Persistence Detection in Slew Dark Data Using Autoencoders

Presenter: Amirmohammad Chegeni
amirmohammad.chegeni@unipd.it

Department of Physics and Astronomy "Galileo Galilei"
April 15th, 2025

Outline

- ❑ **Data Preprocessing Pipeline (CALBLOCK-F-006)**
- ❑ **Training Process**
  - **Autoencoders**
  - **Training data and Optimization**
  - **Reconstruction loss**
- ❑ **Latent Space Analysis**
  - **UMAP for Dimension reduction**
  - **GMM for Clustering UMAP Space**
  - **Number of Clusters Challenge**
  - **Data Diversity Vs XAI**
- ❑ **Slew dark Persistence Unsupervise Recognition across Channels**
- ❑ **Segmentation Pipeline**
- ❑ **Next Steps …**

Data Processing Pipeline

Retrieving Data from DPS

Data Product	**DpdNispRawFrame**
CalblockId	**CALBLOCK-F-006**
Date time	**From 2025/01/10 to 2025/01/13**
Data.FilterWheelPos	**CLOSED**
Detector	**11**

ADU Conversion

[ADU] to [ADU/frame/px]

Data Quality + Bad Pixels + Reference Pixels

Clipping

Max = - 0.09

Min = 0.1

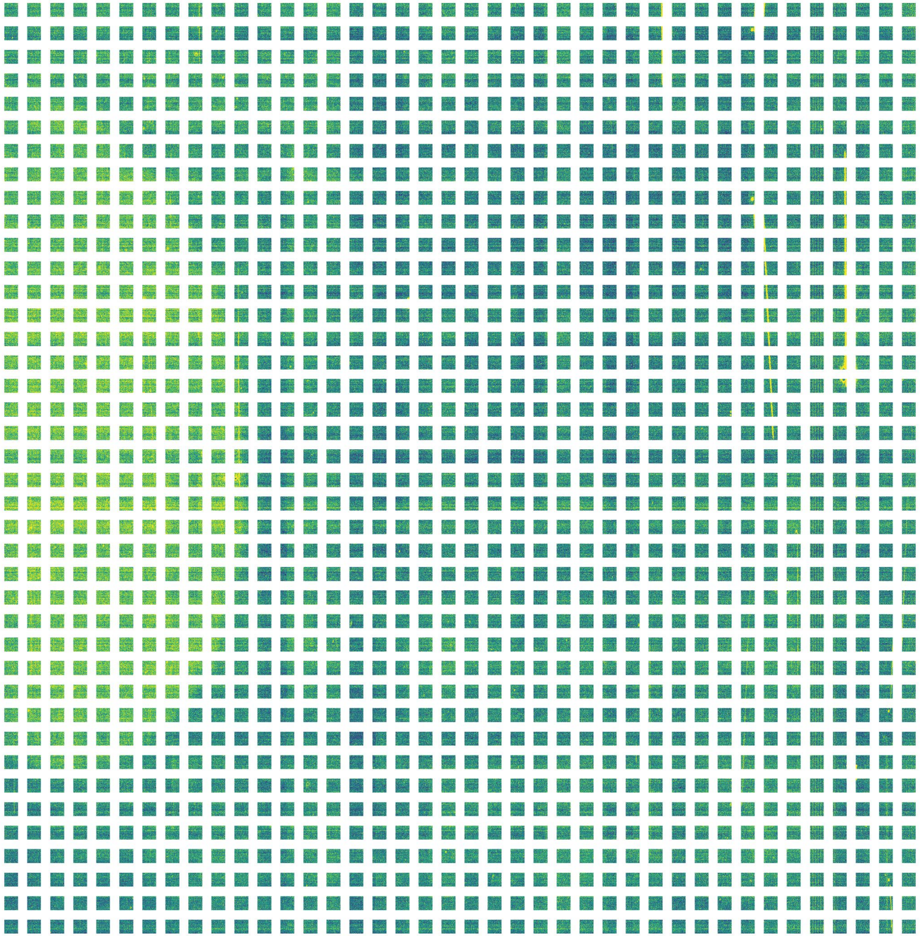
Normalizing

[0,1]

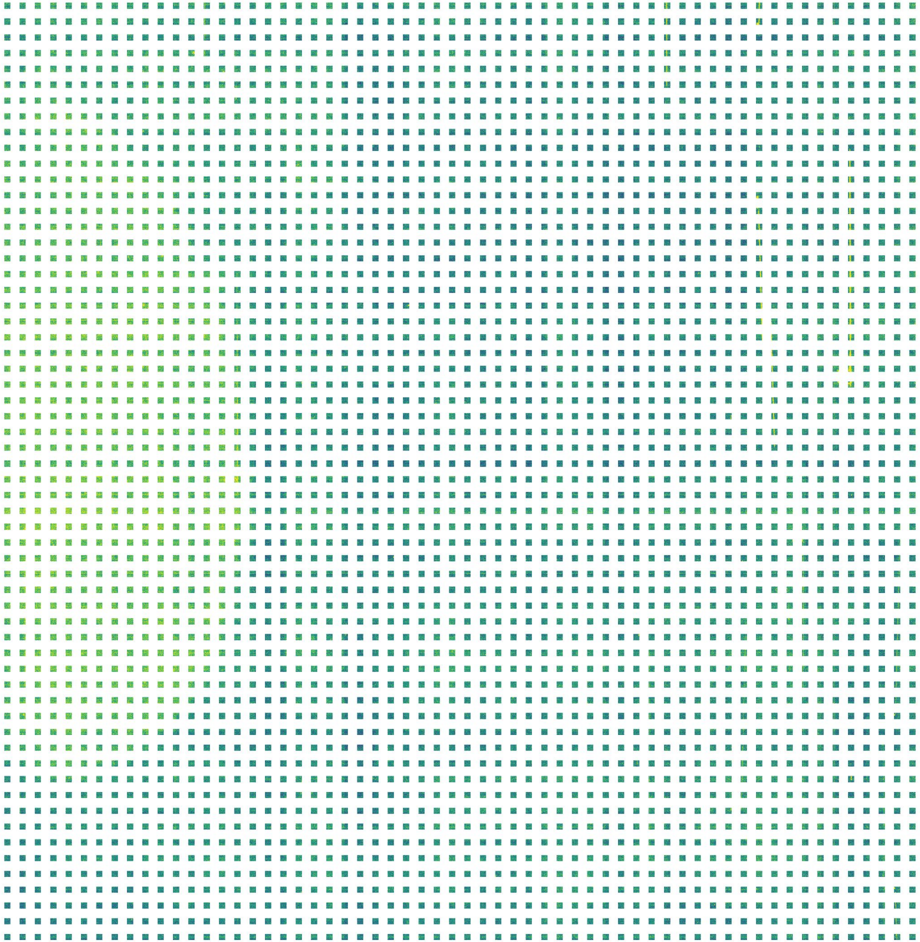
Dividing into Equal Square Patches


136*136 Patches
68*68 Patches

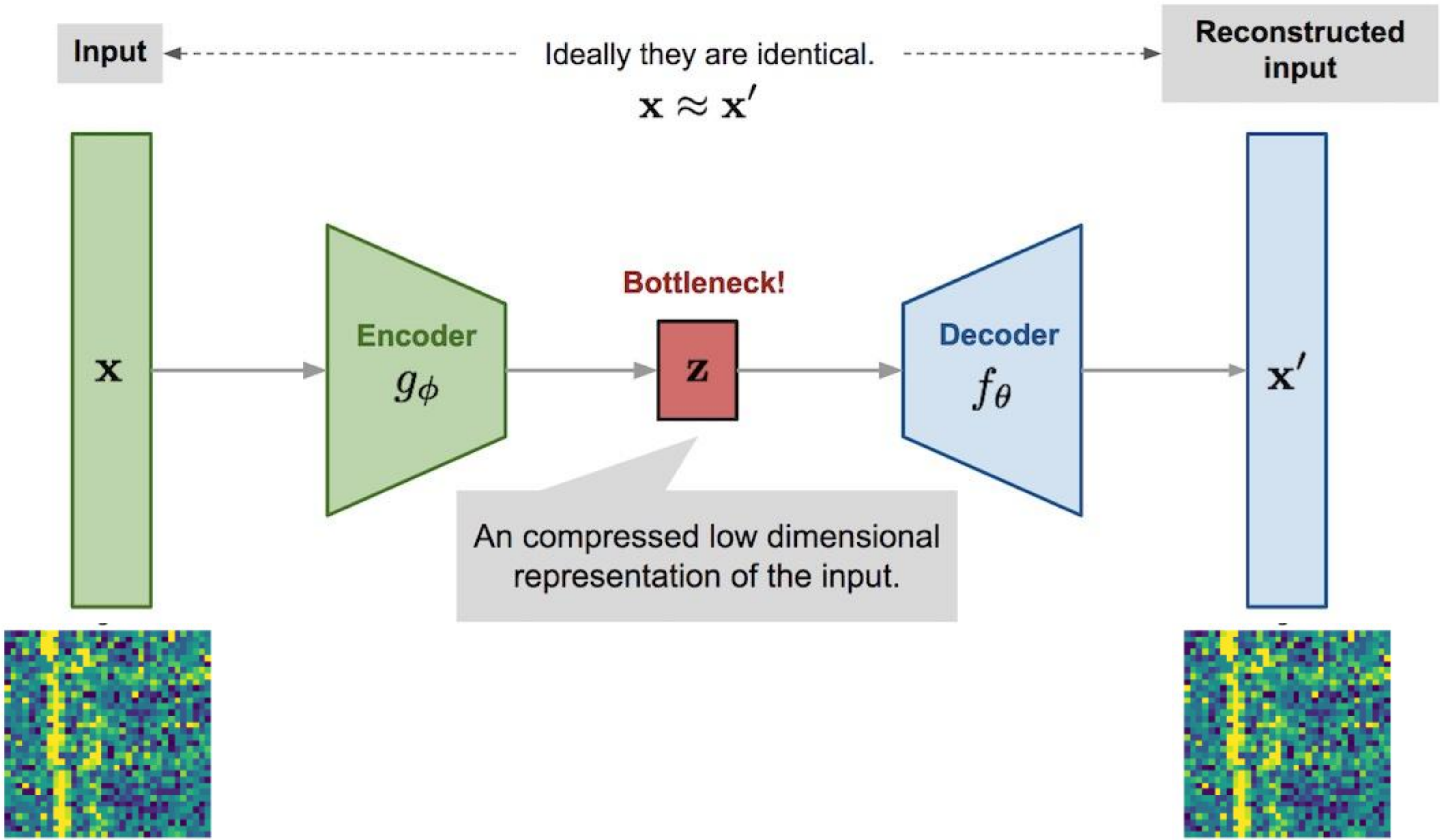
51*51 Patches



34*34 Patches



Training Process - Autoencoders



Training Process - Training data and Optimization

Total Data Shape
(192960,34,34)

70% Training data
15% Validation data
15% Test data

Consider a Simple AE Architecture

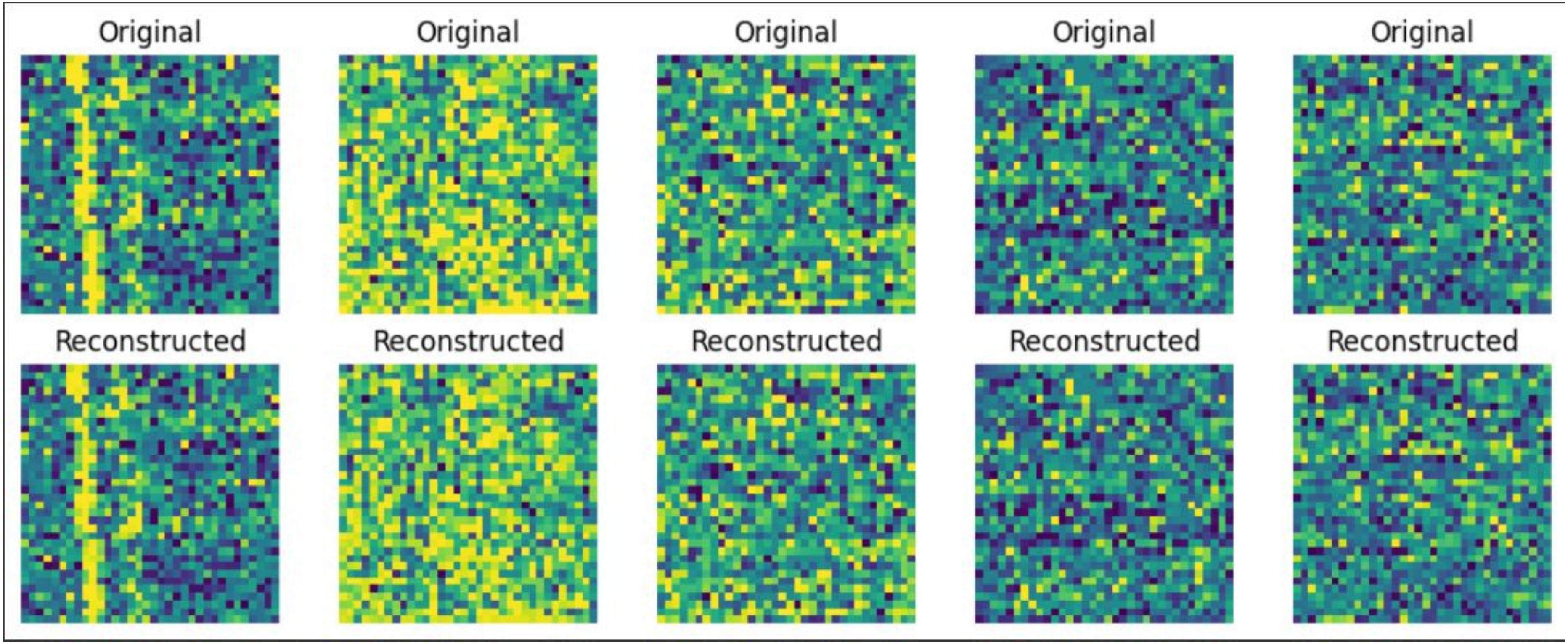
Optimize the AE parameters using Optuna library

Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 136, 136, 1)	0
conv2d (Conv2D)	(None, 136, 136, 16)	160
max_pooling2d (MaxPooling2D)	(None, 68, 68, 16)	0
conv2d_1 (Conv2D)	(None, 68, 68, 32)	4,640
max_pooling2d_1 (MaxPooling2D)	(None, 34, 34, 32)	0
conv2d_2 (Conv2D)	(None, 34, 34, 64)	18,496
conv2d_3 (Conv2D)	(None, 34, 34, 64)	36,928
up_sampling2d (UpSampling2D)	(None, 68, 68, 64)	0
conv2d_4 (Conv2D)	(None, 68, 68, 32)	18,464
up_sampling2d_1 (UpSampling2D)	(None, 136, 136, 32)	0
conv2d_5 (Conv2D)	(None, 136, 136, 1)	289

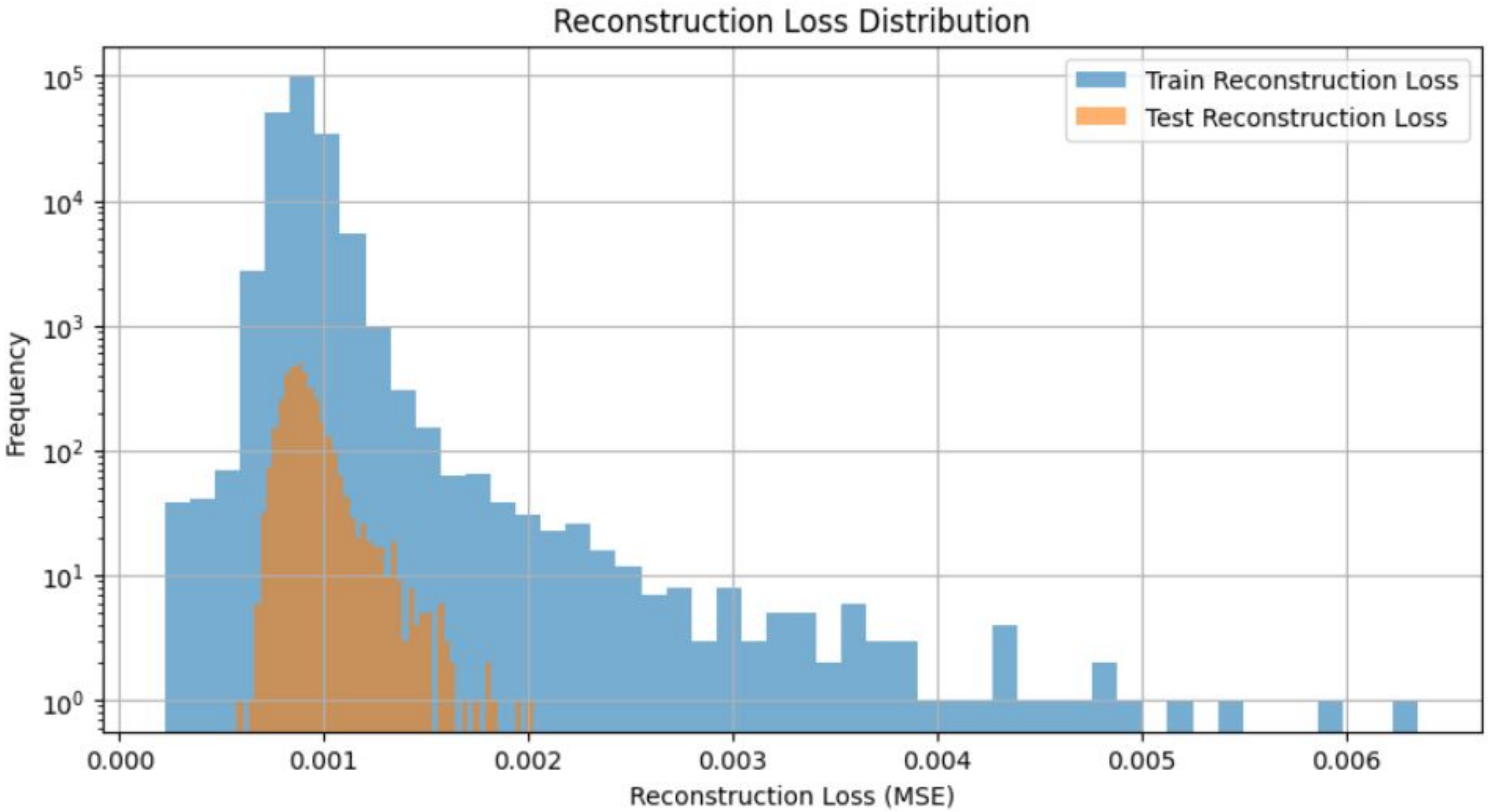
Total params: 78,977 (308.50 KB)
Trainable params: 78,977 (308.50 KB)
Non-trainable params: 0 (0.00 B)

ModelCheckpoint
EarlyStopping
ReduceLROnPlateau

Training Process - Reconstruction loss



Training Process - Reconstruction loss



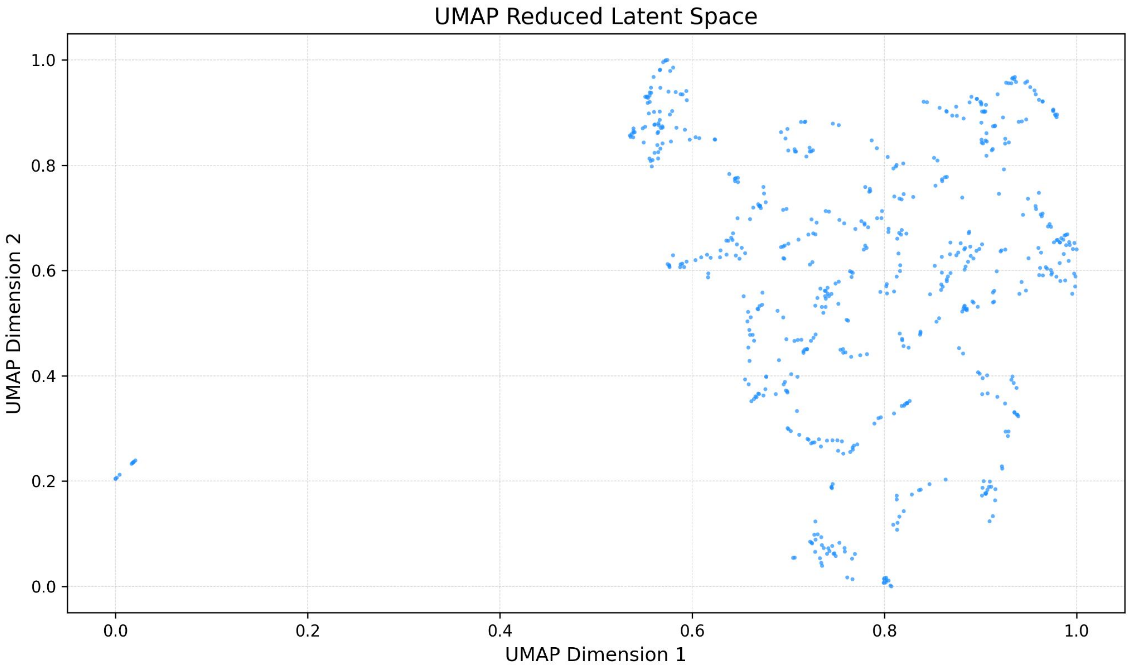
Latent Space Analysis - UMAP as A Dimensionality Reduction Tool

Latent Space Shape
(34,34,64)

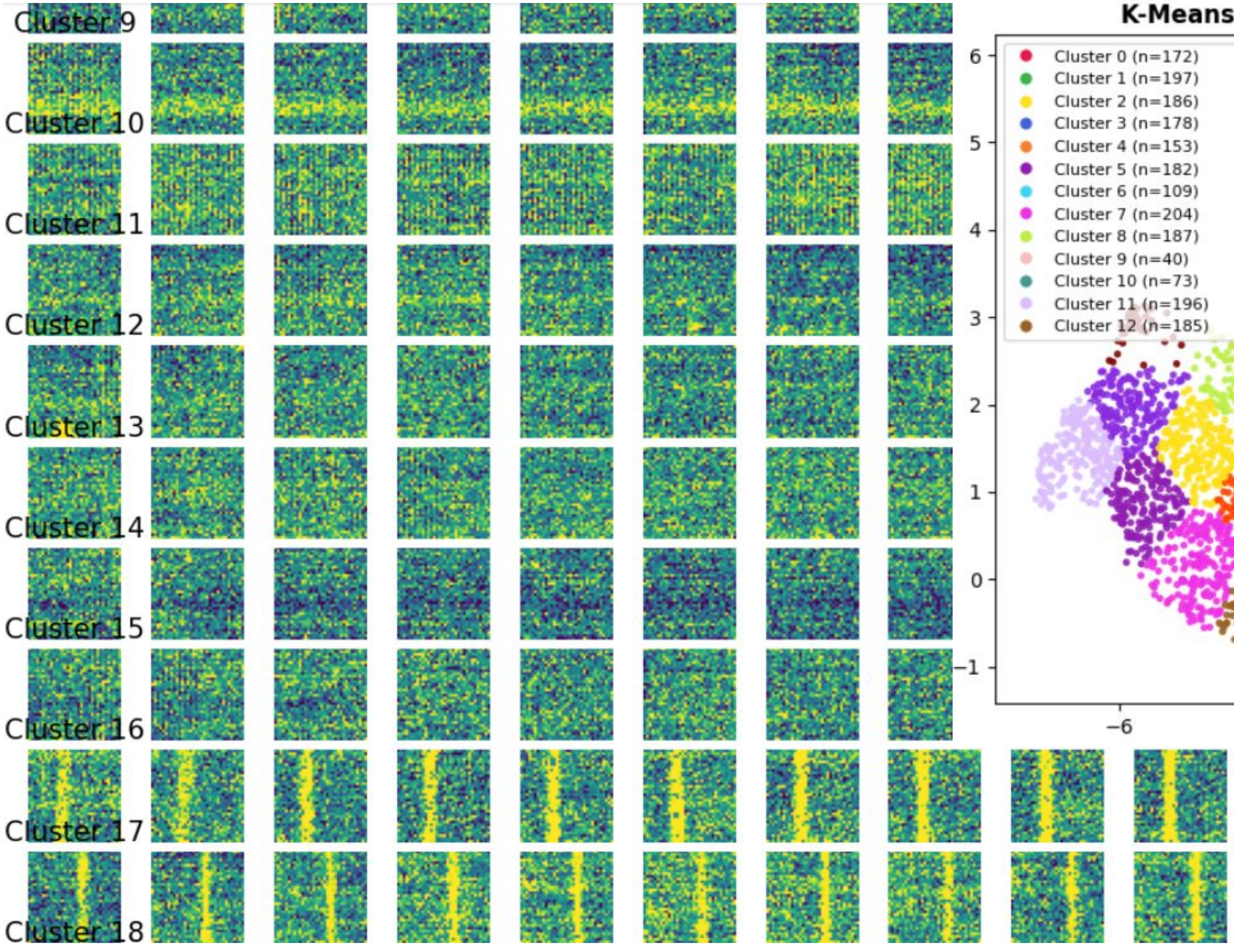
High Dimensional Space

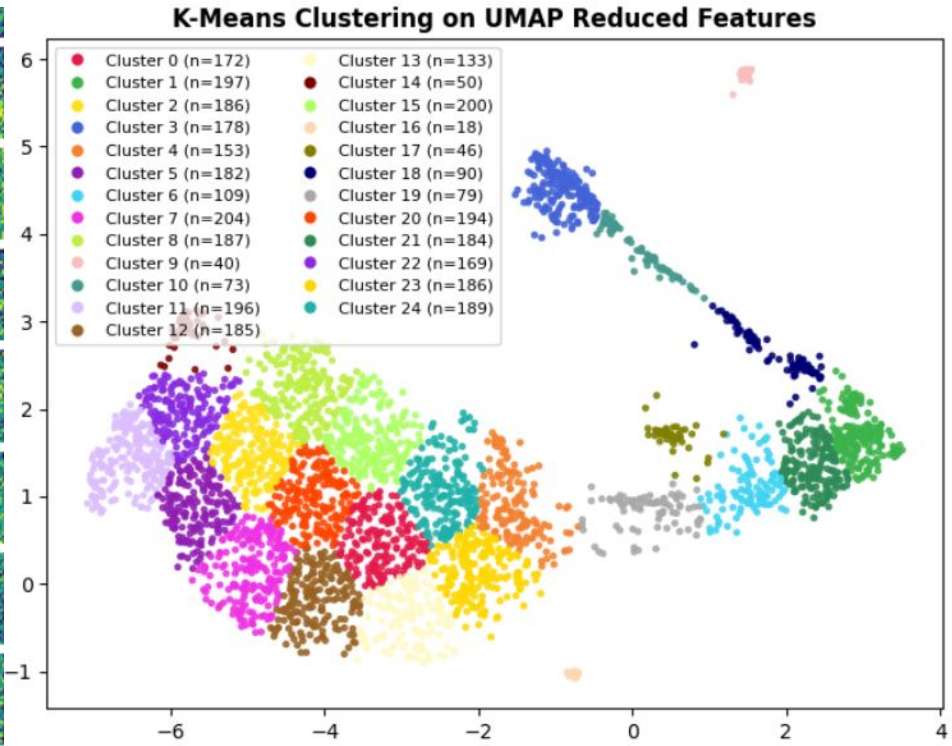
Uniform Manifold Approximation and Projection

2 Dimensional Space



Latent Space Analysis - GMM for Clustering UMAP Space



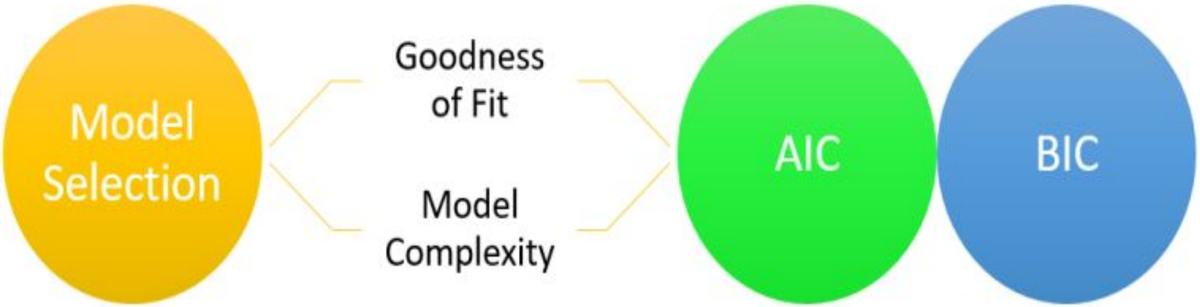


Latent Space Analysis - Gaussian Mixture Model for Clustering UMAP Space

How Many Clusters ?

AIC = Akaike Information Criterion

BIC = Bayesian Information Criterion



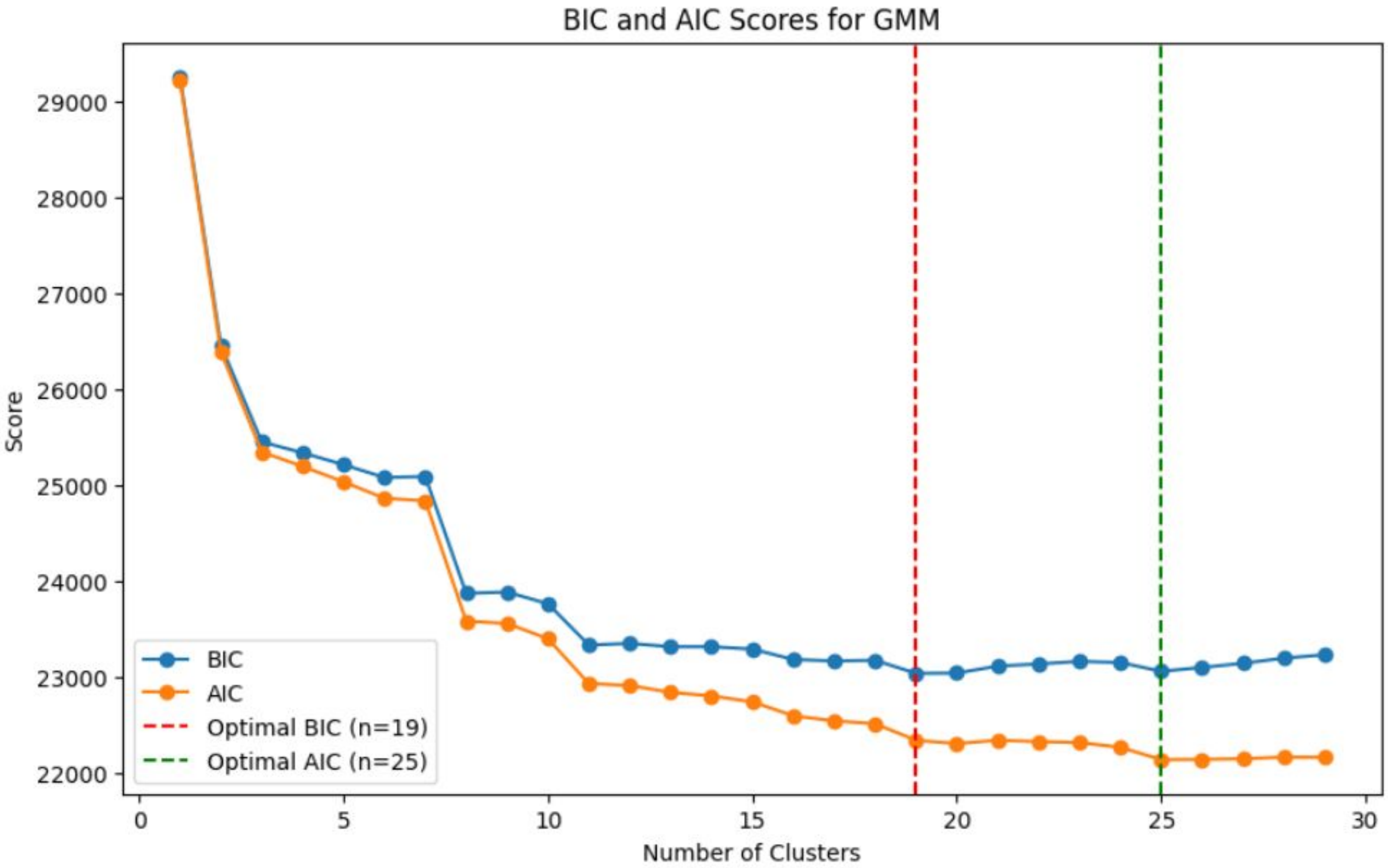
$$\text{Score} = \underbrace{kp}_{\text{Model Complexity}} - \underbrace{2\log(L)}_{\text{Model Performance}}$$

where

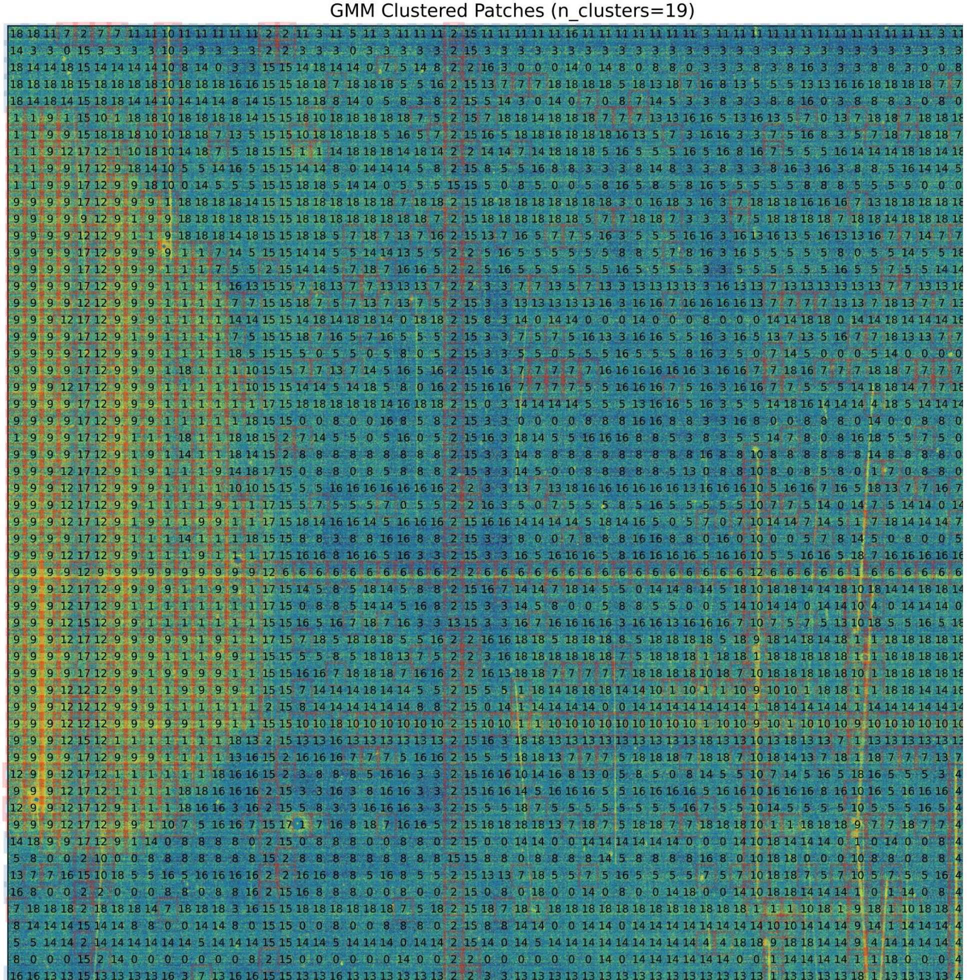
k = For AIC: 2
For BIC: log (sample-size)
L = Likelihood function (mse, log_loss)
p = No of parameters

© Shachi

Latent Space Analysis - GMM for Clustering UMAP Space

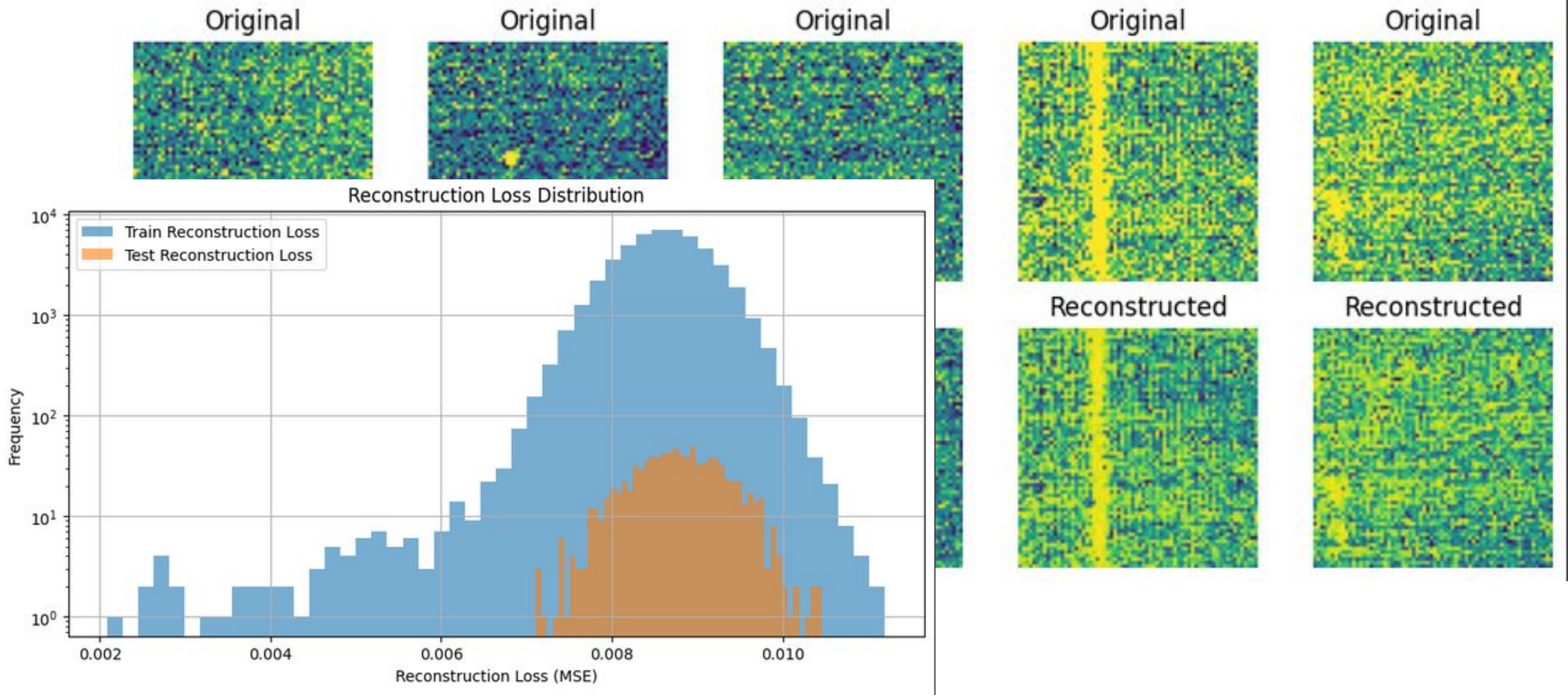


Feature Clustering - Identifying Clusters (Red Edgecolor Patches)

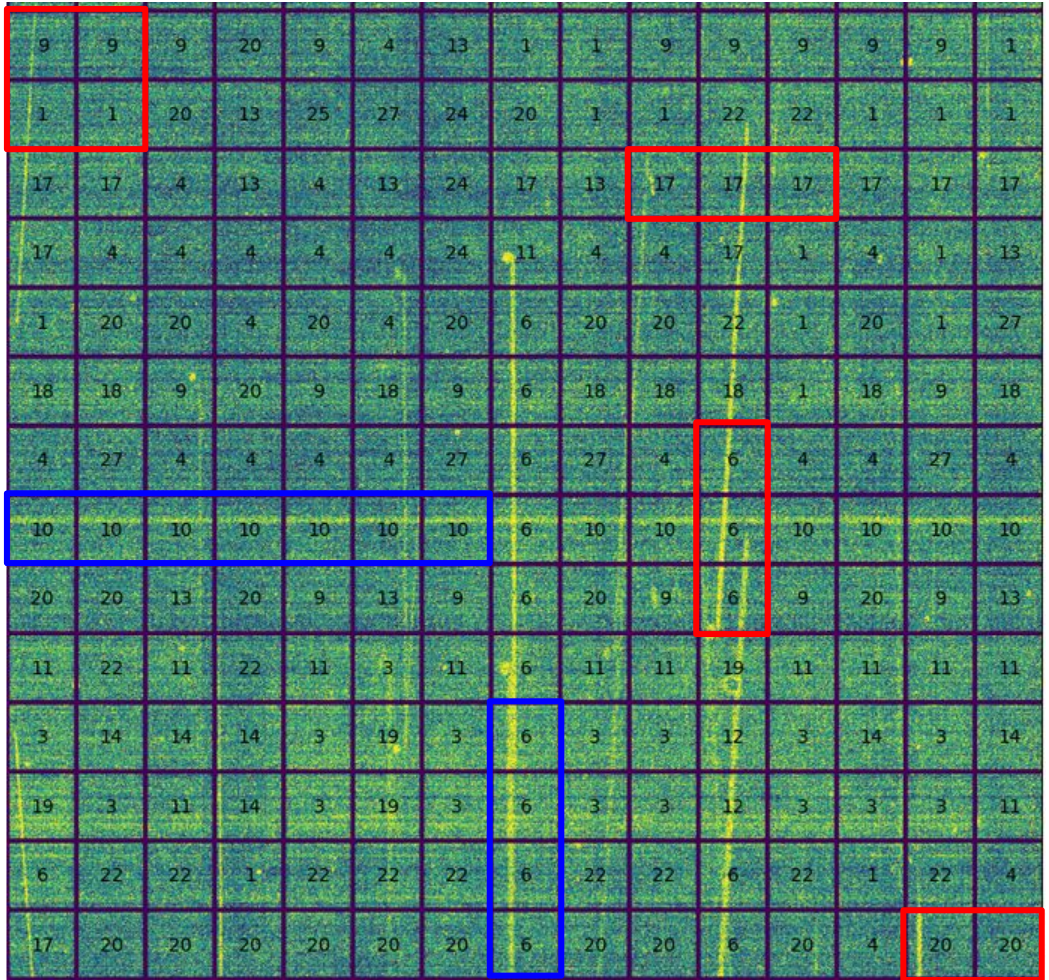


Testing Clustering Algorithm

Apply the algorithm on **1st** exposure of **2025/01/18** and detector **11**



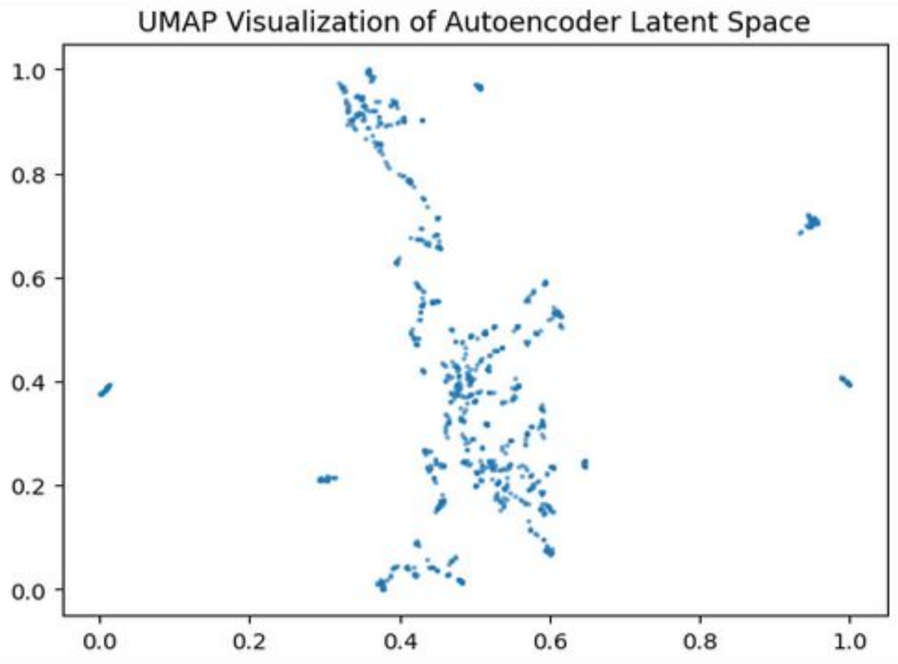
How much our clustering is effective!

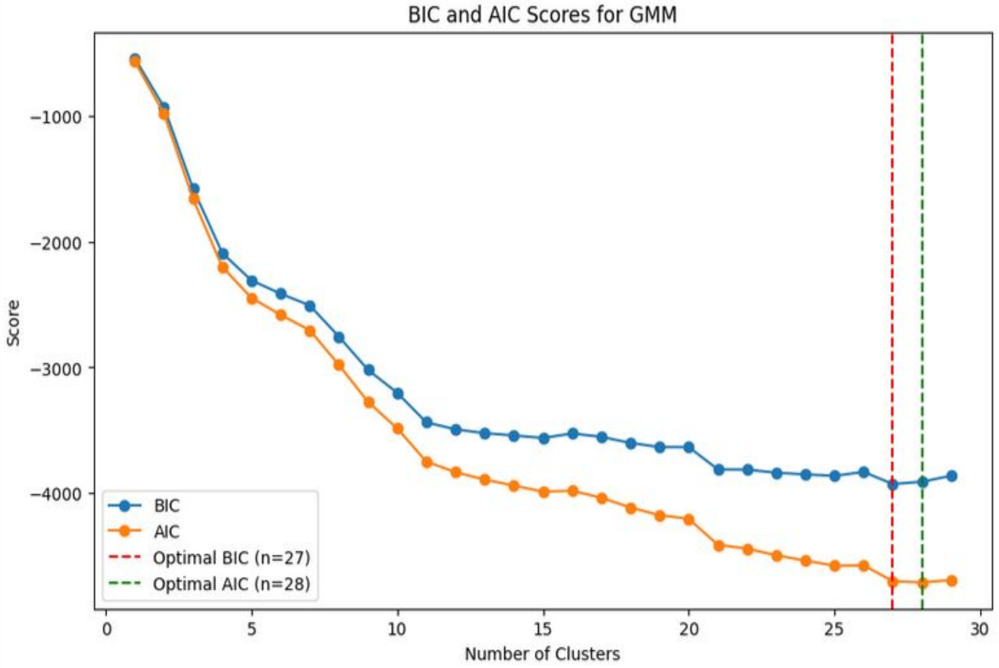


Apply GMM for clustering and the number of clusters is 29

Latent Space Clustering

1) Apply UMAP algorithm on latent space with **n_neighbors = 3**
2) Apply GMM for clustering the latent space and measure the BIC and AIC



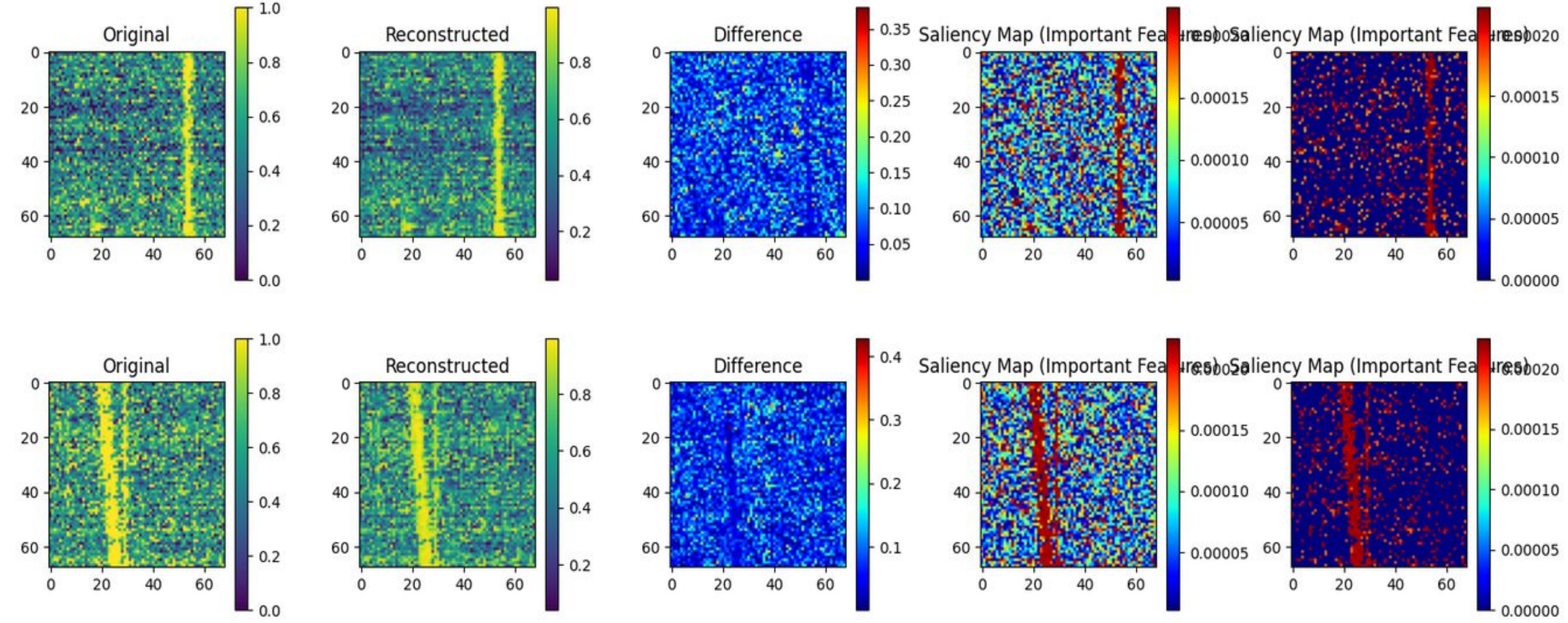


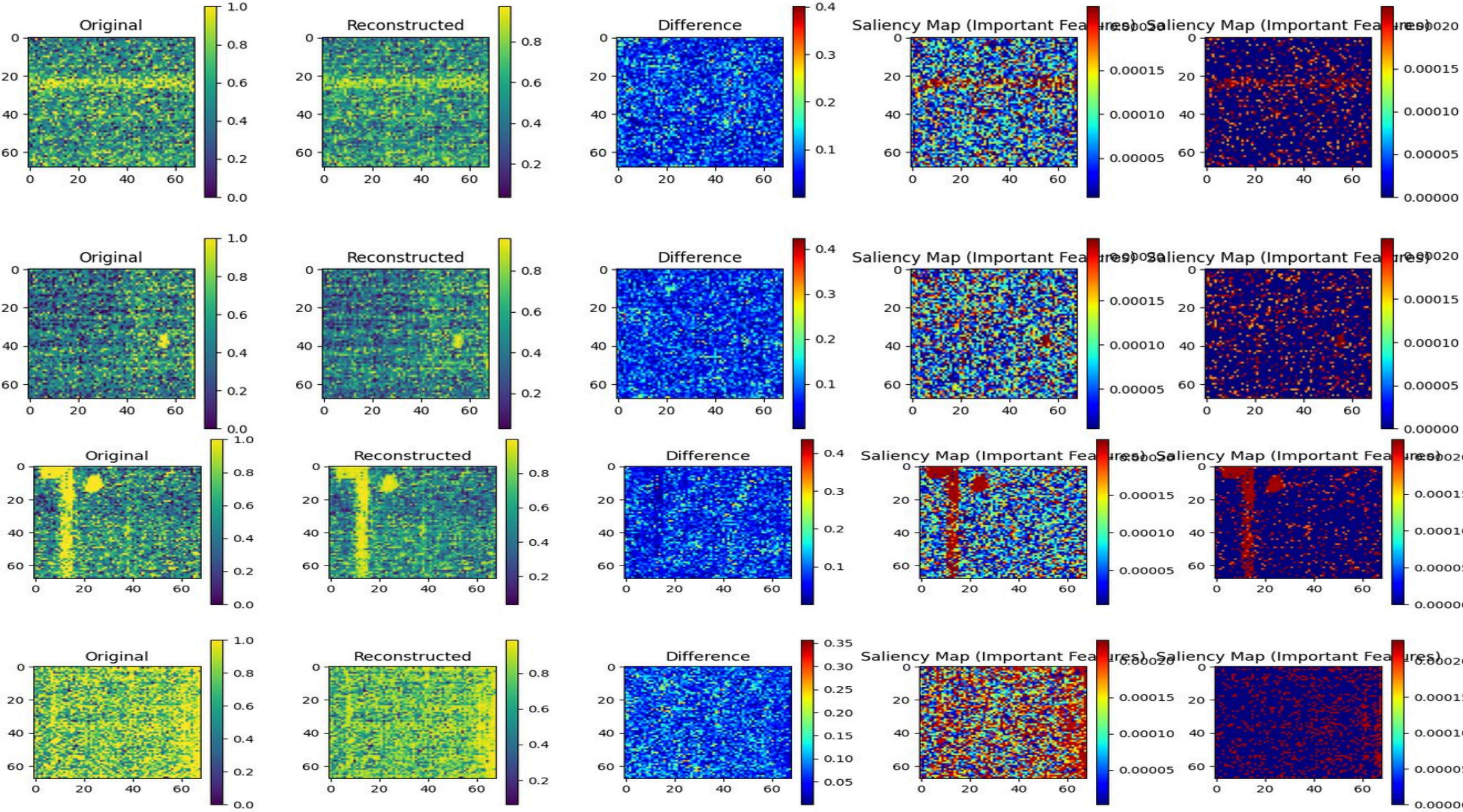
Do we need a large number of clusters for effective clustering?

Does this requirement arise from the training process, or is it a result of high data diversity?

Does the machine learn using the correct features in the data?

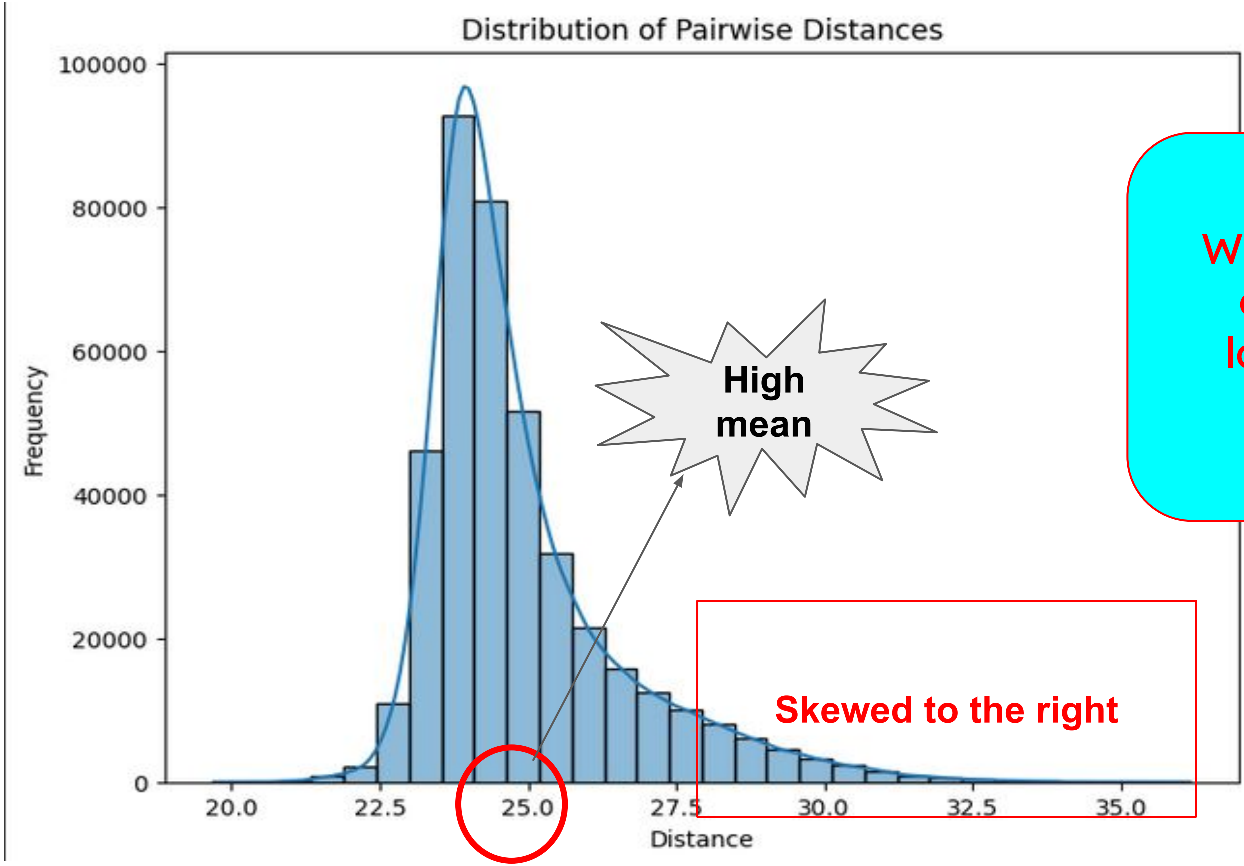
1) Apply **Gradient-based** method to find the most effective pixels in training
2) **Saliency map** highlights the most influential pixels




Original
Reconstructed
Difference
Saliency Map (Important Features)
Saliency Map (Important Features)

Calculating data diversity

1) Computes the pairwise Euclidean distances between rows of data

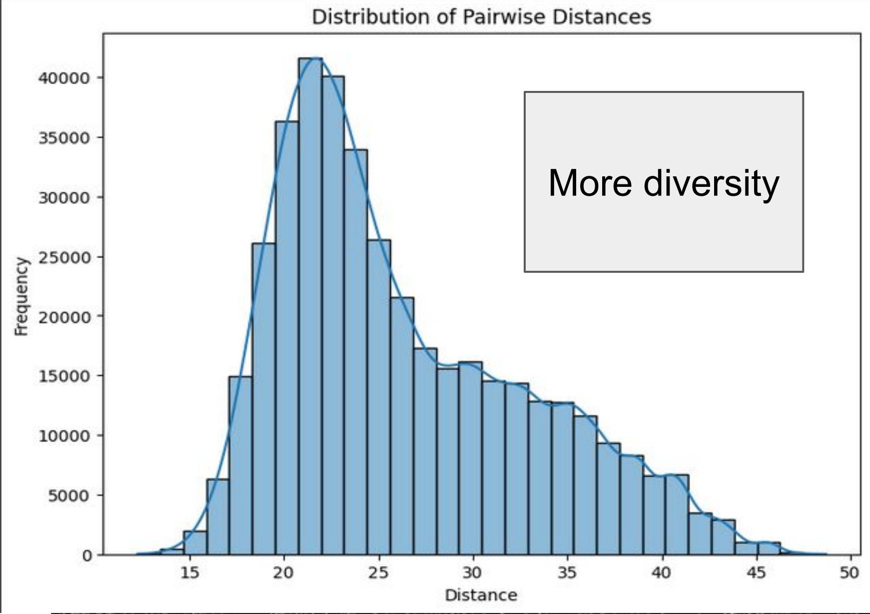


We need large number of clusters to **effectively** labelling each group of features

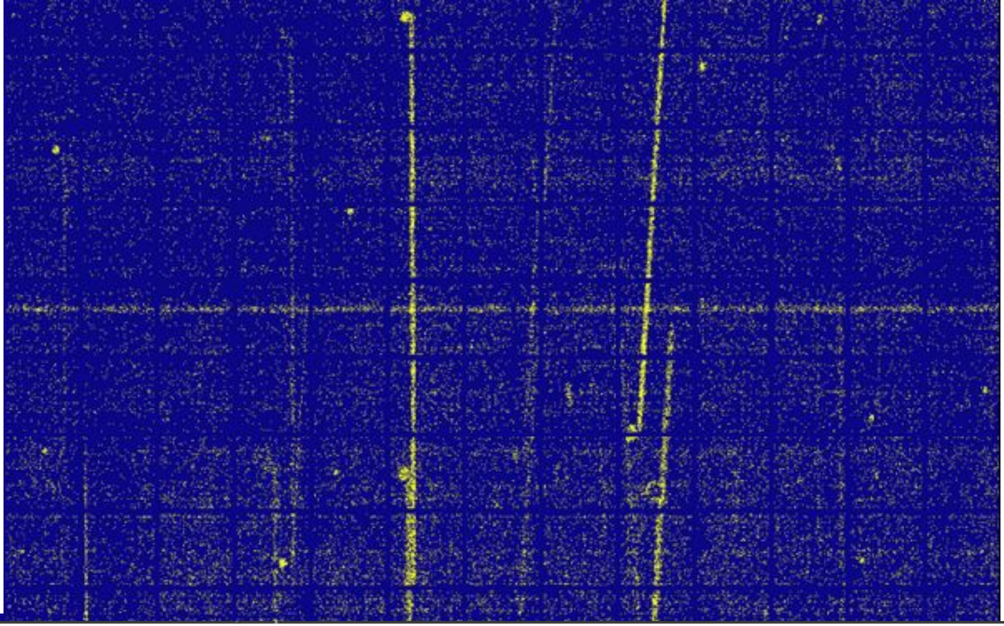
Solution 1: Removing Background to reduce the data diversity

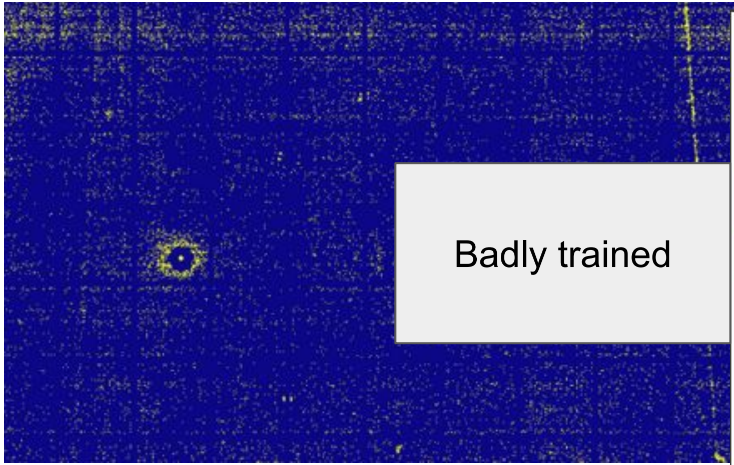
1) Removing data background should be a in a way that important features of cosmics retained!
2) Subtracting pixels with values lower that **0.95** (the data is normalized between 0 and 1)
3) Retaining pixels with values greater than **Median + (MAD*0.5)**
4) Subtracting smaller regions (clusters of connected pixels) that have fewer than **min_neighbor = 2** neighboring pixels

```python
if mad:
    med = np.median(part)
    mad_value = np.median(np.abs(part - med)) / 0.6745  # Compute MAD

    # Define positive and negative thresholds
    pos_thr = med + (mad_value*0.5)
    neg_thr = med - (mad_value*0.5)  # Fix: Should subtract instead of adding

    # Fix: Use bitwise operators instead of 'and'
    intense_mask = (part > pos_thr)
    # Retain only the intense areas
    part = np.where(intense_mask, part, 0)  # Set background to 0

if remove_mean_background:
    mask_part = part > threshold_mean
    part = part * mask_part

if remove_neighbor:
    labeled_clusters, num_labels = label(part)  # Label connected components
    refined_part = np.copy(part)

    for label_val in range(1, num_labels + 1):  # Labels start from 1
        cluster_pixels = np.where(labeled_clusters == label_val)
        if len(cluster_pixels[0]) < min_neighbor:
            refined_part[labeled_clusters == label_val] = 0  # Remove small clusters
```

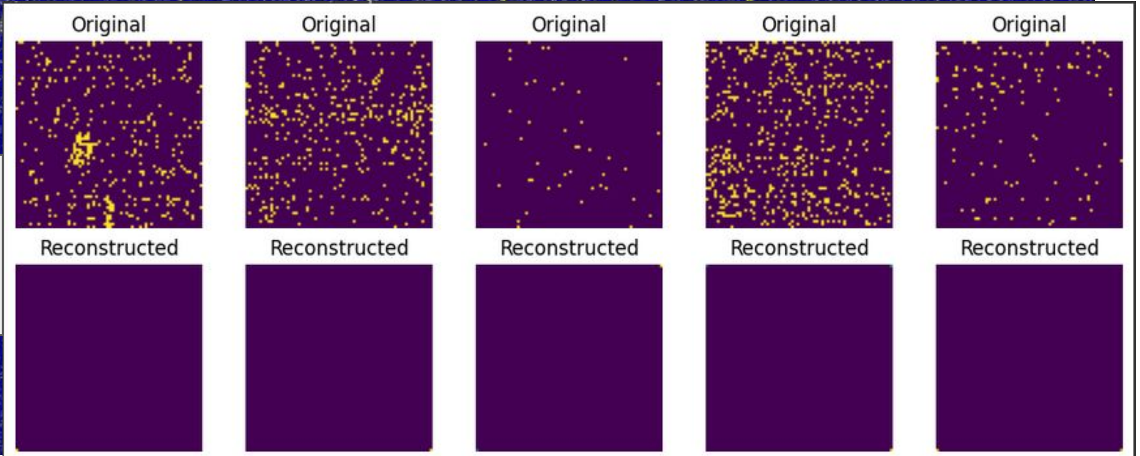


More diversity



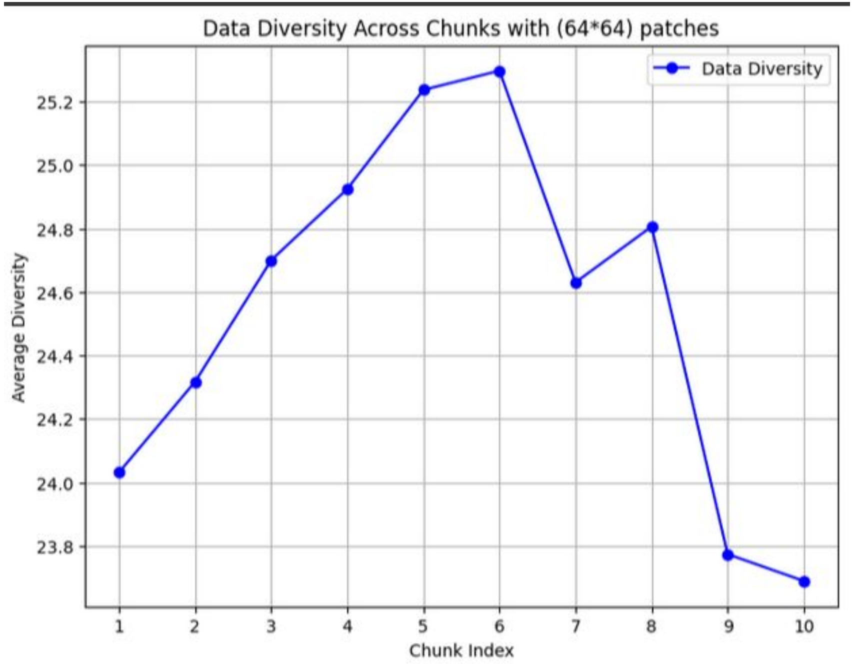


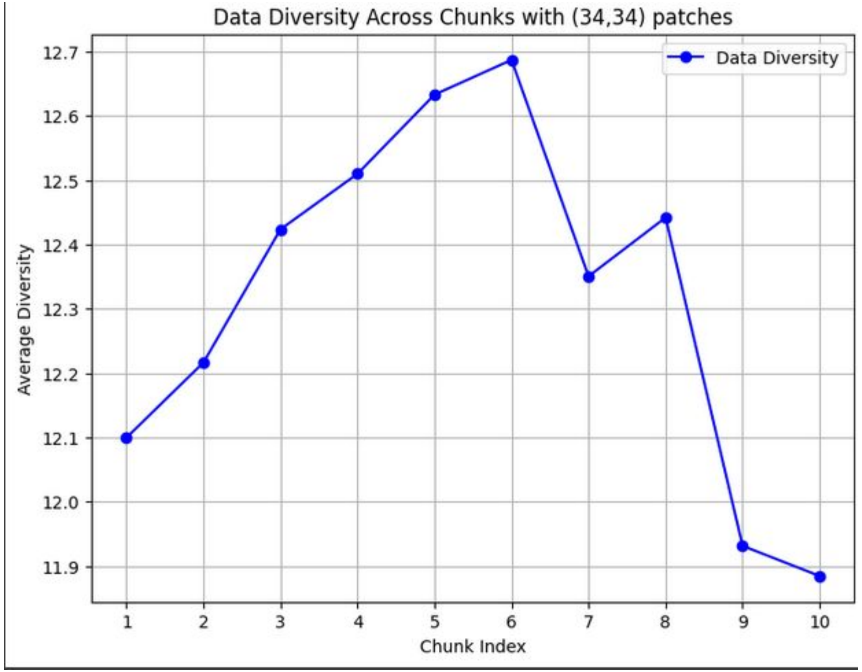
Badly trained



Solution 2: Dividing into different channels without removing background

1) Considering two different size patches including (900, 64, 64) and (3600, 34, 34)
2) Splitting each image into 10 horizontally channels to reduce the complexity in the latent space
3) Applying BIC and AIC algorithms on each channel to find the effective number of clusters (Minimum numbers of clusters, maybe we need more)
4) Applying GMM on latent space and find the clusters including cosmics

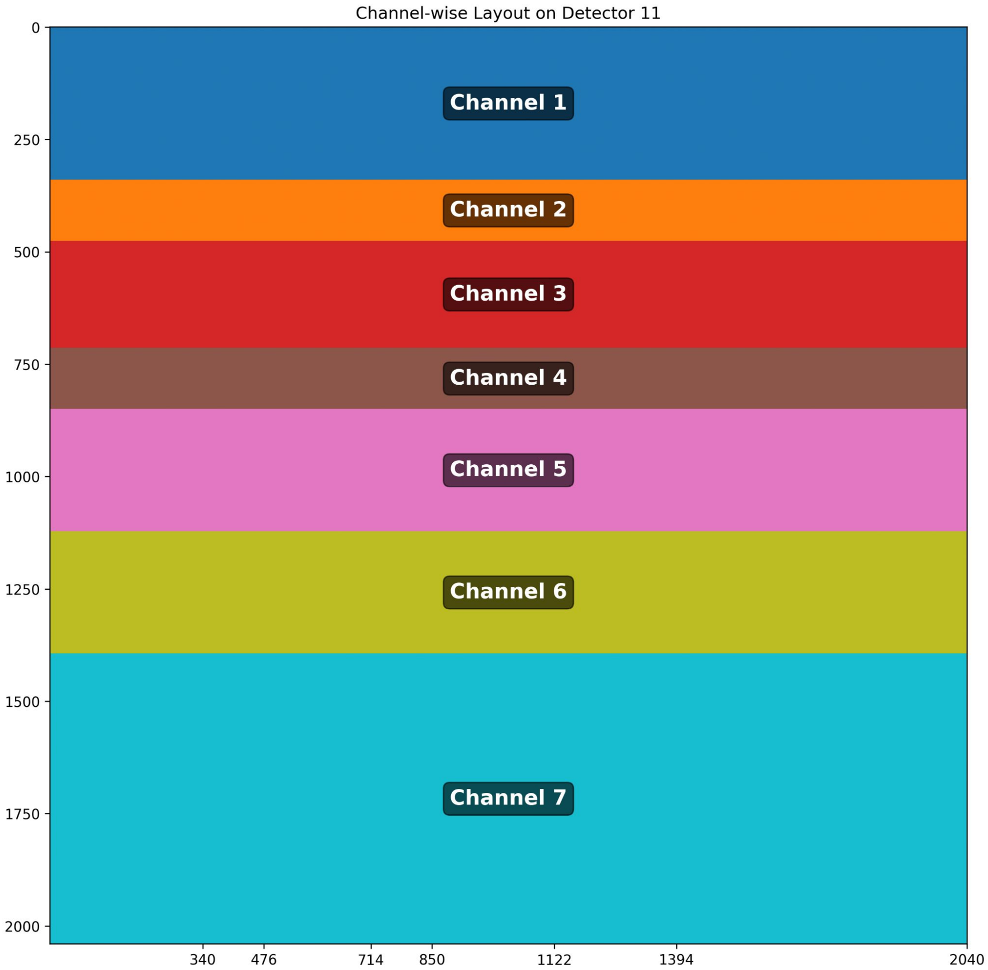




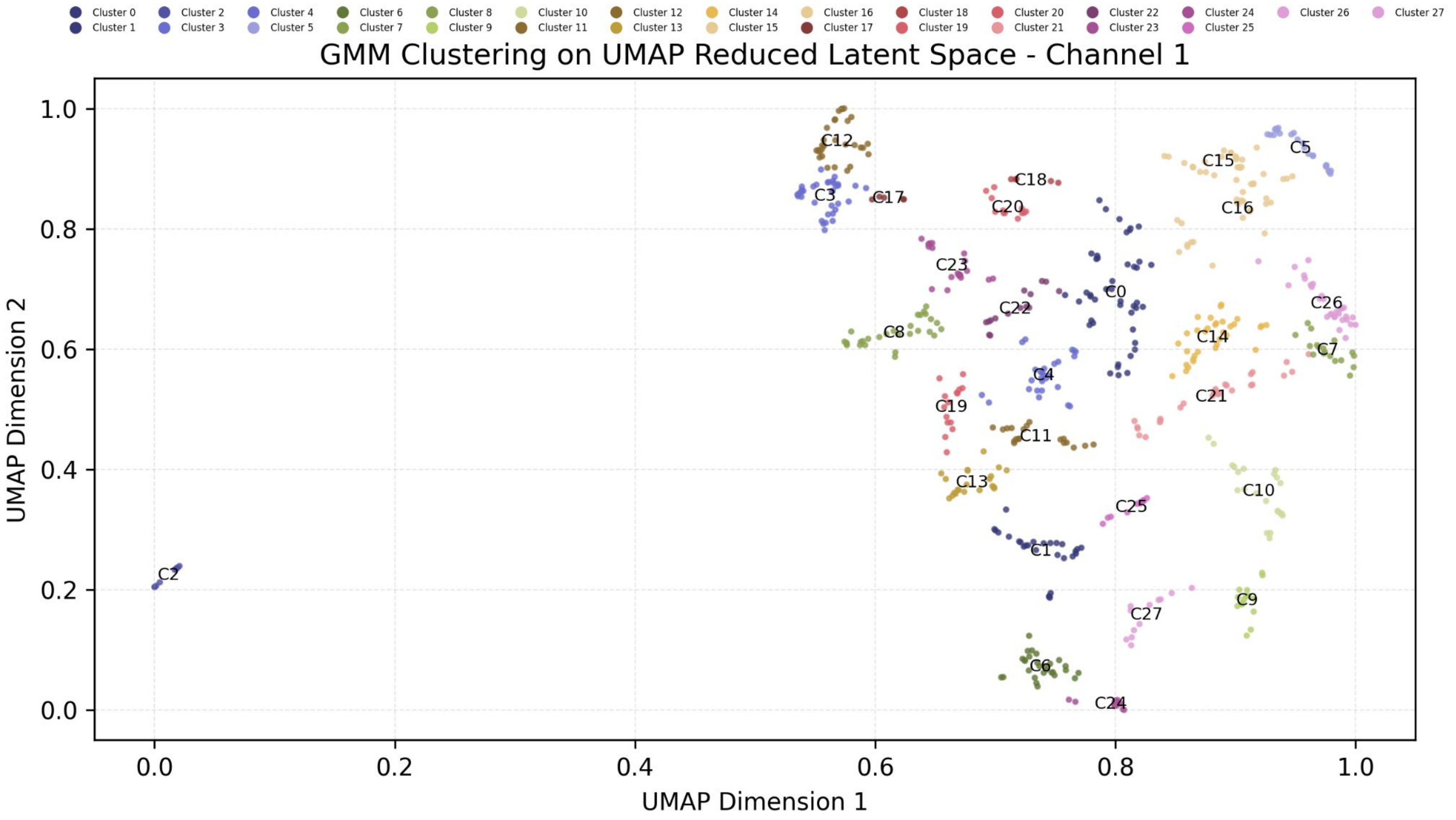
Channel-Wise Analysis Pipeline

1) Channel-Wise Horizontally Separation
2) Making latent space for each channel and applying umap to project high-dimensional latent space to 2D space
3) Applying BIC and AIC algorithms on each channel to find the effective number of clusters (This is defining Minimum numbers of clusters, maybe we need more)
4) Applying GMM on latent space and find the clusters including persistences
5) Labelling persistences clusters based on the NISP persistences features in spectro images
6) Segmenting persistences from each channel
7) Making masking maps based on this segmentation and concatenating all of the making maps

Channel-Wise Layouting

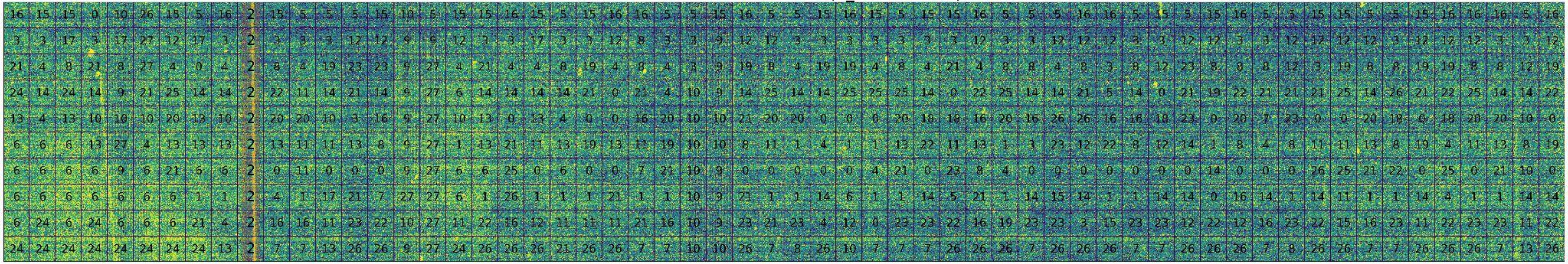


Channel 1 Latent Space Analysis

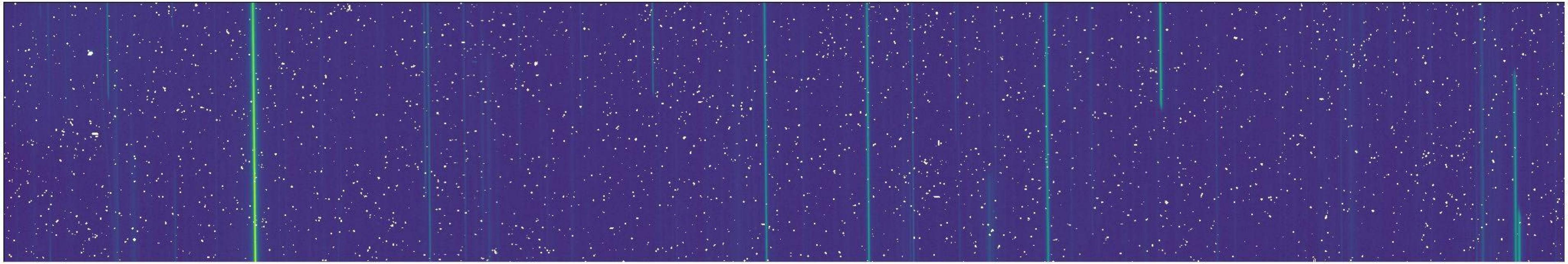


Channel 1 : Slew dark Persistence Unsupervise Recognition

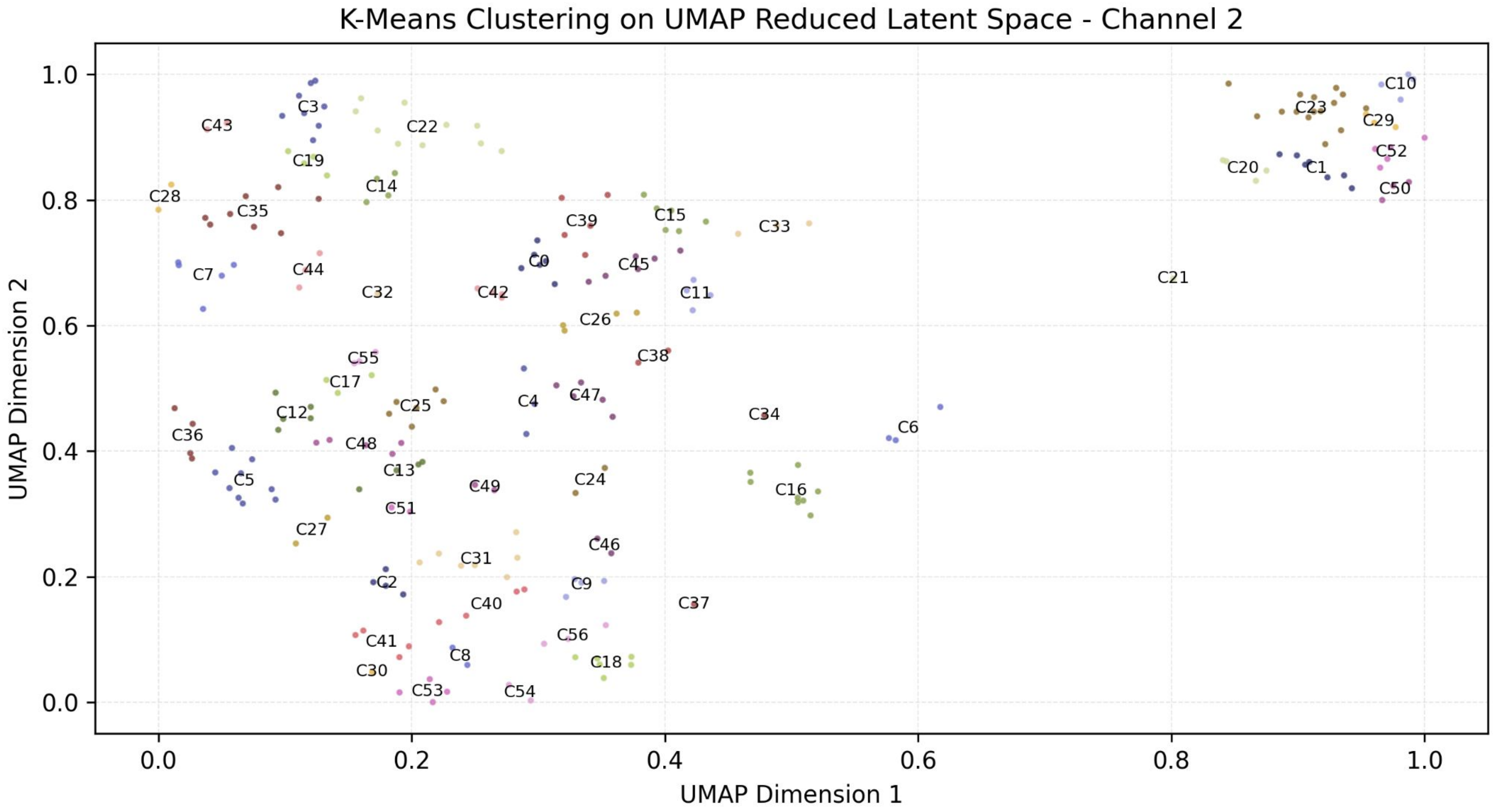
GMM Clustered Channel 1 (n_clusters=28)



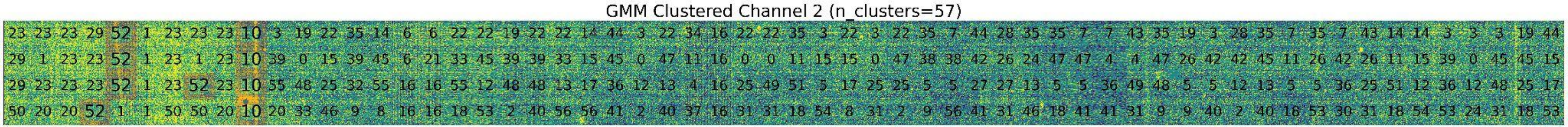
RGS180 and Tilt is 0 - Channel 1

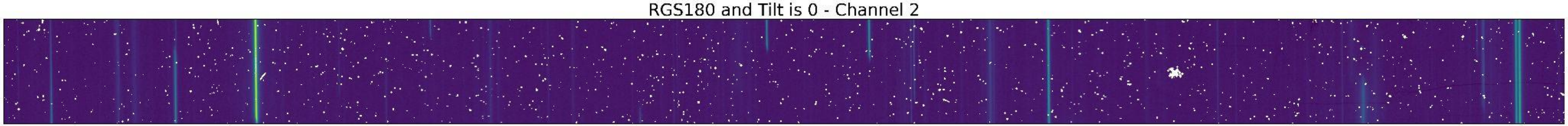


Channel 2 Latent Space Analysis

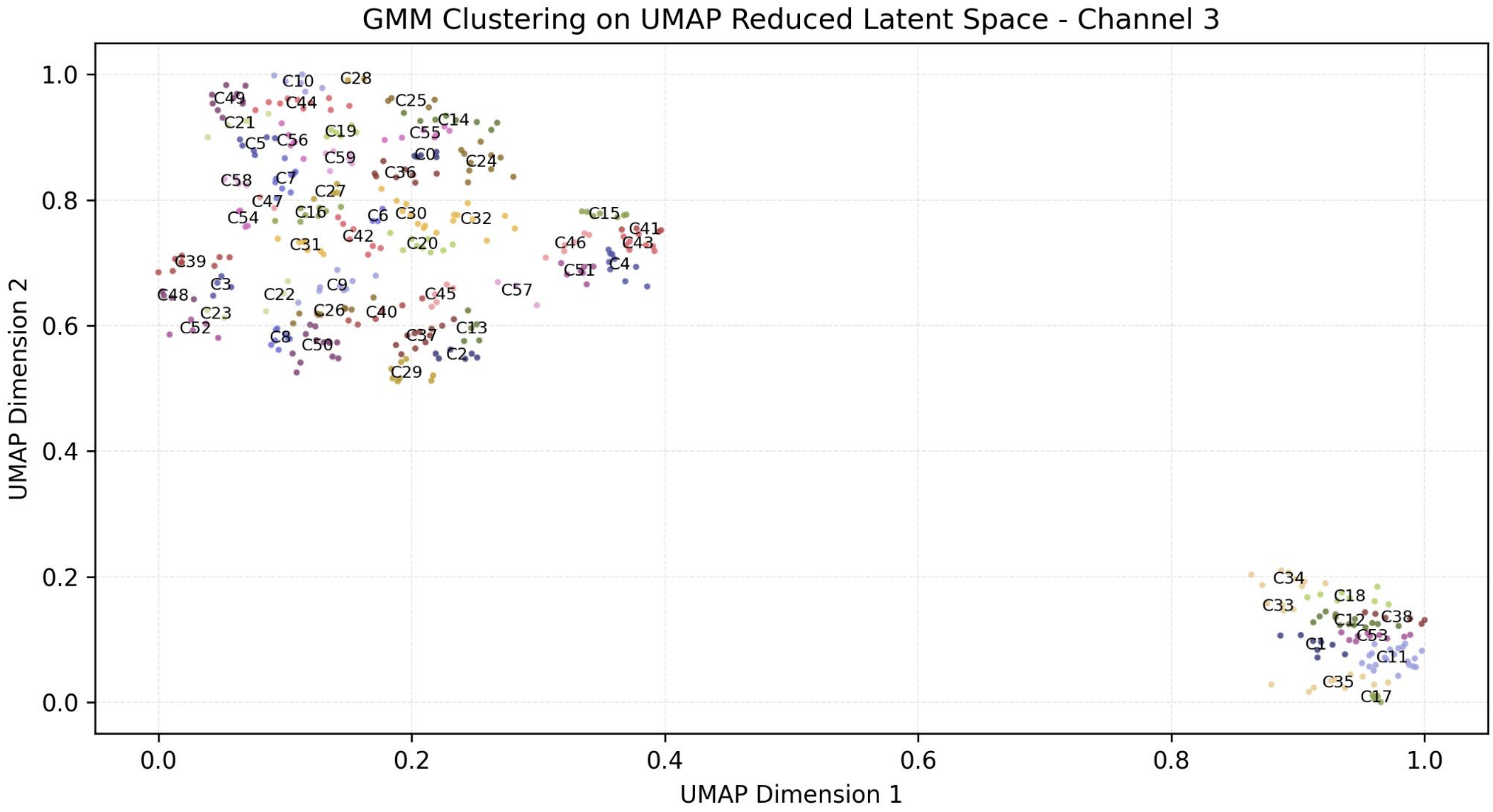


Channel 2 : Slew dark Persistence Unsupervise Recognition



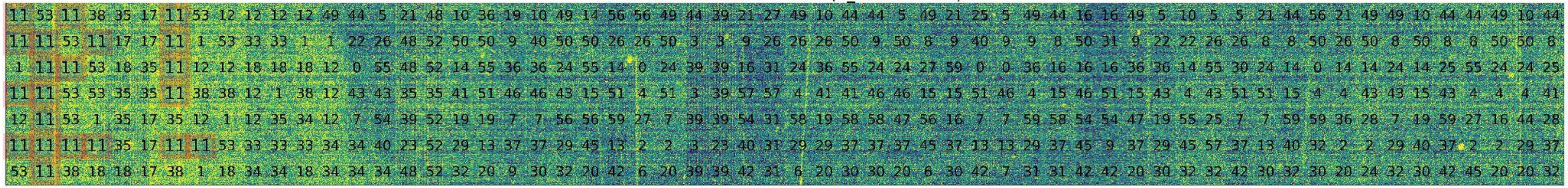


Channel 3 Latent Space Analysis

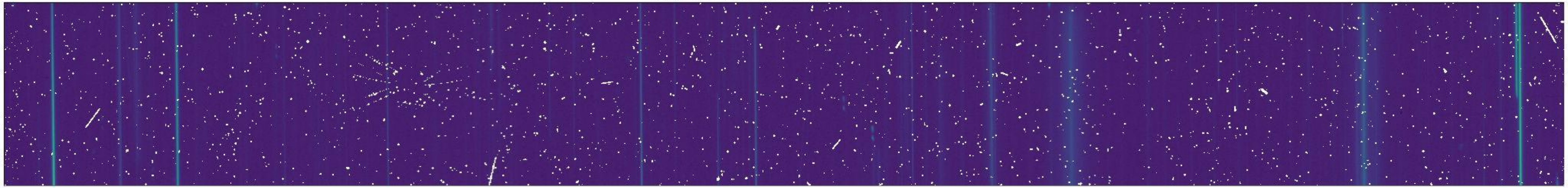


Channel 3 : Slew dark Persistence Unsupervise Recognition

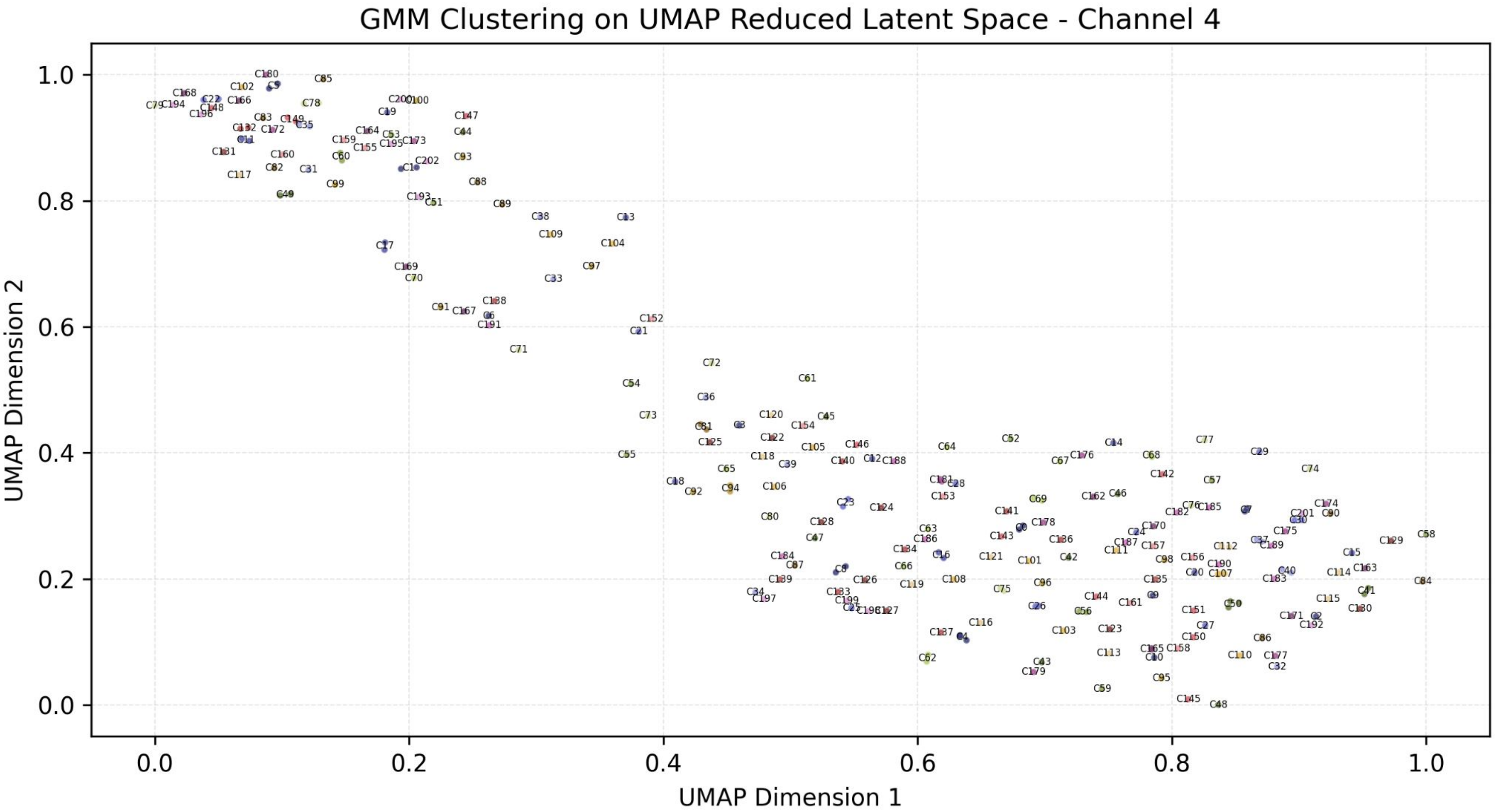
GMM Clustered Channel 3 (n_clusters=60)



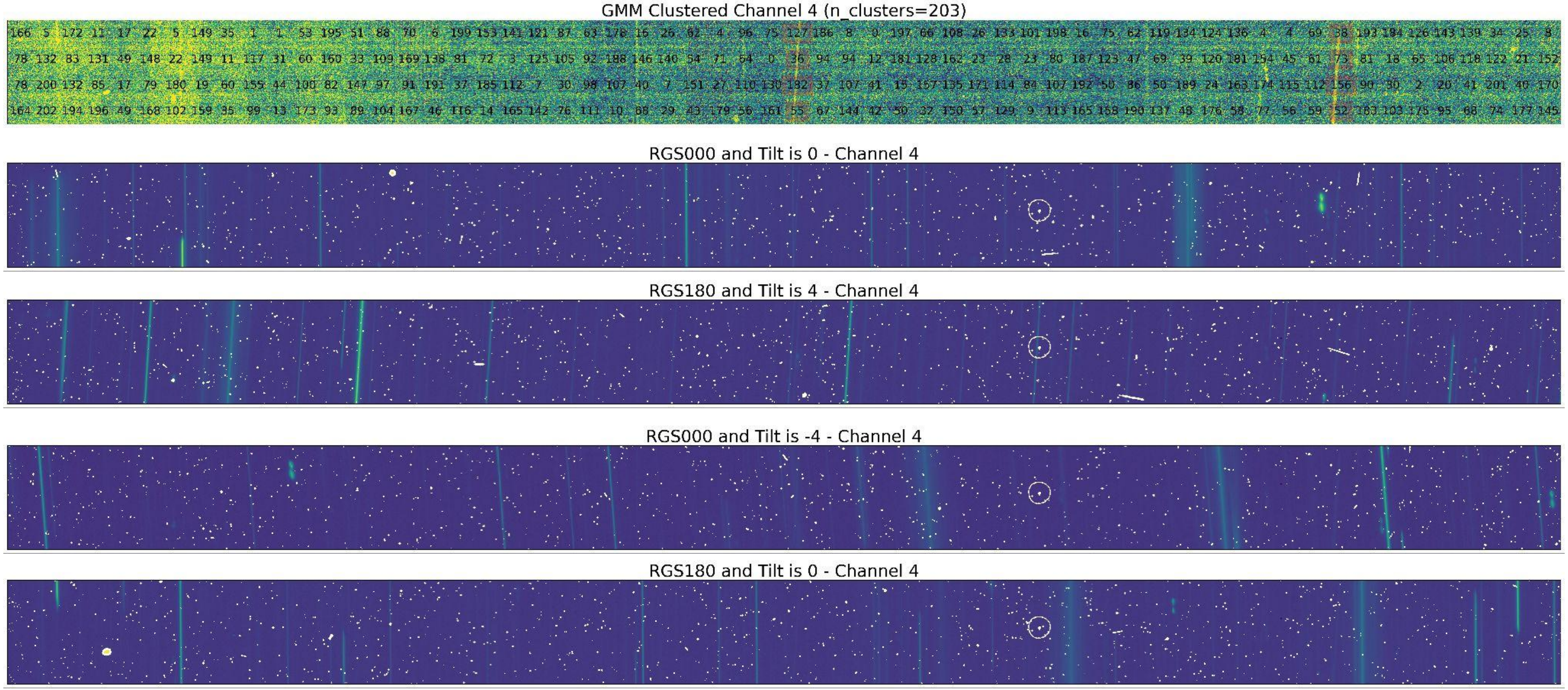
RGS180 and Tilt is 0 - Channel 3



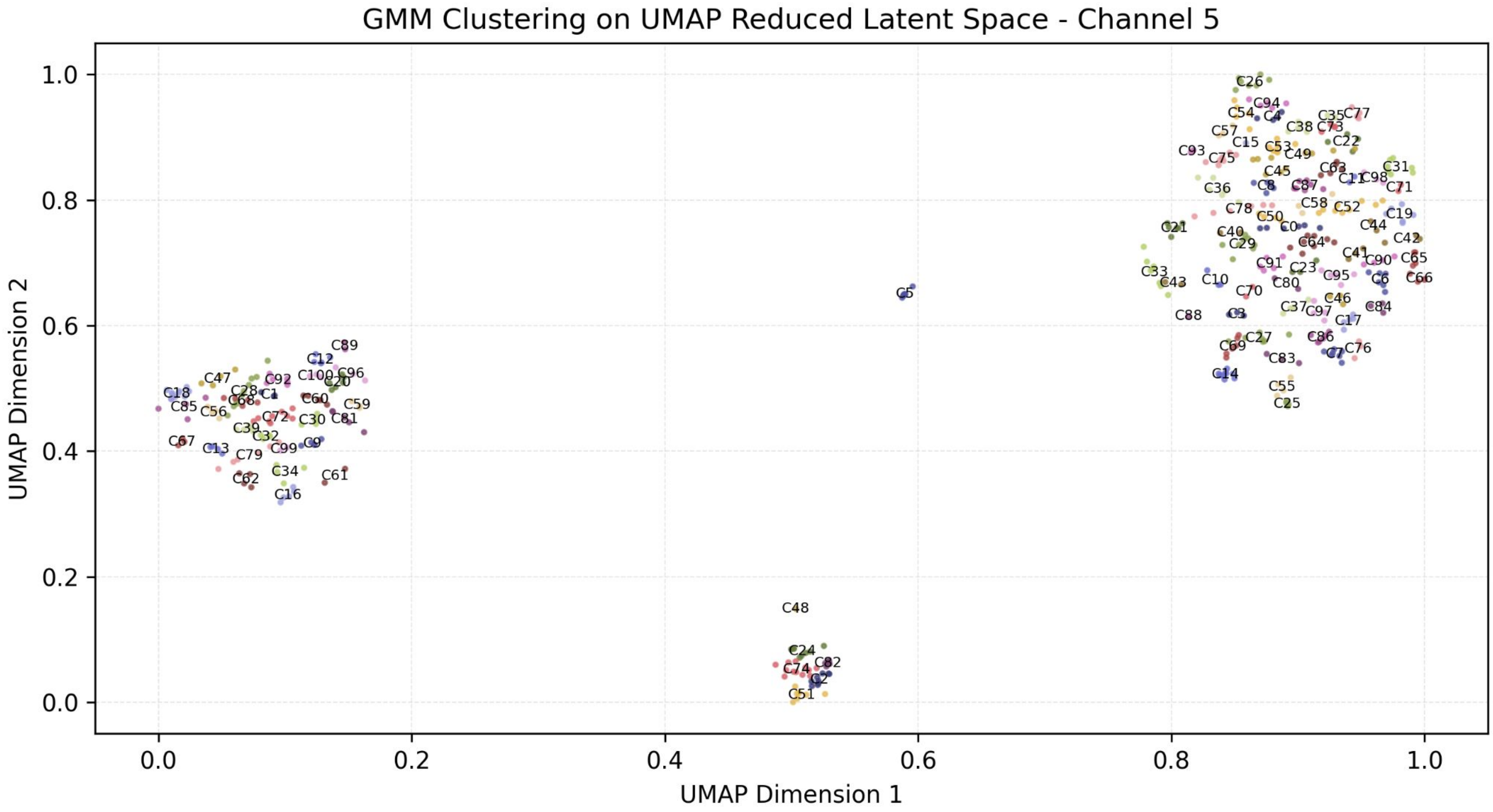
Channel 4 Latent Space Analysis



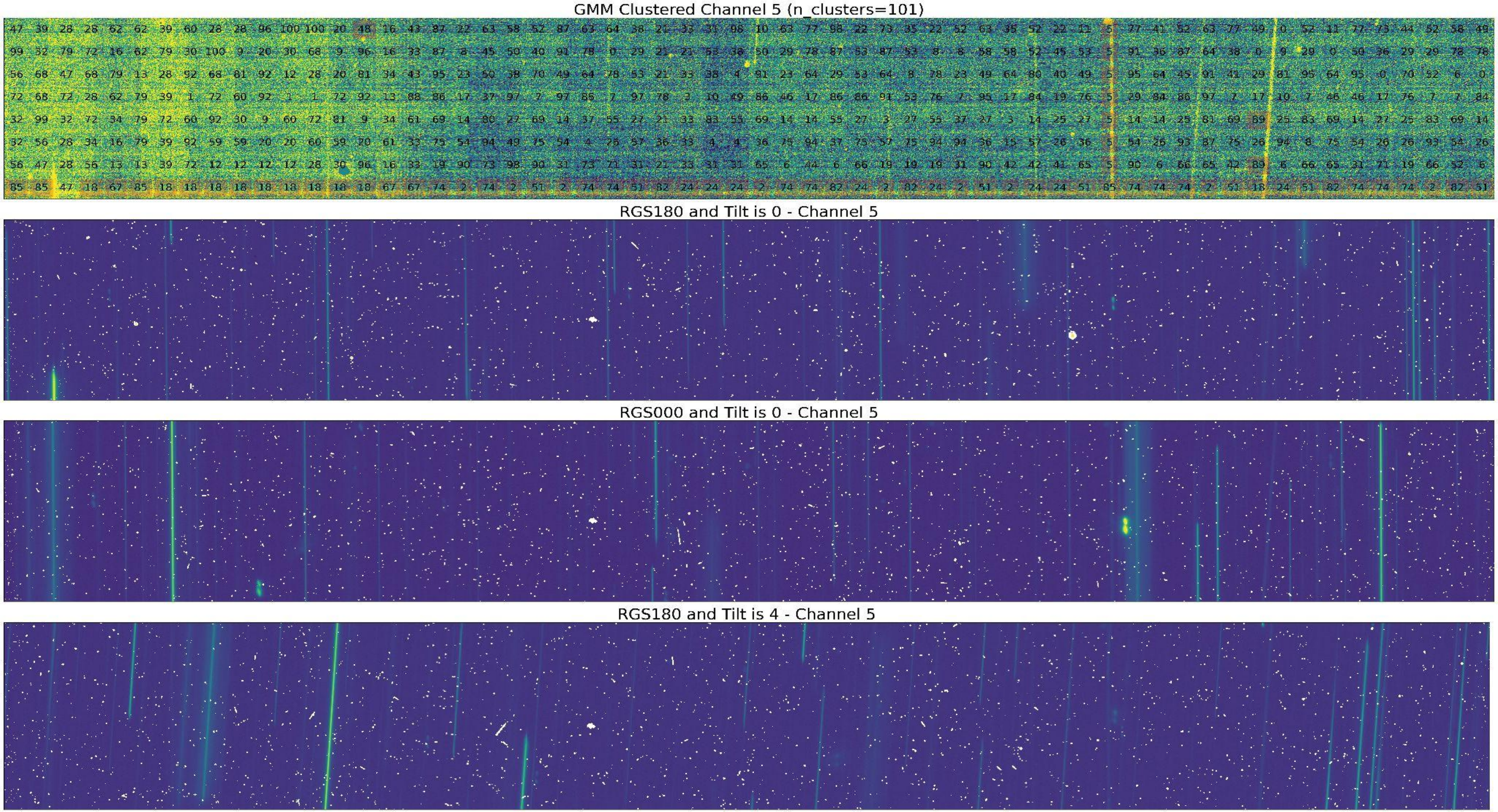
Channel 4 : Slew dark Persistence Unsupervise Recognition



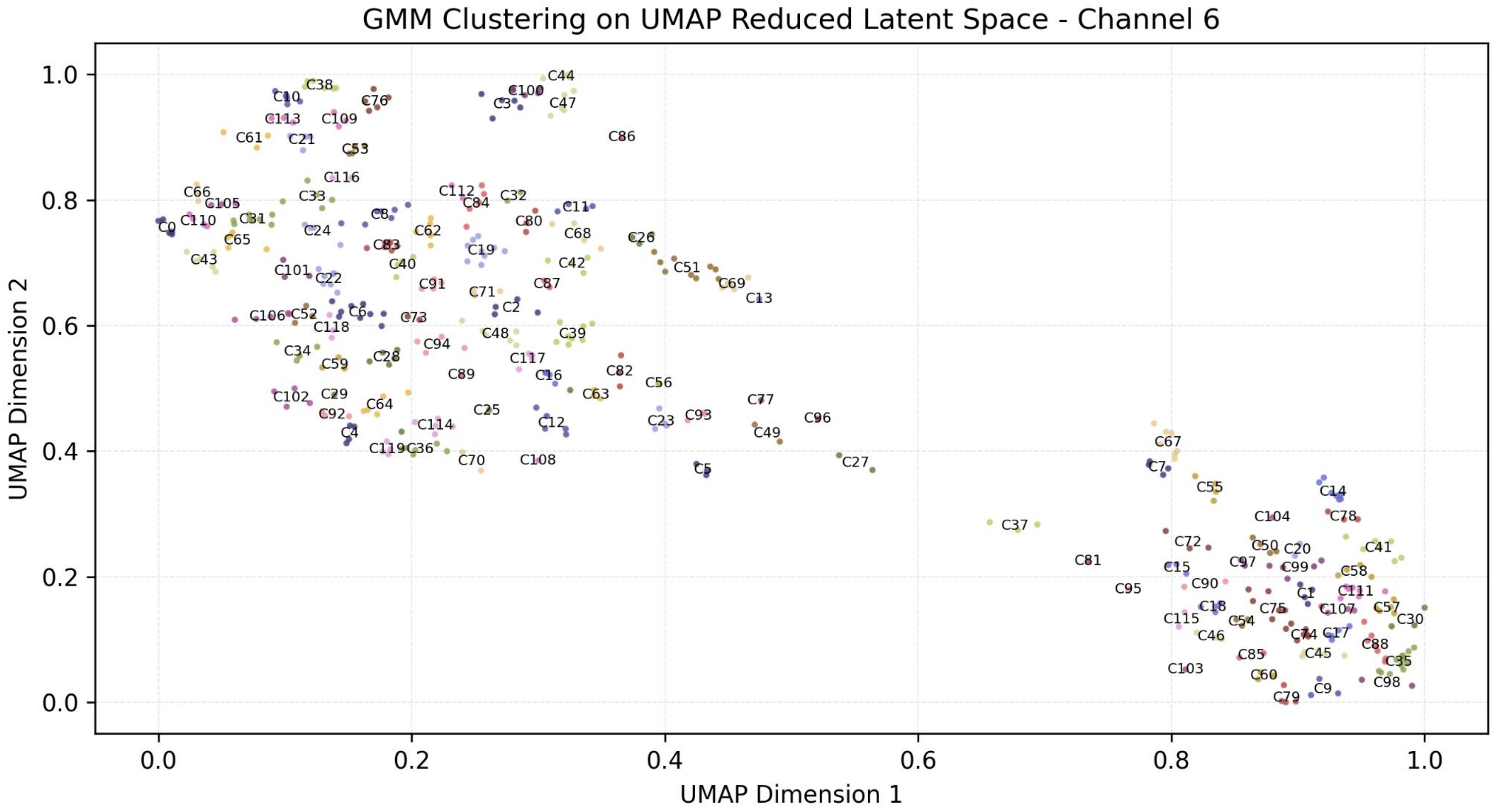
Channel 5 Latent Space Analysis



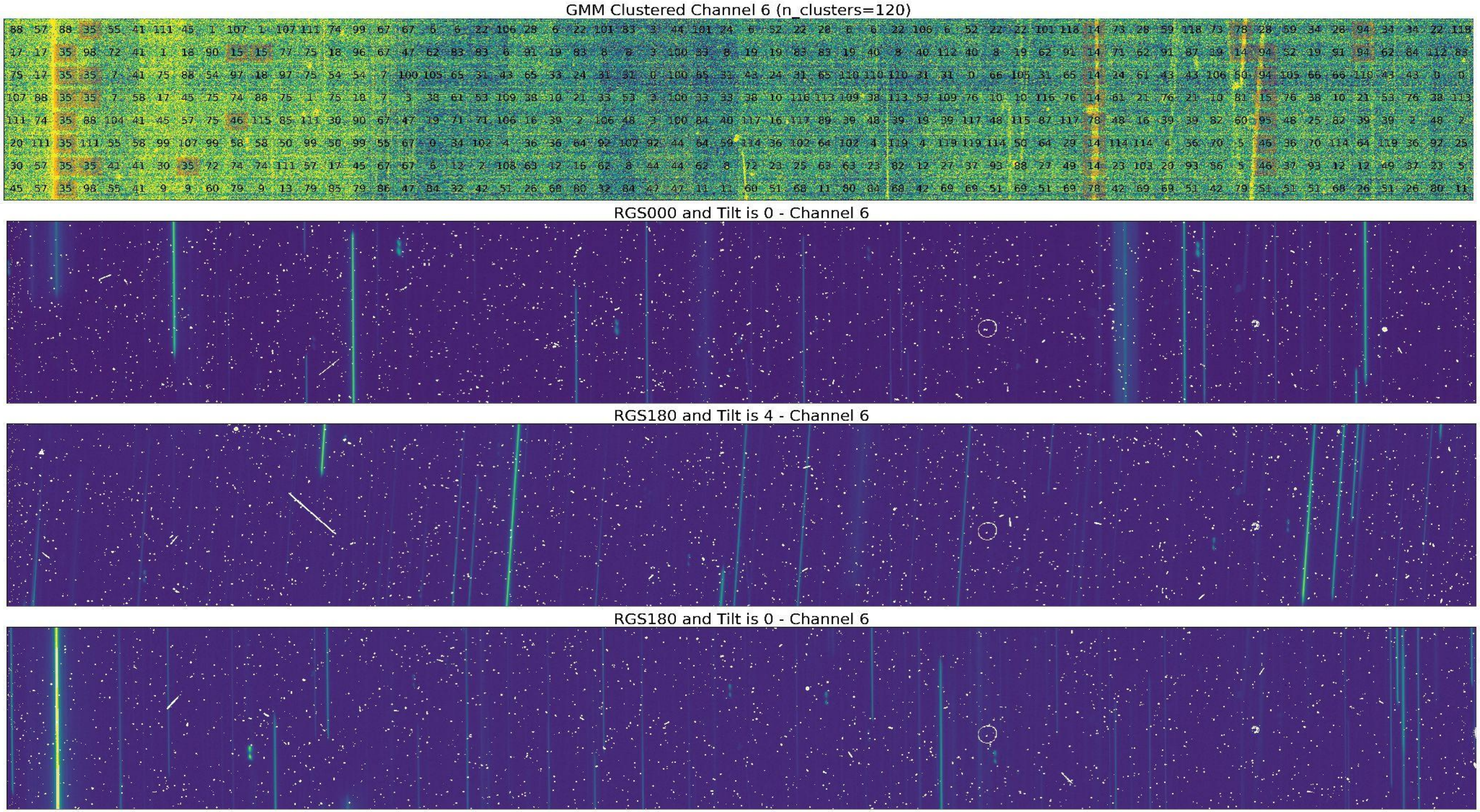
Channel 5 : Slew dark Persistence Unsupervise Recognition



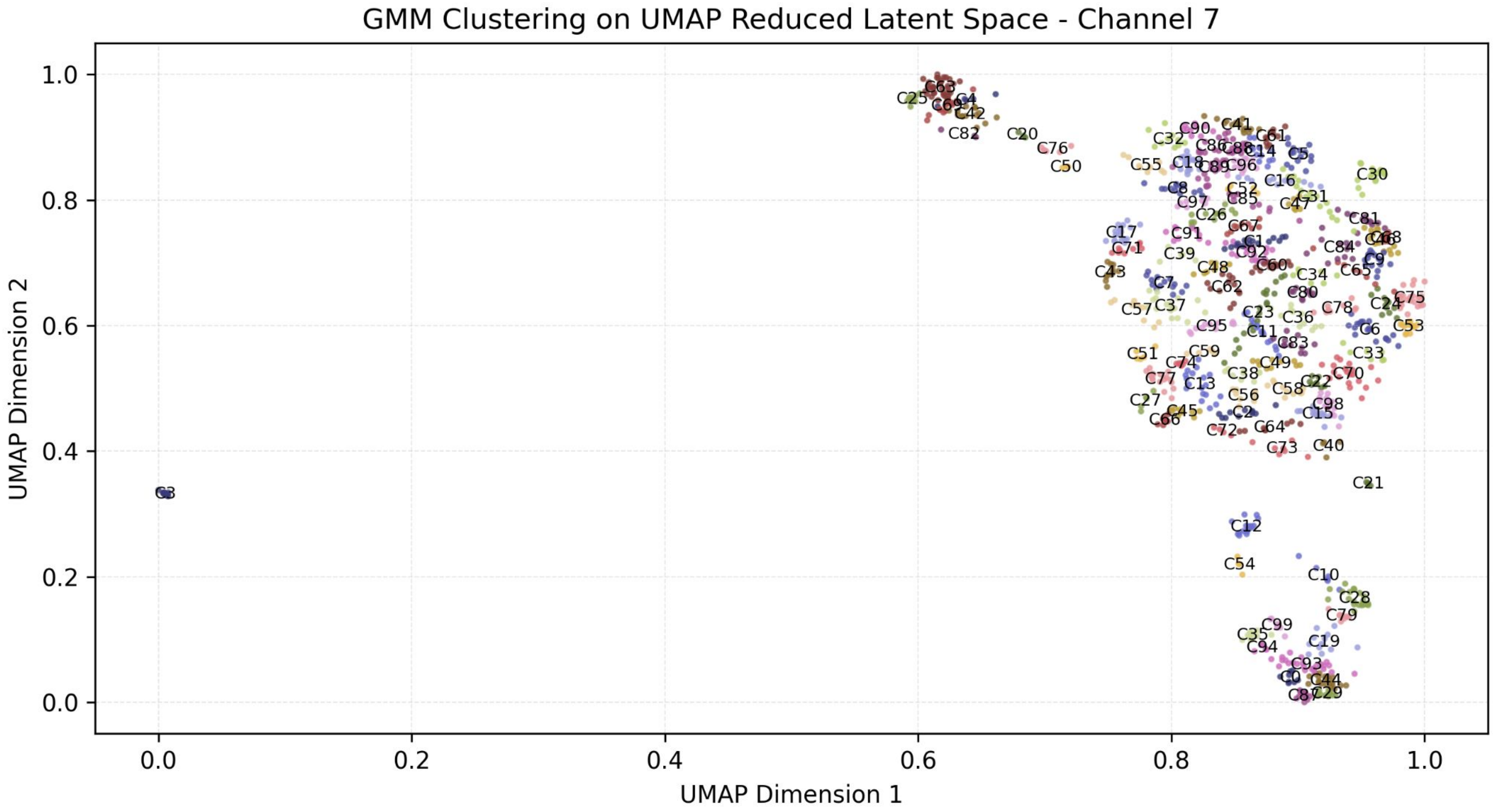
Channel 6 Latent Space Analysis



Channel 6 : Slew dark Persistence Unsupervise Recognition

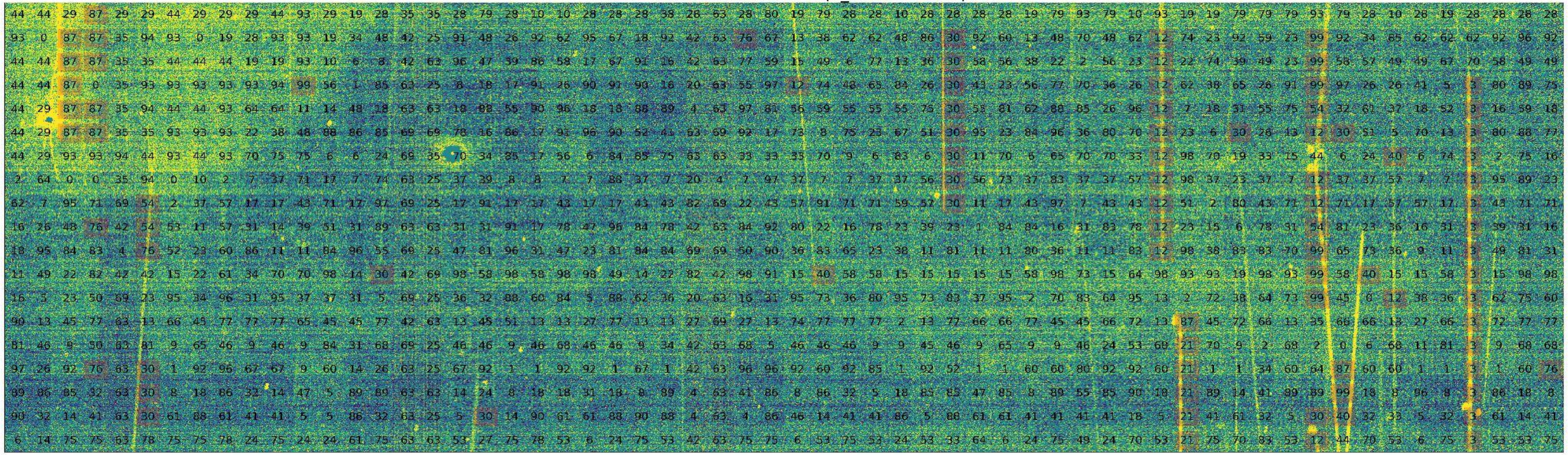


Channel 7 Latent Space Analysis

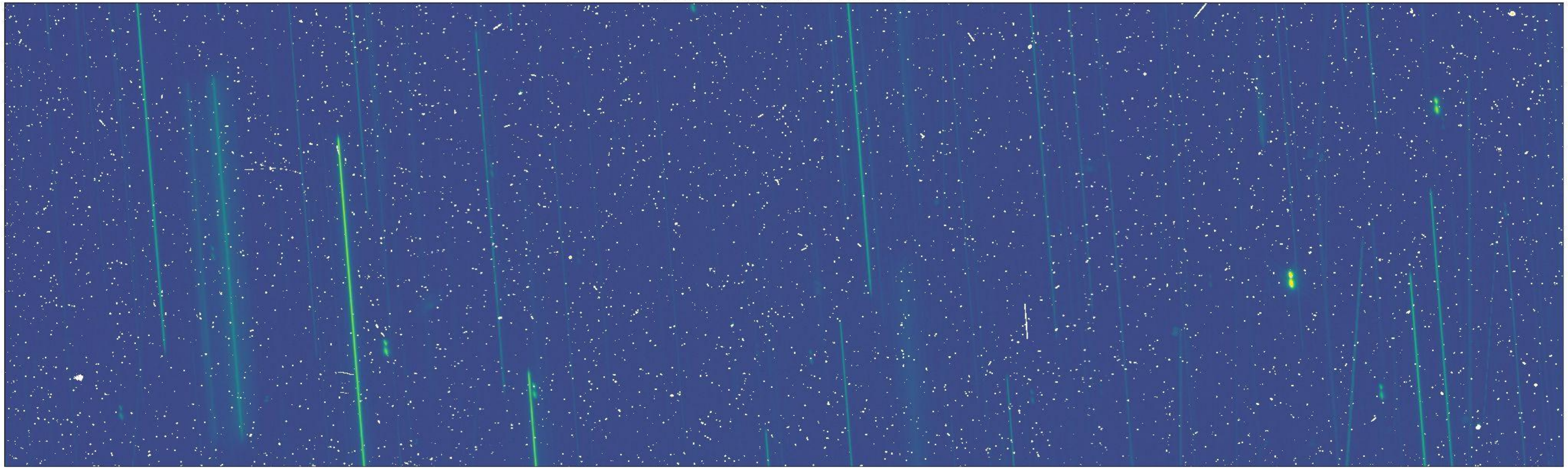


Channel 7: Slew dark Persistence Unsupervise Recognition

GMM Clustered Channel 7 (n_clusters=100)

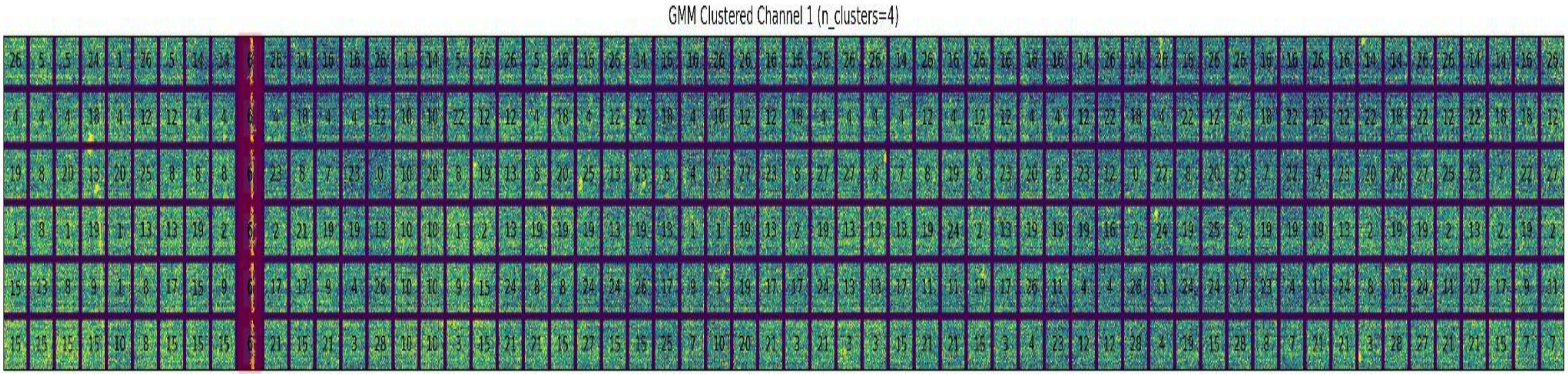


RGS000 and Tilt is -4 - Channel 7

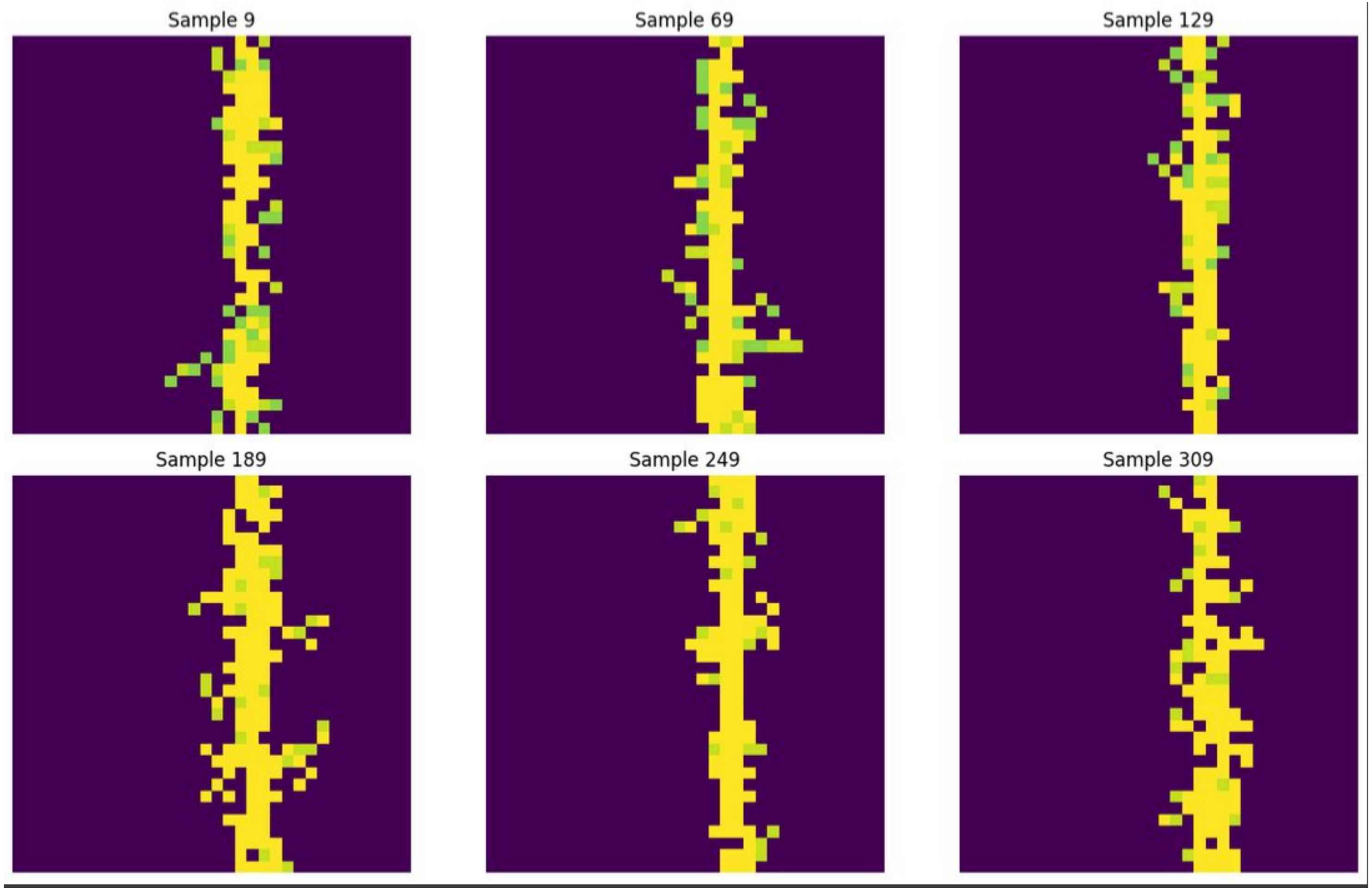


Segmentation Pipeline

Step	What It Does	Effect
1️⃣ Thresholding (Mean)	Keeps pixels above a brightness threshold	Removes background
2️⃣ Standard Deviation Filtering	Keeps only high-contrast regions	Removes weak features
3️⃣ Connected Component Analysis	Groups connected bright pixels	Identifies separate objects
4️⃣ Small Cluster Removal	Removes small noisy blobs	Keeps only significant clusters



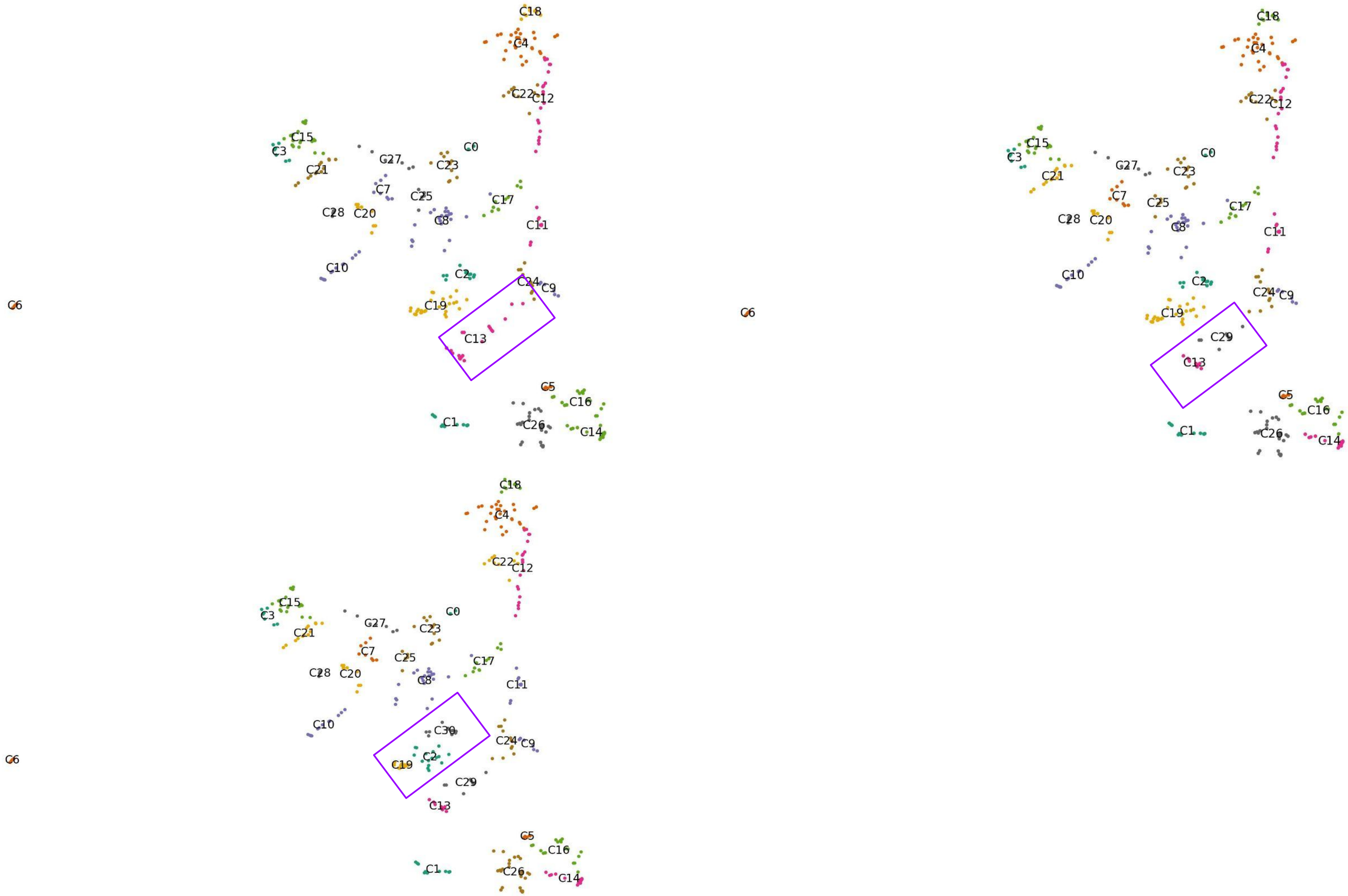
Segmentation Examples



Next Steps …

1) Merging latent space of 7 channels
2) Labeling clusters including persistances and define a neighboring area for each cluster
3) Applying this approach on rest of exposures and consider any sample as persistence if they are in the neighboring area of persistence clusters
4) Applying segmentation on persistence samples and labelling them for supervise training
5) Training A UNet using the training data from segmented persistence samples
6) Apply the trained Unet on whole samples to improve the segmentation
7) …

Extra Slides


C18
C4
C22
C12
C15
C3
C21
C27
C23
C0
C7
C25
C17
C28
C20
C8
C11
C10
C2
C24
C9
C6
C19
C13
C5
C16
C1
C26
C14
C29
C30

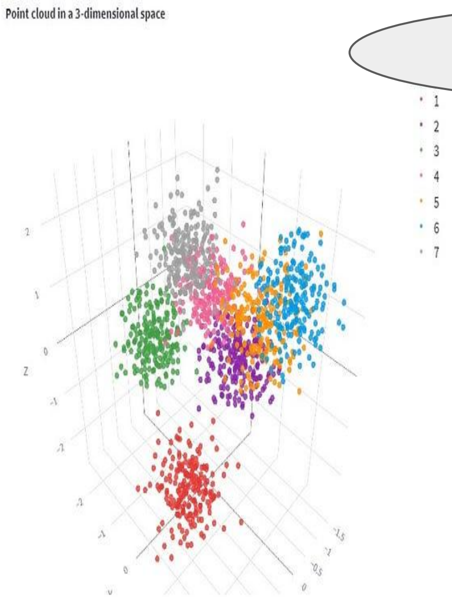
Latent Space Analysis - UMAP as A Dimensionality Reduction Tool

Latent Space Shape
(34,34,64)

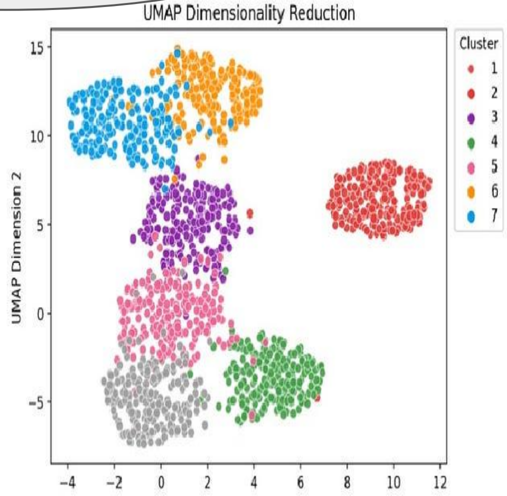
High Dimensional Space

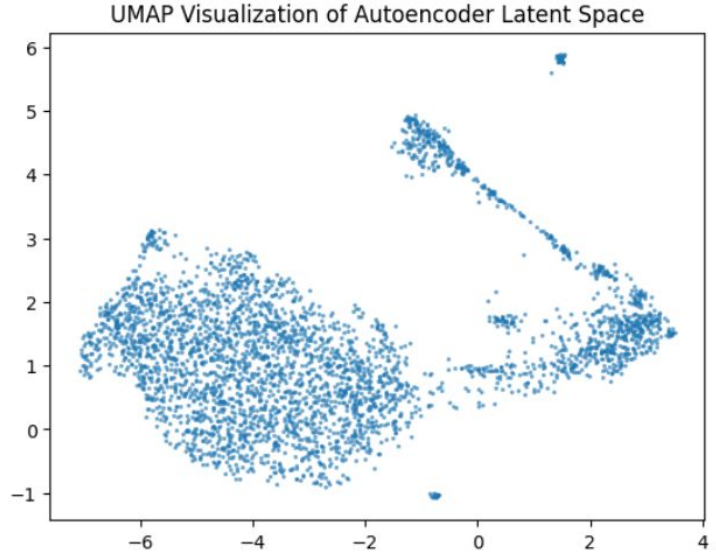
Uniform Manifold Approximation and Projection

2 Dimensional Space



Example for UMAP 3D to 2D reduction

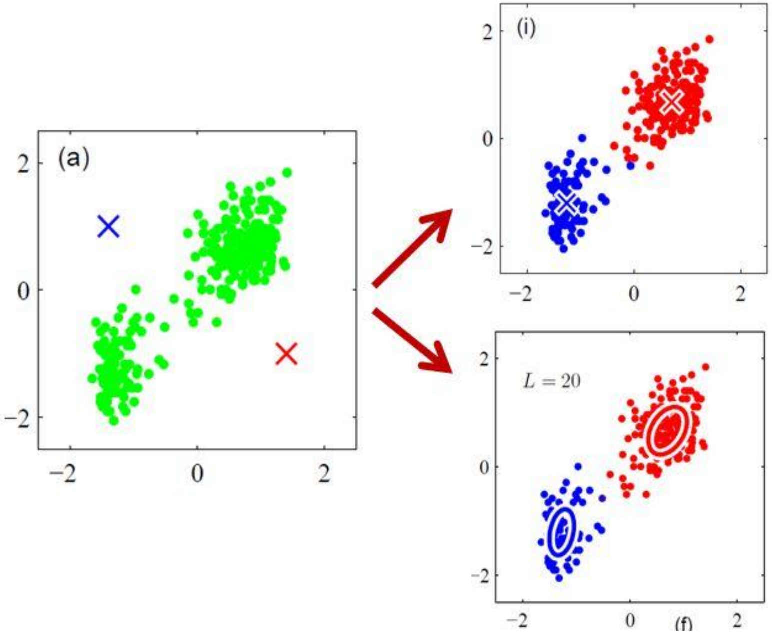




Latent Space Analysis - K-means for Clustering UMAP Space

K-means vs GMM

Two standard methods are k-means and Gaussian Mixture Model (GMM).
K-means assigns data points to the nearest clusters, while GMM represents data by multiple Gaussian densities.



Hard clustering: a data point is assigned a cluster.

Soft clustering: a data point is explained by a mix of multiple Gaussians probabilistically.

K-means
# of Clusters = 25