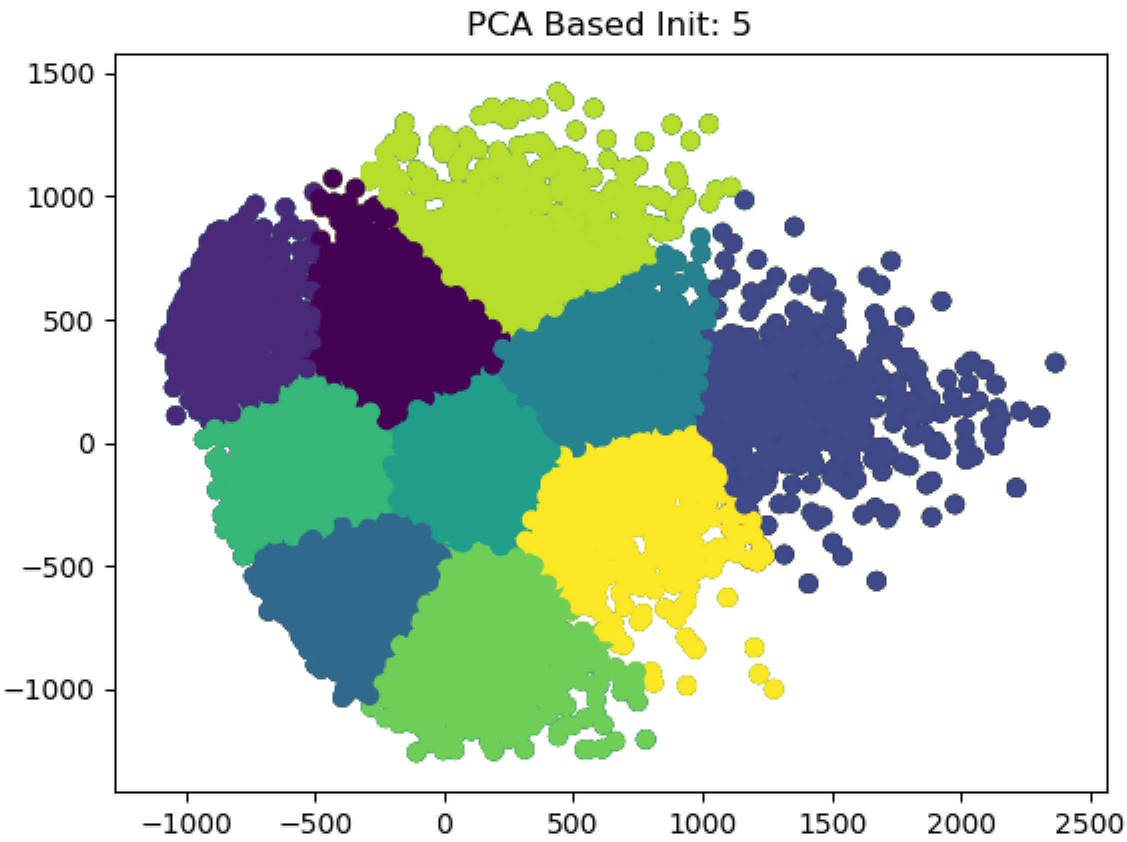

PCA Based Init: 0
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500


PCA Based Init: 1
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

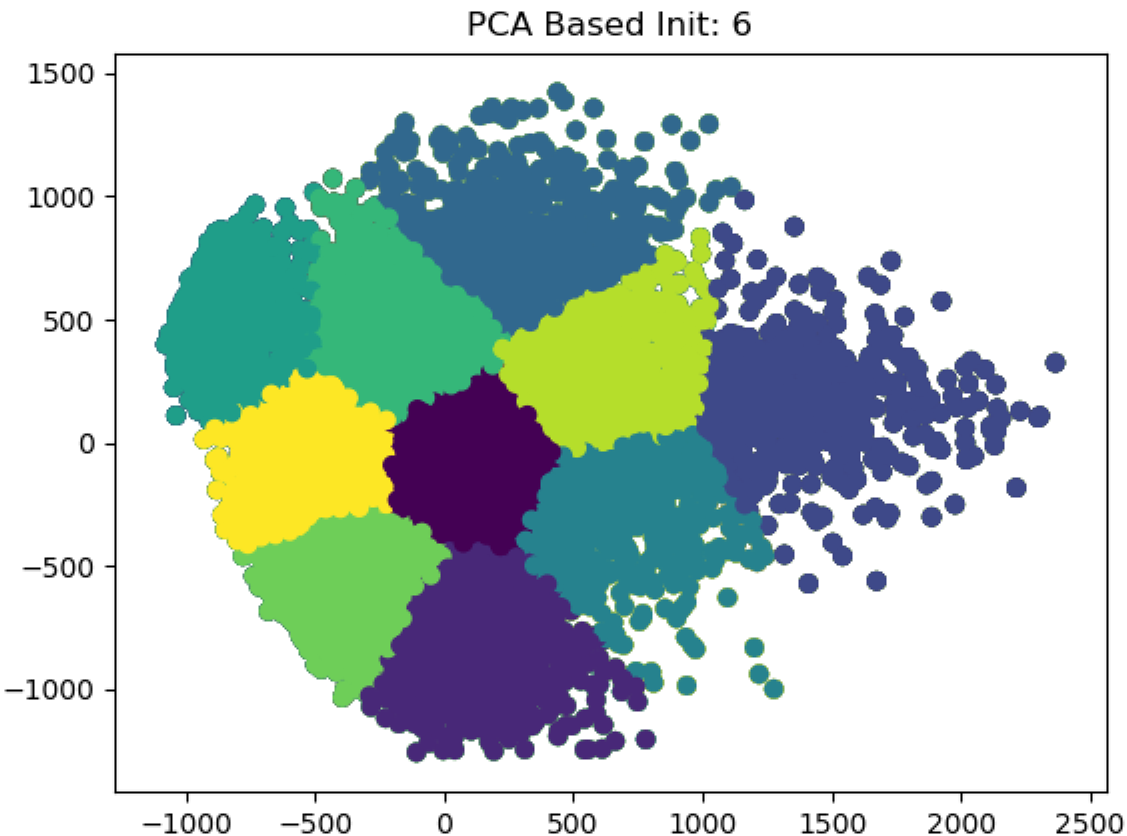

PCA Based Init: 2
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

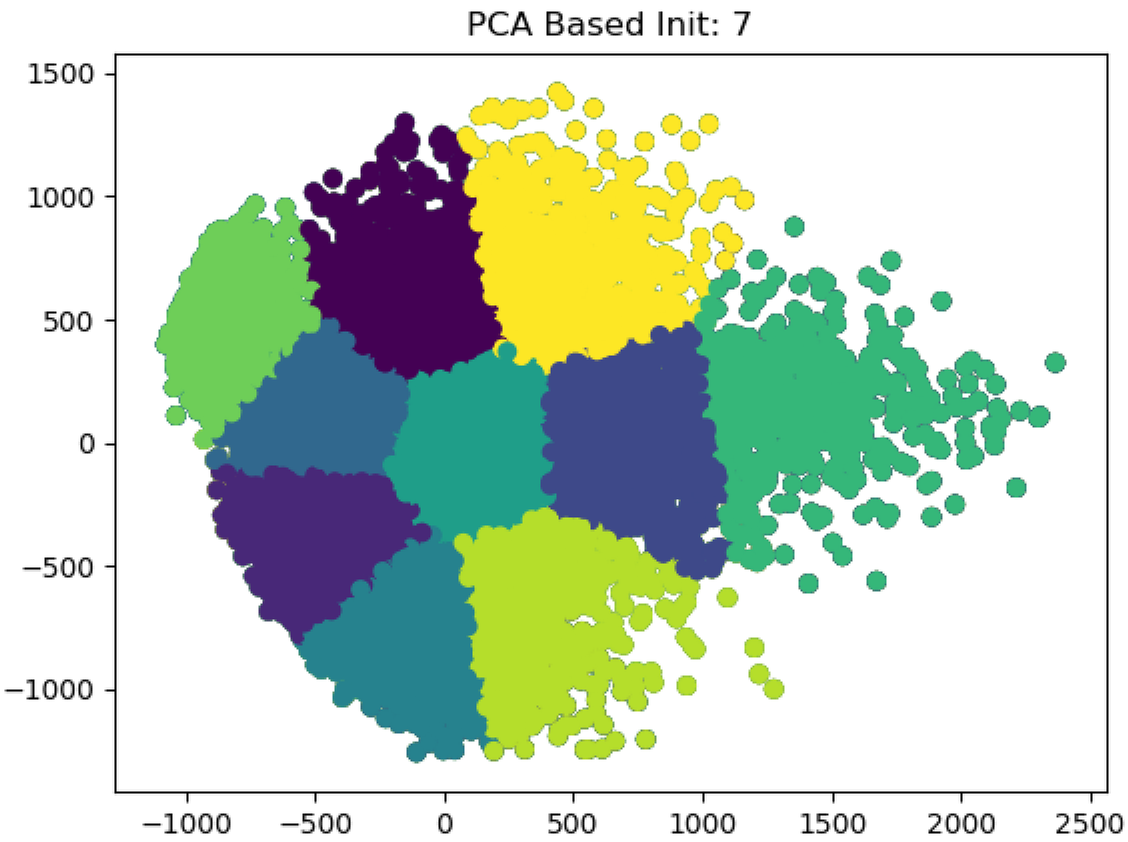

PCA Based Init: 3
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500


PCA Based Init: 4
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

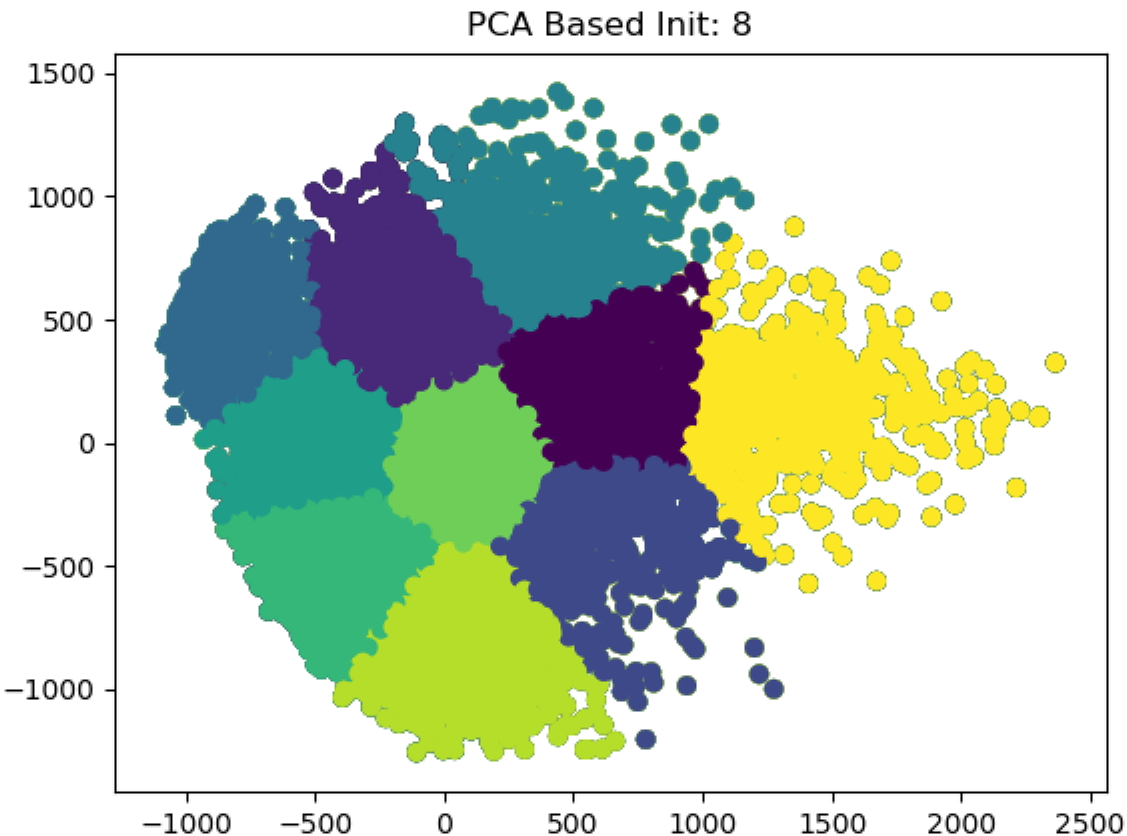


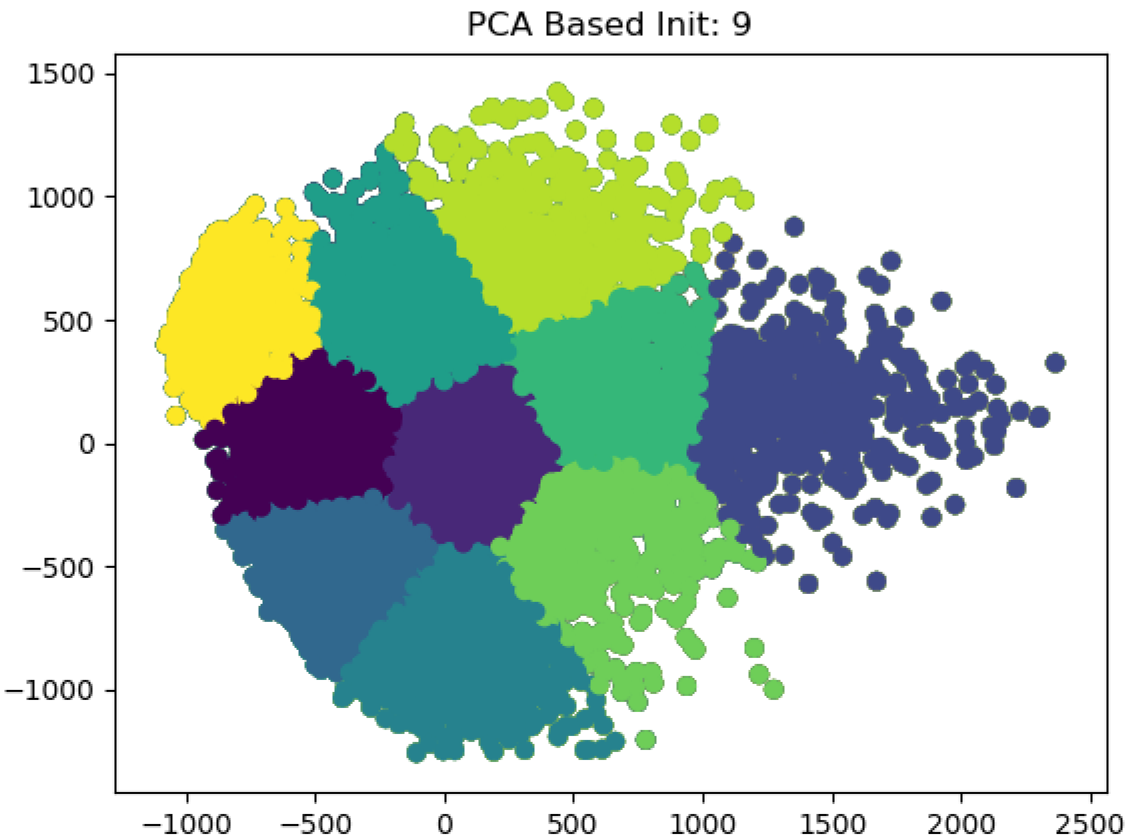
PCA Based Init: 5
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

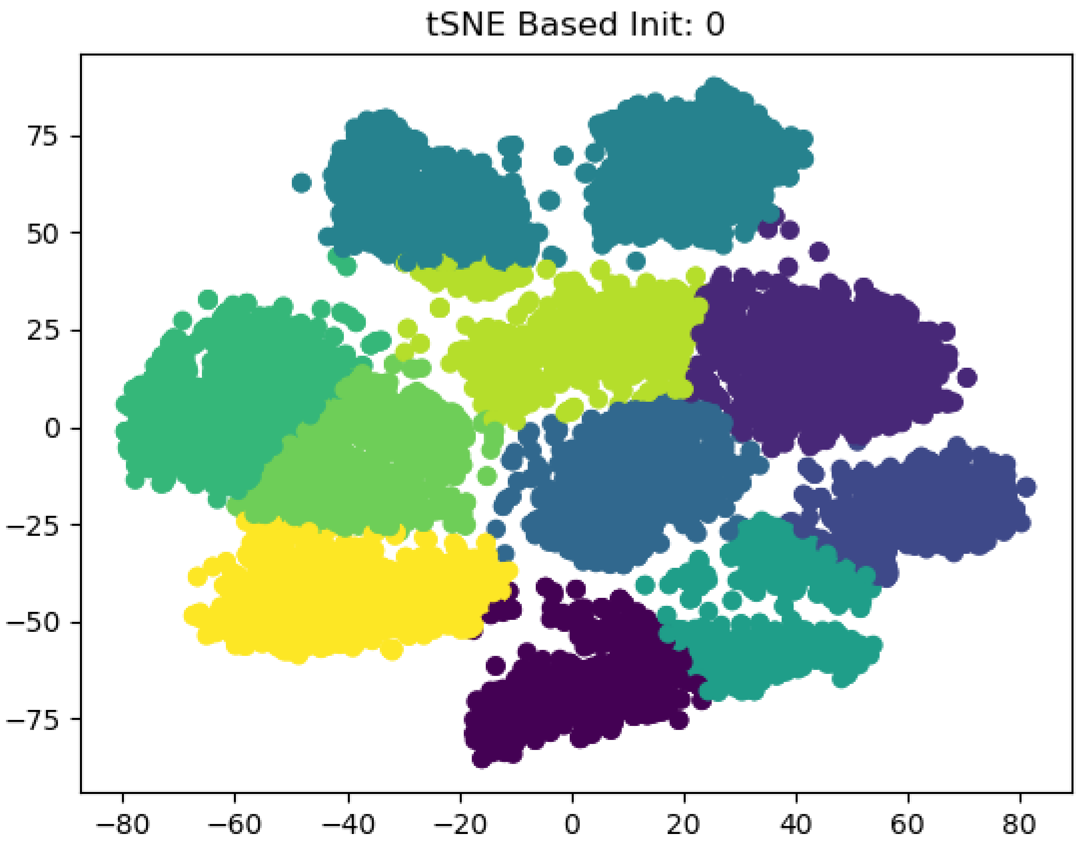

PCA Based Init: 6
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

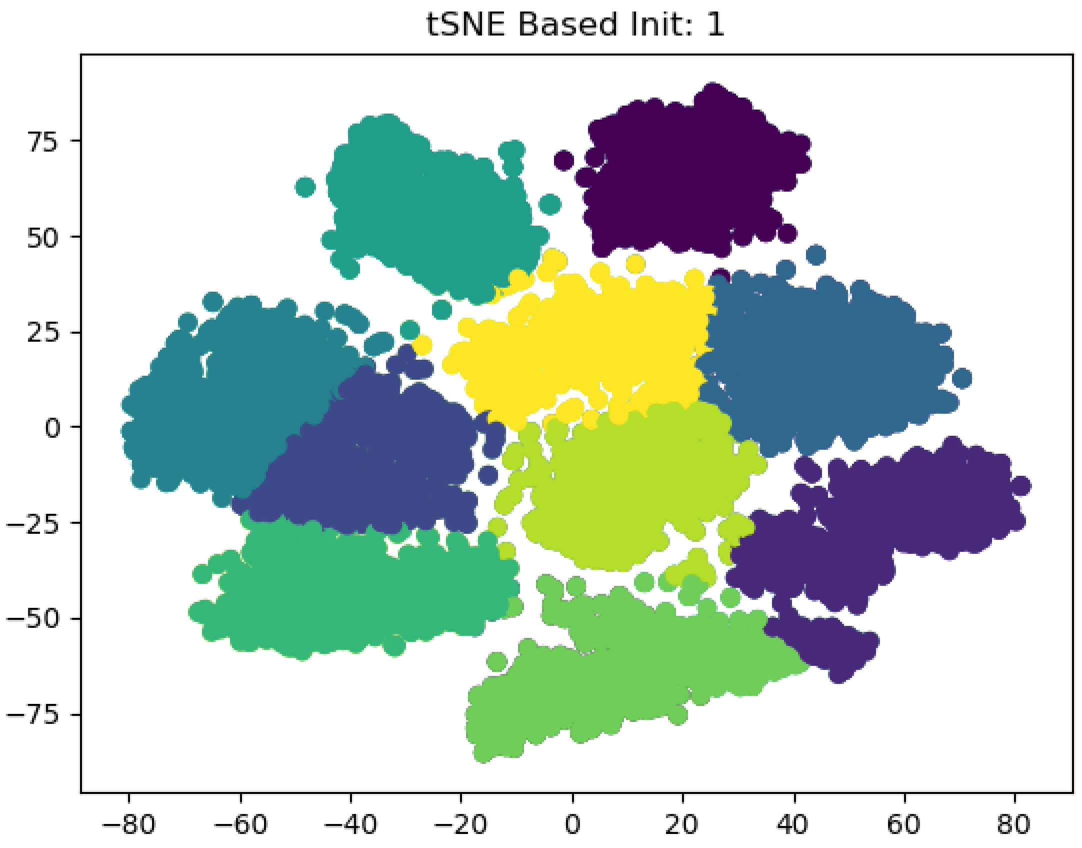


PCA Based Init: 7
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

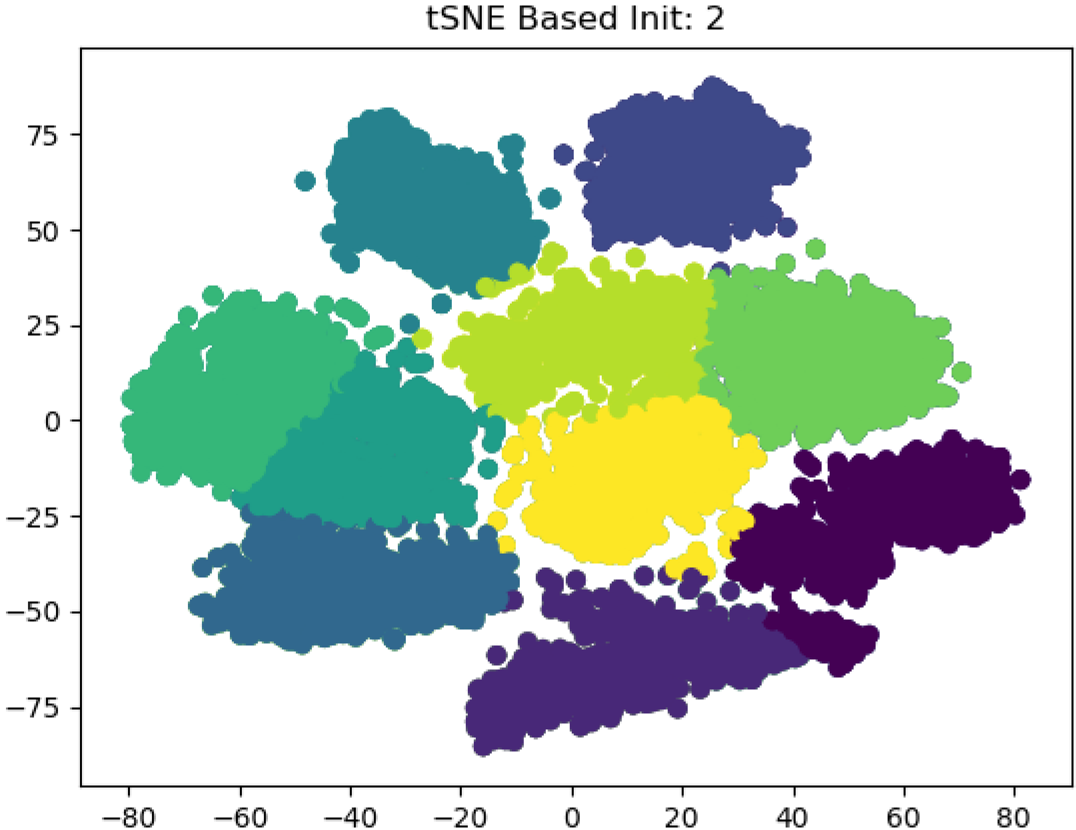

PCA Based Init: 8
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

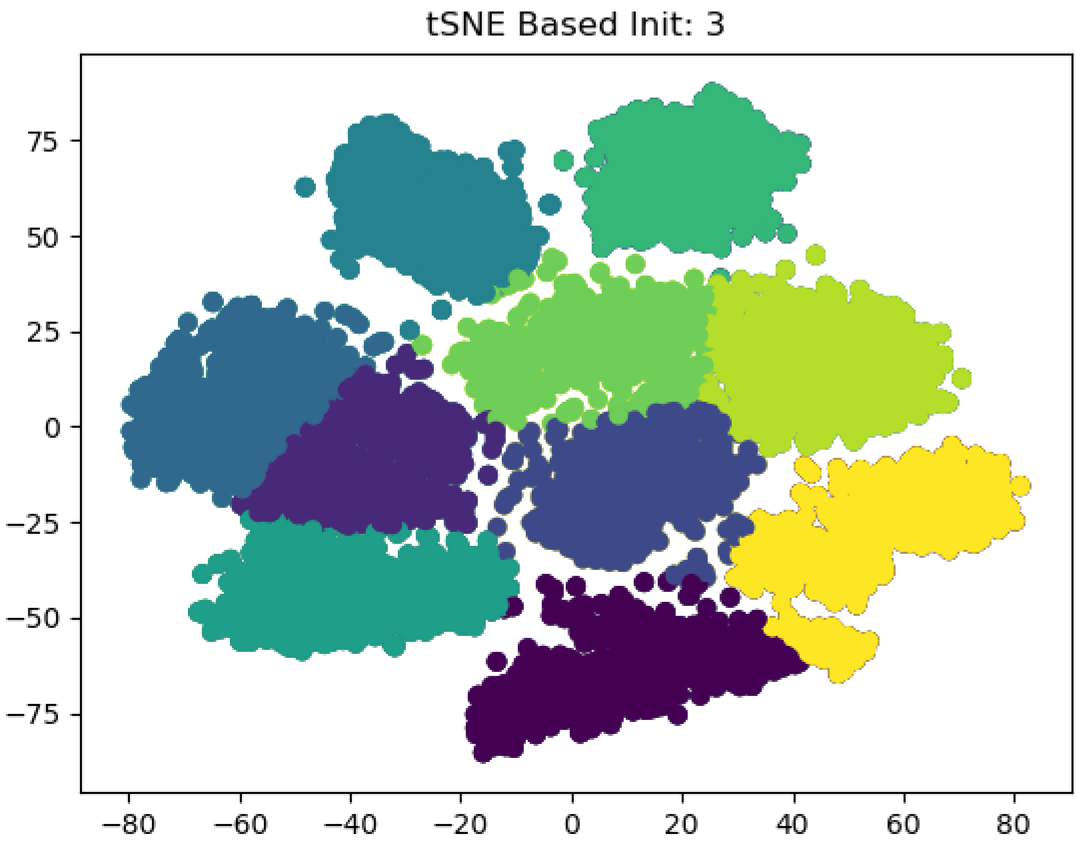

PCA Based Init: 9
1500
1000
500
0
−500
−1000
−1000
−500
0
500
1000
1500
2000
2500

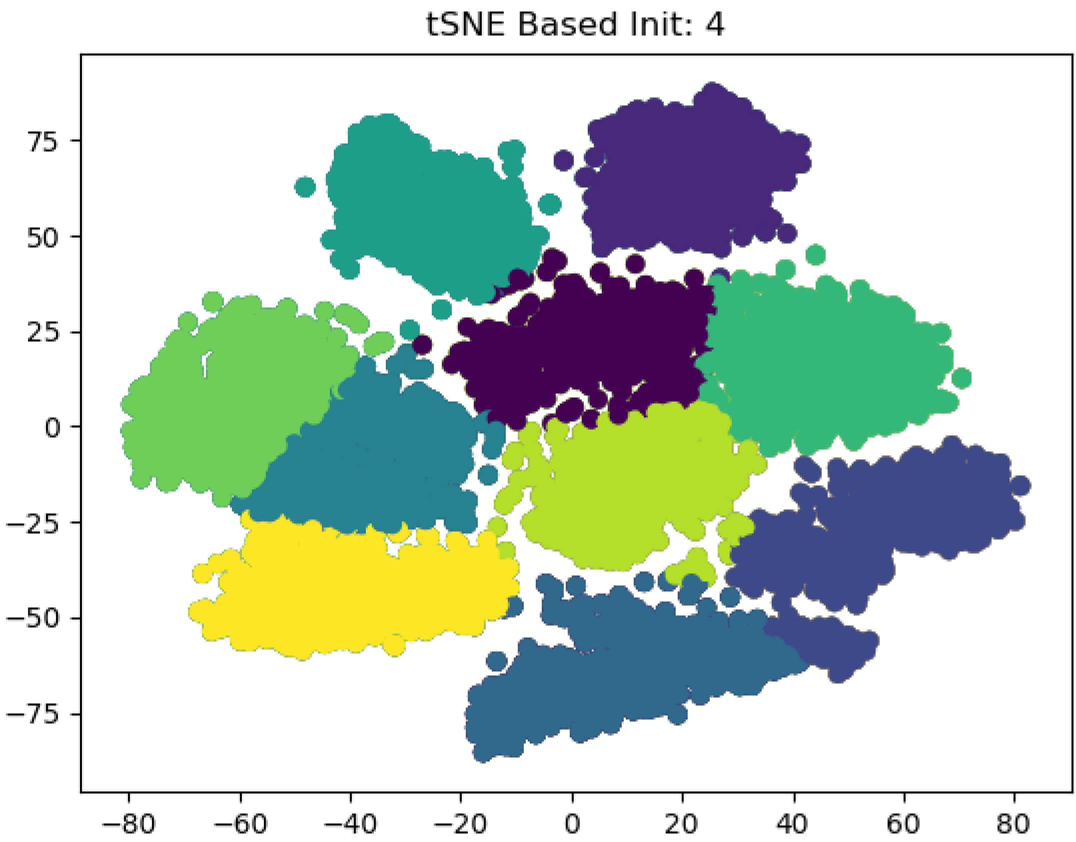

tSNE Based Init: 0
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80



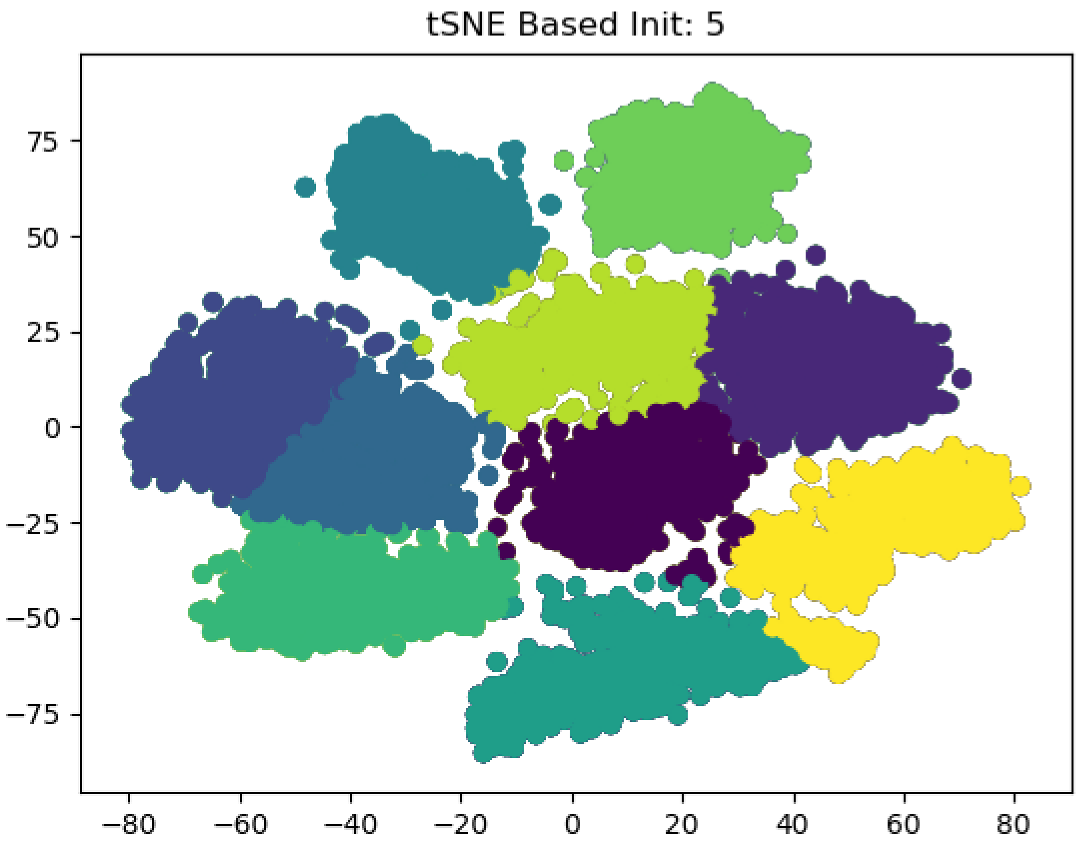
tSNE Based Init: 1
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80

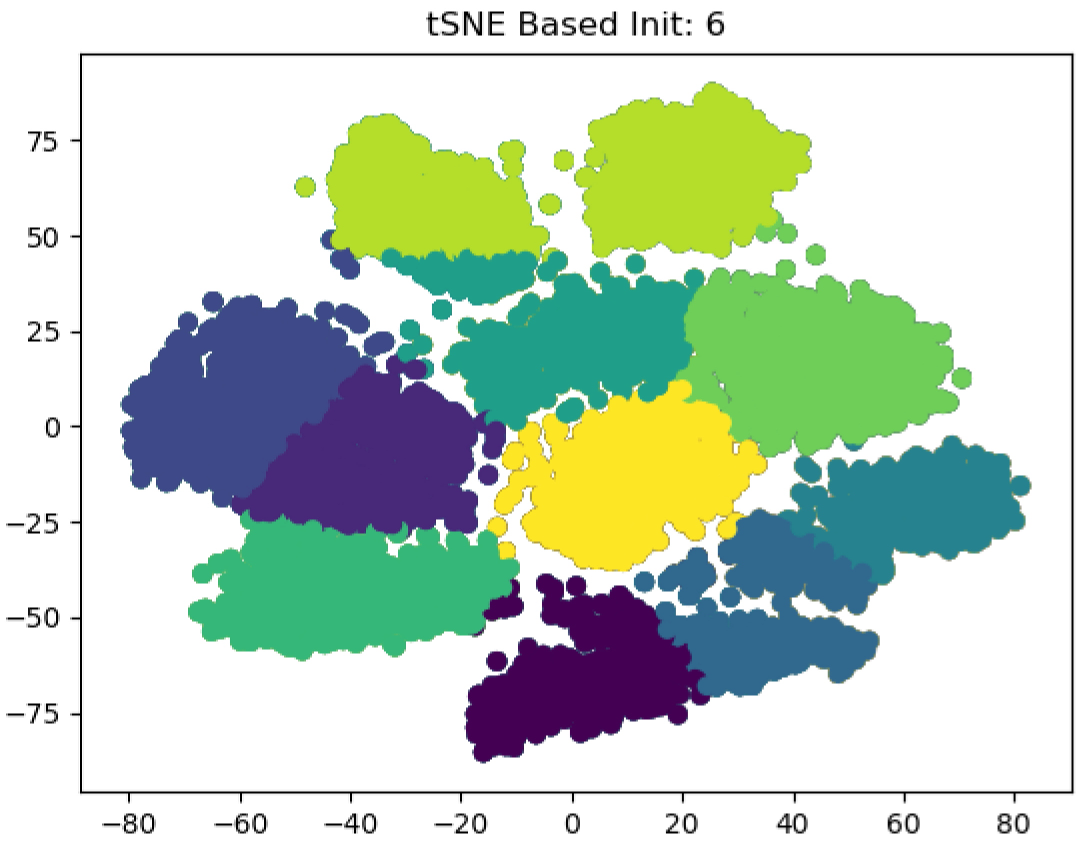

tSNE Based Init: 2
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80

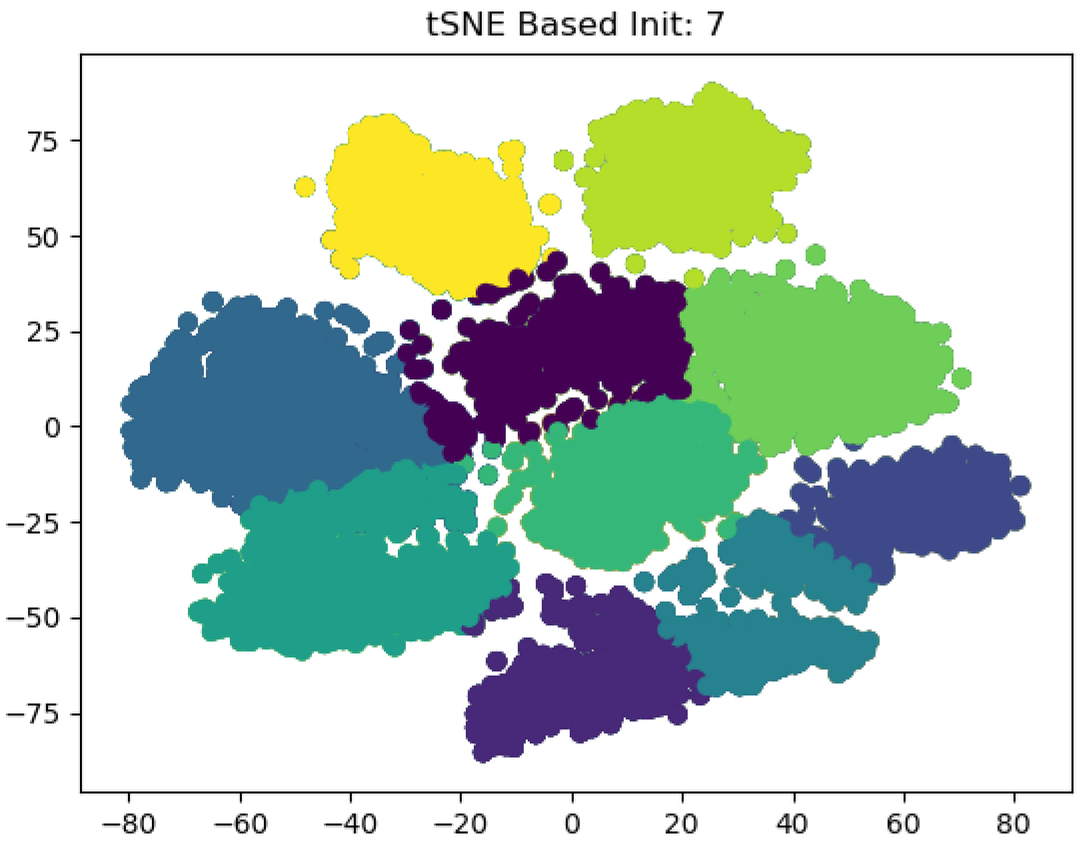

tSNE Based Init: 3
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80



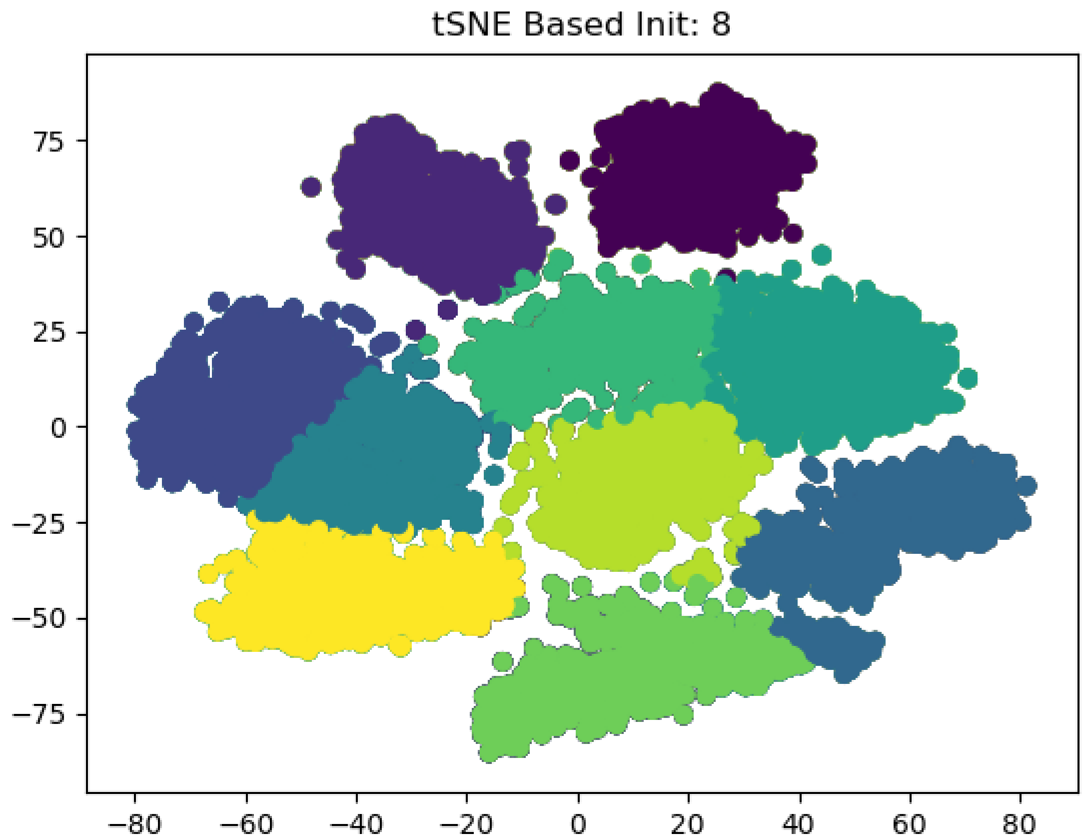
tSNE Based Init: 4
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80

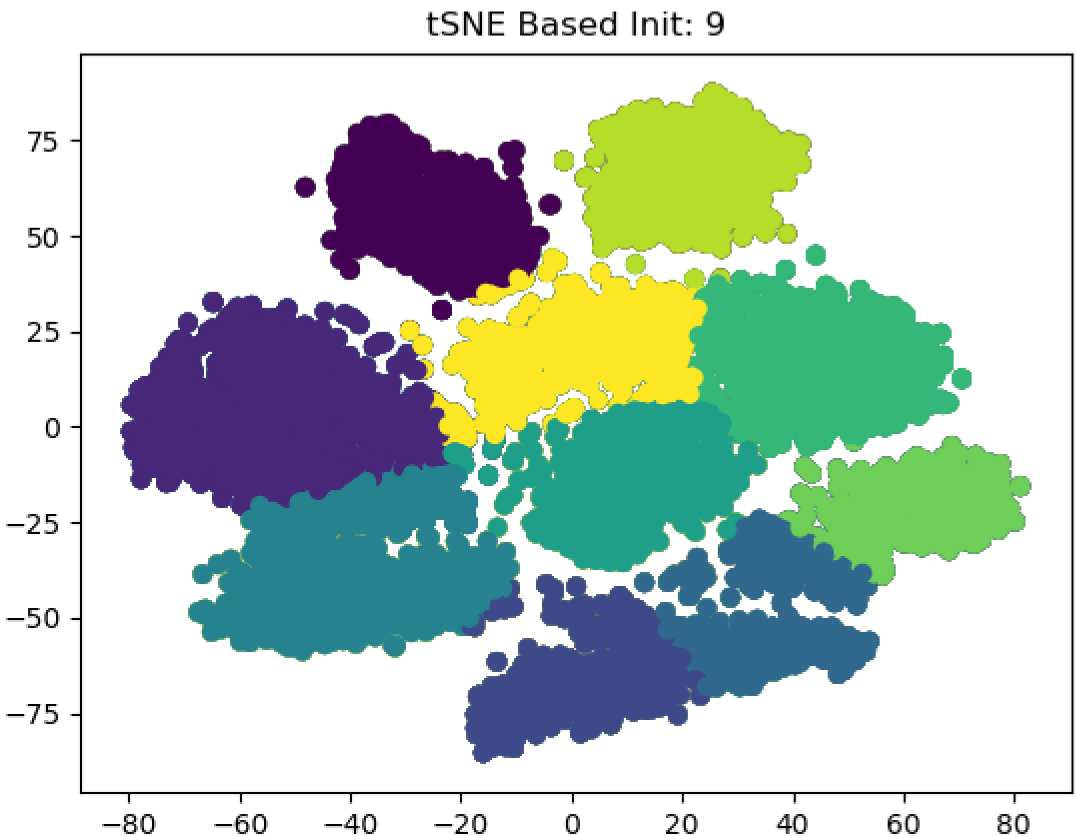

tSNE Based Init: 5
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80


tSNE Based Init: 6
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80



tSNE Based Init: 7
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80


tSNE Based Init: 8
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80


tSNE Based Init: 9
75
50
25
0
−25
−50
−75
−80
−60
−40
−20
0
20
40
60
80