CS529 Homework 2 Report

Venkata Balaji Koniki

September 24, 2024

1. Abstract

Network Protocol Reverse Engineering is a task that involves learning hidden patterns in an unknown protocol's packet trace. We are supposed to use unsupervised learning algorithms to tackle this problem. I use **Autoencoders** to reduce dimensionality and then apply **K-Means** upon encoded latent space vectors to cluster similar packets. The elbow graph has been leveraged to set the optimal value for hyperparameter **k**. After obtaining cluster labels for each packet header, the goal is to find such a byte index that has the least **Intra-Cluster-Variance** and maximum **Inter-Cluster-Variance**.

2. Data Exploration

1. Byte frequencies of all bytes in training data for three protocols are computed using **collections** package. code submitted in *histogram-Plots.ipynb.*

2. It has been observed that packet header sizes are variable in protocol-2,3, unlike protocol-1. As we need samples of equal dimensions before feeding to an ML model, shorter packets are padded with zeroes at the end.

3. There are a few packets in protocol-3 data that are the longest and have lots of zeroes at the end. These packets add unnecessary padding to shorter packets which are in the majority. This has been avoided by deleting all-zero columns at the end of the data numpy array.

4. There are multiple fields (2-byte sequence) in the header that stay constant all the time. These fields have the least variance of 0 across packets, which need to be ignored while looking for *message type* field which has some minimum non-zero variance.

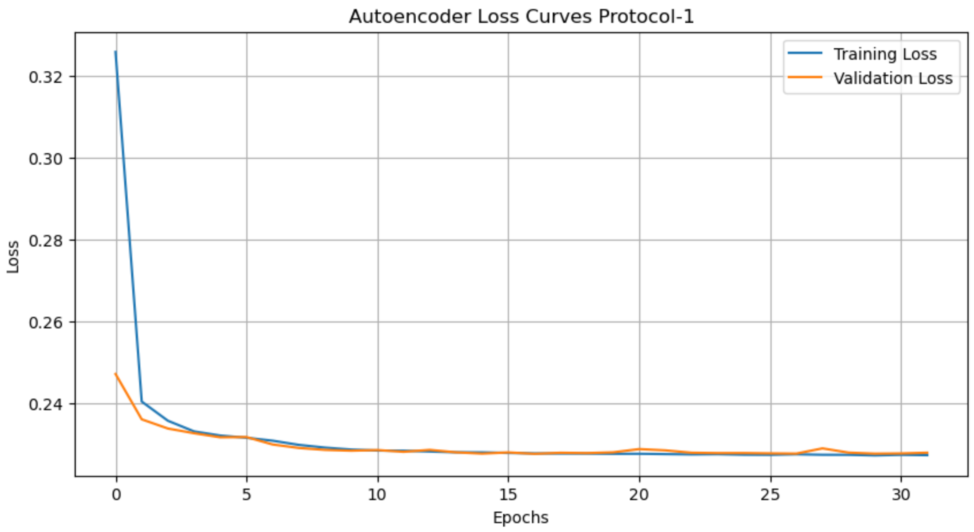
3. Algorithm Development

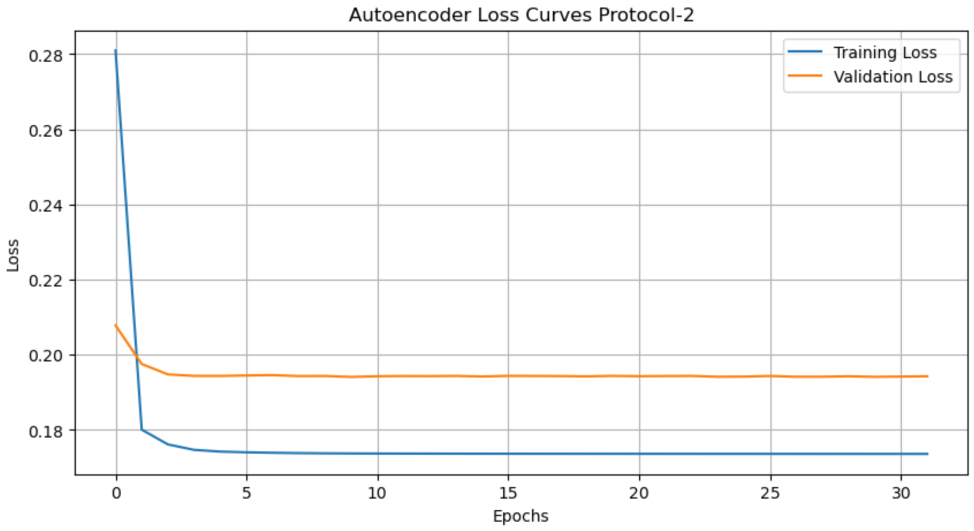
1. Autoencoder has been selected to achieve dimensionality reduction. This is preferred over PCA as packet header data is non-linear. After training the autoencoder, its components, encoder, and decoder models are saved to transform data from input space to latent space and reconstruct back respectively. Model is trained with Adam as gradient descent optimizer and binary_crossentropy loss as our data lies between 0 and 1 after pre-processing. After thorough experimentation, epoch and batch_size are set to 32 for effective training.

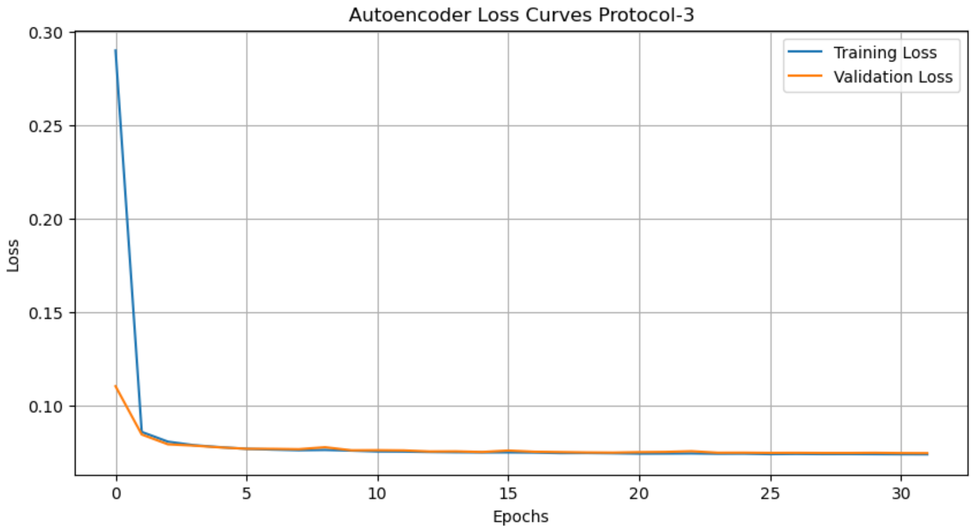
Autoencoder Model Architecture Protocol-3

```
    Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_15 (InputLayer)       [(None, 360)]             0

 dense_40 (Dense)            (None, 256)               92416

 batch_normalization_26 (Bat  (None, 256)              1024
 chNormalization)

 dense_41 (Dense)            (None, 64)                16448

 batch_normalization_27 (Bat  (None, 64)               256
 chNormalization)

 dense_42 (Dense)            (None, 32)                2080

 dense_43 (Dense)            (None, 64)                2112

 batch_normalization_28 (Bat  (None, 64)               256
 chNormalization)

 dense_44 (Dense)            (None, 256)               16640

 batch_normalization_29 (Bat  (None, 256)              1024
 chNormalization)

 dense_45 (Dense)            (None, 360)               92520

=================================================================
Total params: 224,776
Trainable params: 223,496
Non-trainable params: 1,280
_________________________________________________________________
```

Loss Curves

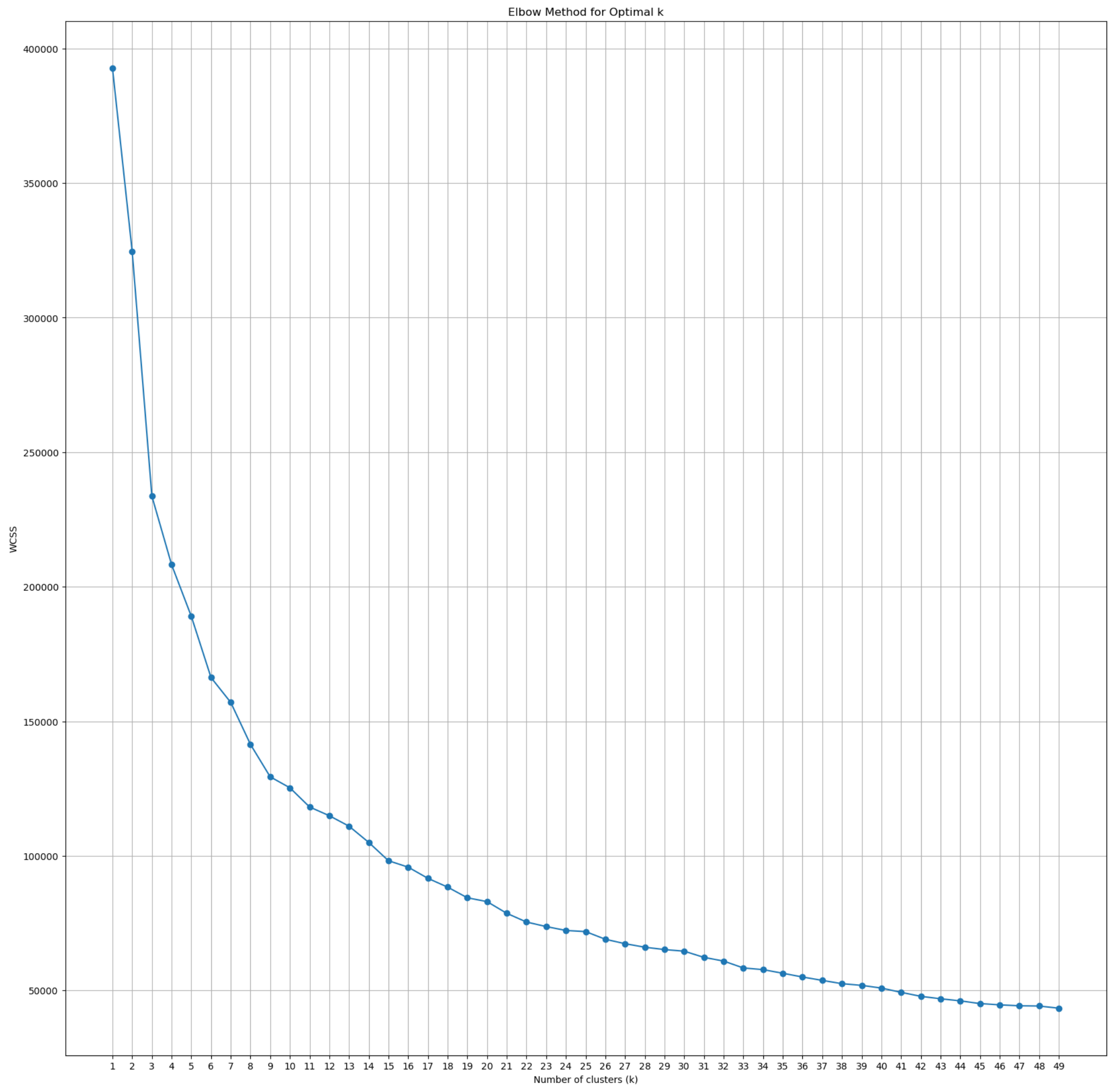

Autoencoder Loss Curves Protocol-1
Training Loss
Validation Loss
Loss
Epochs
0.32
0.30
0.28
0.26
0.24
0
5
10
15
20
25
30


Autoencoder Loss Curves Protocol-2
Training Loss
Validation Loss
Loss
Epochs
0.28
0.26
0.24
0.22
0.20
0.18
0
5
10
15
20
25
30


Autoencoder Loss Curves Protocol-3
Training Loss
Validation Loss
Loss
Epochs
0.30
0.25
0.20
0.15
0.10
0
5
10
15
20
25
30

2. As we are working with unknown protocol data, there is no scope for deep neural networks that are trained with labels in supervised learning. A basic K-Means algorithm is used to cluster packets. The elbow curve helps us set a k-value to cluster data optimally.

Elbow graph for Protocol-3



3. After getting the cluster labels for each packet, intra and inter-cluster variances are computed. Theoretically, assuming that K-Means clusters packets based on the message type, this field should have the least variance within a cluster and maximum variance across clusters. Finally, the field index which has the least variance among most clusters is interpreted as a message-type index.

4. Model Optimization and Evaluation

1. It has been observed that autoencoder architecture depends on the size of the packet header to transform data optimally. After training with multiple combinations of layers, we ended up with three different models.

Autoencoder Model Architecture Protocol-1

```
   Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 32)]              0

 dense_4 (Dense)             (None, 16)                528

 batch_normalization_2 (Batc  (None, 16)               64
 hNormalization)

 dense_5 (Dense)             (None, 16)                272

 dense_6 (Dense)             (None, 16)                272

 batch_normalization_3 (Batc  (None, 16)               64
 hNormalization)

 dense_7 (Dense)             (None, 32)                544

=================================================================
Total params: 1,744
Trainable params: 1,680
Non-trainable params: 64
_________________________________________________________________
```

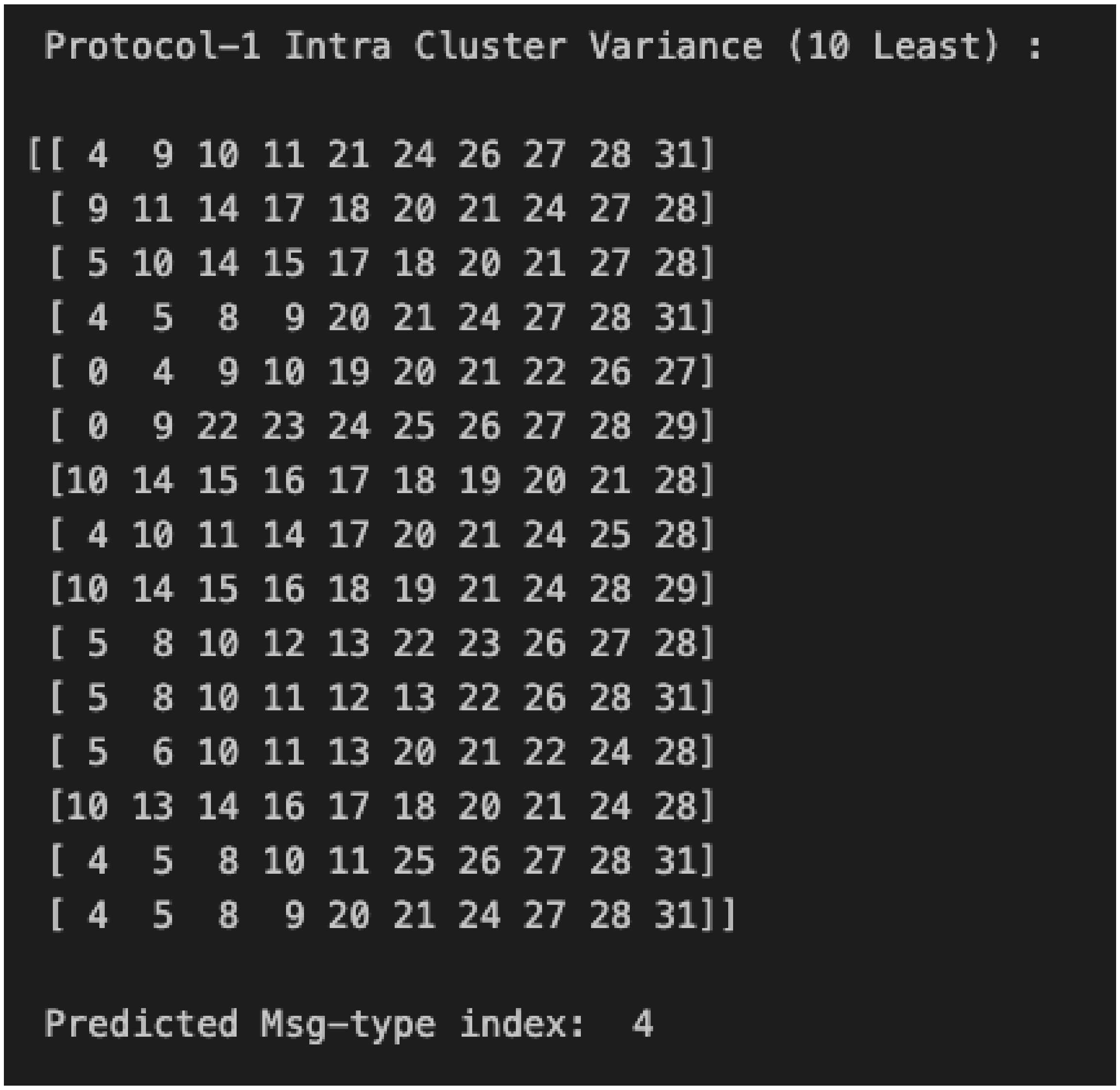
Autoencoder Model Architecture Protocol-2

```
   Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_3 (InputLayer)        [(None, 12)]              0

 dense_2 (Dense)             (None, 10)                130

 dense_3 (Dense)             (None, 12)                132

=================================================================
Total params: 262
Trainable params: 262
Non-trainable params: 0
_________________________________________________________________
```

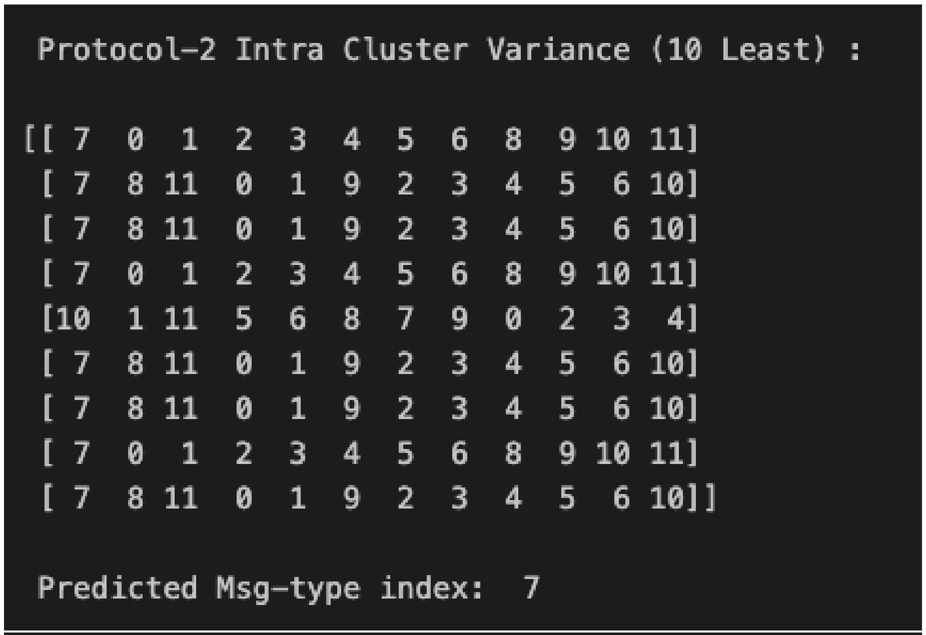
2. Latent space vector size for protocol-1,2,3 are 16, 10, and 32. We can infer that encoding dimensions are proportional to header size. K values for protocol-1,2,3 as per approximate elbow points from the graphs are 15, 9, and 16.

3. After running the entire designed algorithm, predicted message-type field indices for protocol-1,2,3 are 4,7 and 8 respectively.

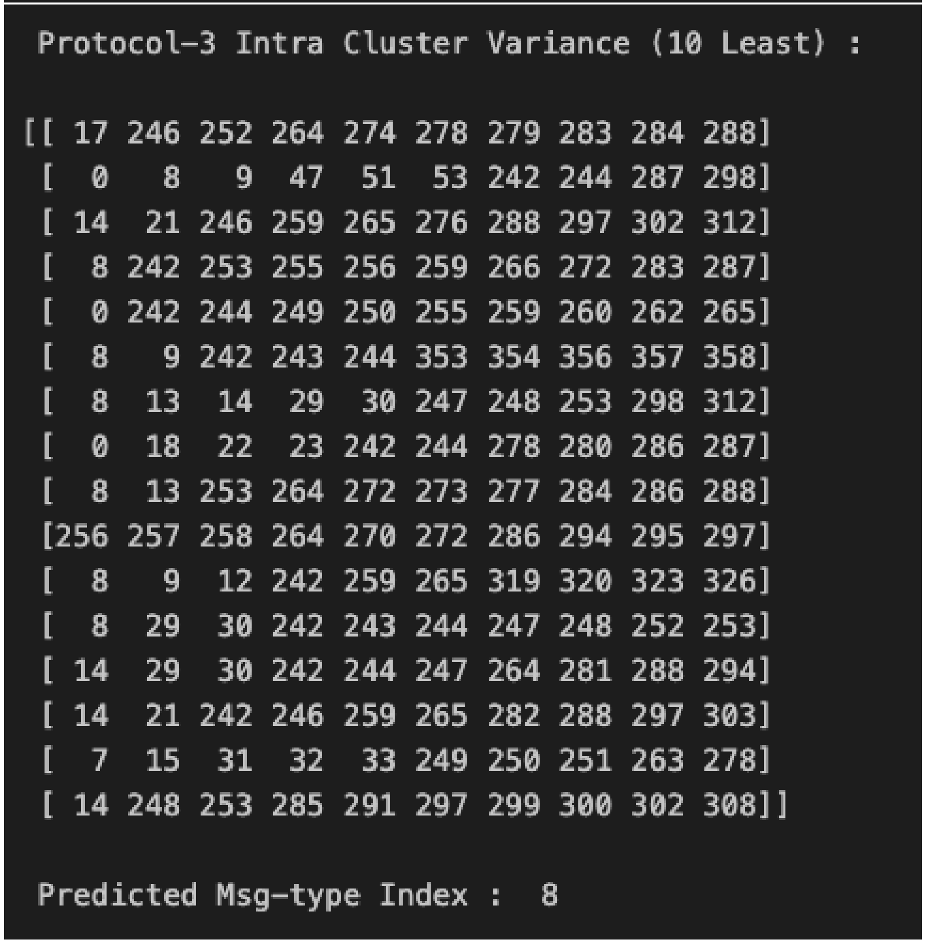
Predicted Message Type Index



```
 Protocol-1 Intra Cluster Variance (10 Least) :


[[ 4  9 10 11 21 24 26 27 28 31]
 [ 9 11 14 17 18 20 21 24 27 28]
 [ 5 10 14 15 17 18 20 21 27 28]
 [ 4  5  8  9 20 21 24 27 28 31]
 [ 0  4  9 10 19 20 21 22 26 27]
 [ 0  9 22 23 24 25 26 27 28 29]
 [10 14 15 16 17 18 19 20 21 28]
 [ 4 10 11 14 17 20 21 24 25 28]
 [10 14 15 16 18 19 21 24 28 29]
 [ 5  8 10 12 13 22 23 26 27 28]
 [ 5  8 10 11 12 13 22 26 28 31]
 [ 5  6 10 11 13 20 21 22 24 28]
 [10 13 14 16 17 18 20 21 24 28]
 [ 4  5  8 10 11 25 26 27 28 31]
 [ 4  5  8  9 20 21 24 27 28 31]]

 Predicted Msg-type index:  4
```



```
Protocol-2 Intra Cluster Variance (10 Least) :

[[ 7  0  1  2  3  4  5  6  8  9 10 11]
 [ 7  8 11  0  1  9  2  3  4  5  6 10]
 [ 7  8 11  0  1  9  2  3  4  5  6 10]
 [ 7  0  1  2  3  4  5  6  8  9 10 11]
 [10  1 11  5  6  8  7  9  0  2  3  4]
 [ 7  8 11  0  1  9  2  3  4  5  6 10]
 [ 7  8 11  0  1  9  2  3  4  5  6 10]
 [ 7  0  1  2  3  4  5  6  8  9 10 11]
 [ 7  8 11  0  1  9  2  3  4  5  6 10]]

Predicted Msg-type index:  7
```



```
Protocol-3 Intra Cluster Variance (10 Least) :

[[ 17 246 252 264 274 278 279 283 284 288]
 [  0   8   9  47  51  53 242 244 287 298]
 [ 14  21 246 259 265 276 288 297 302 312]
 [  8 242 253 255 256 259 266 272 283 287]
 [  0 242 244 249 250 255 259 260 262 265]
 [  8   9 242 243 244 353 354 356 357 358]
 [  8  13  14  29  30 247 248 253 298 312]
 [  0  18  22  23 242 244 278 280 286 287]
 [  8  13 253 264 272 273 277 284 286 288]
 [256 257 258 264 270 272 286 294 295 297]
 [  8   9  12 242 259 265 319 320 323 326]
 [  8  29  30 242 243 244 247 248 252 253]
 [ 14  29  30 242 244 247 264 281 288 294]
 [ 14  21 242 246 259 265 282 288 297 303]
 [  7  15  31  32  33 249 250 251 263 278]
 [ 14 248 253 285 291 297 299 300 302 308]]

Predicted Msg-type Index :  8
```