Assessing the performance and quality of dimensionality reduction techniques with t-SNE (Rapids-AI), UMAP (Rapids-AI), PaCMAP and TriMap for a large-scale dataset.

Patrycja Lewczuk, Sara Świętek

19.7.2023

Contents

1 Introduction **3**

2 Dimensionality reduction techniques **4**
2.1 Rapids AI 4
2.1.1 t-Distributed Stochastic Neighbor Embedding (t-SNE) 4
2.1.2 Uniform Manifold Approximation and Projection (UMAP) 4
2.2 Probabilistic Agglomerative Network Map (PaCMAP) 5
2.3 Large-scale Dimensionality Reduction Using Triplets (TriMap) 5

3 Dataset description **7**

4 Feature Selection **8**
4.1 PCA 8
4.2 SelectKBest 9
4.3 Correlations 9

5 Visualizations **10**
5.1 t-SNE (Rapids-AI) 10
5.2 UMAP (Rapids-AI) 10
5.3 PaCMAP 17
5.4 TriMap 17

6 Cluster analysis **24**

7 Performance Calculations **28**

8 Summary **31**

1 Introduction

The aim of our project is to perform dimensionality reduction using four different techniques: t-SNE (Rapids-AI), UMAP (Rapids-AI), PaCMAP, and TriMap on a large-scale dataset. For this purpose, we used a dataset describing particle collision measurements in the Large Hadron Collider (LHC) in CERN, which was obtained from the Kaggle platform [1]. This dataset consists of 50 features and 1.2 million rows.

In this report, we will present three different feature selection methods aimed at reducing the number of features to approximately 30-35. For this purpose, we employed Principal Component Analysis (PCA), the SelectKBest tool from Scikit-Learn, and correlation analysis between features.

We also performed visualizations for the subset of 100,000 rows for each technique. For this, we varied feature selection techniques and hyperparameters to obtain the best results of clusterisation.

For obtained clusters, we performed detailed analysis to gain a better understanding of the factors driving their formation and composition.

For chosen features and hyperparameters, we performed a visualisation of 1 million datapoints for each technique. To compare the performances of each method, we recorded runtimes and memory usages.

2 Dimensionality reduction techniques

For this work we used 4 techniques for visualizing high-dimensional data in a lower-dimensional space: t-SNE, UMAP, PaCMAP, TriMap. In our case, we concentrate on two-dimensional visualisations. Chosen techniques are useful for exploring and visualizing complex datasets where the relationships between data points are non-linear and difficult to capture with traditional linear techniques (such as PCA).

2.1 Rapids AI

Rapids-AI [3] library is an open-source data science and machine learning library that aims to accelerate the data processing and model training workflows.

For our work, we will use the Rapids-AI implementation of t-SNE and UMAP. These versions leverage GPU acceleration for faster computation, making it suitable for large-scale datasets. With the power of GPUs, Rapids-AI can significantly speed up the t-SNE and UMAP computation compared to traditional CPU-based implementations.

2.1.1 t-Distributed Stochastic Neighbor Embedding (t-SNE)

The t-SNE algorithm works by modeling the similarity between data points in the high-dimensional space and mapping them to a lower-dimensional space. It constructs a probability distribution that represents the similarities between pairs of data points in the original space and a similar probability distribution in the lower-dimensional space. It aims to minimize the divergence between these two distributions.

The parameters we focus on when using the t-sne technique are:

- metric - determines the distance metric used to measure the similarity between data points in the high-dimensional space,
- perplexity - controls the balance between preserving local and global structures in the data and roughly represents the number of close neighbors that each point considers during the optimization process,
- learning_rate - determines the step size at each iteration during the optimization process. A higher learning rate may result in faster convergence, but can lead to poor local optima. Conversely, a lower learning rate may provide better embeddings but requires more iterations to converge,
- n_iter - specifies the number of iterations or optimization steps performed by t-SNE. Increasing the number of iterations allows for further refinement of the embeddings, but can also increase the computational time.

2.1.2 Uniform Manifold Approximation and Projection (UMAP)

The Uniform Manifold Approximation and Projection (UMAP) algorithm constructs a high-dimensional graph representation of the data, capturing the local and global structure. It then optimizes the embedding in the lower-dimensional space by seeking a balance between preserving the local neighborhood relationships and maintaining the global structure of the data.

The parameters we focus on when using the UMAP technique are:

- metric - determines the distance metric used to measure the similarity or dissimilarity between data points in the high-dimensional space,
- n_neighbors - specifies the number of neighboring data points considered for each data point during the construction of the high-dimensional graph. These neighbors play a crucial role in capturing the local structure of the data.
- min_dist - controls the minimum distance in the low-dimensional space between embedded points. It determines the extent to which UMAP spreads out the clusters or points in the embedding.

2.2 Probabilistic Agglomerative Network Map (PaCMAP)

Unlike some other dimensionality reduction methods (t-SNE or UMAP), which focus on preserving local structures and may lose global structures, PACMAP [2] aims to retain both aspects of the data. It achieves this through a probabilistic agglomerative clustering approach.

The PACMAP algorithm constructs a hierarchical clustering of the data using a density-based clustering technique. It starts with individual data points and gradually merges them into clusters based on their density and proximity. This hierarchical clustering structure captures both local neighborhoods and global connections within the data. Once the clustering is established, PACMAP performs dimensionality reduction by representing each cluster as a single point in the low-dimensional space. The position of each cluster point is determined based on its connectivity to other clusters, preserving the global relationships. The resulting PACMAP visualization provides a low-dimensional representation of the data that reflects both the local and global structures.

The parameters we focus on when using the PaCMAP technique are:

- n_neighbors - specifies the number of nearest neighbors considered for each data point to define neighbor pairs. Neighbor pairs capture the local structure of the data and are used to preserve local relationships in the low-dimensional embedding.
- MN_ratio - determines the ratio of mid-near pairs (pair_MN) to neighbor pairs (pair_neighbors). Mid-near pairs capture relationships between points that are neither too close nor too far from each other, aiming to preserve both local and global information.
- FP_ratio - controls the ratio of further pairs (pair_FP) to neighbor pairs (pair_neighbors). Further pairs capture relationships between points that are farther apart and are used to preserve the global structure in the low-dimensional embedding.

2.3 Large-scale Dimensionality Reduction Using Triplets (TriMap)

The Trimap [4] is a method based on triplet constraints, which preserves the global structure of the data better than the other commonly used methods such as t-SNE, LargeVis, and UMAP. The triplet constraints are of the form "point i is closer to point j than point k". The triplets are sampled from the high-dimensional representation of the points, and a weighting scheme is used to reflect the importance of each triplet.

The parameters we focus on when using the TriMap technique are:

- n_inliers - represents the number of inlier points to sample from the high-dimensional space. Inliers are data points that are well-behaved and conform to the underlying structure of the data. TriMap uses the inlier points to form triplets that capture the relationships between points.

- n_outliers - represents the number of outlier points to sample from the high-dimensional space. Outliers are data points that deviate significantly from the underlying structure of the data. TriMap uses the outlier points to form triplets that represent dissimilar relationships.
- n_random - represents the number of randomly sampled points used to form triplets in TriMap. Randomly sampled triplets provide additional information about the data relationships and can help with the overall structure of the embedding.
- distance - determines the type of distance or similarity measure used to define the pairwise relationships between points in the high-dimensional space.

3 Dataset description

The dataset used in this study contains measurements recorded at the LHC during particle collisions. It contains 50 features, which describe particle energies and momenta, analysis of particle trajectory, flags (0 or 1) indicating whether a particle passes through a specific detector and other relevant attributes that provide crucial information about the physics processes involved. One feature contains labels, denoting particle types. It can take values "Electron", "Muon", "Kaon", "Proton", "Pion" and "Ghost". There is a balanced distribution of data points for each particle, ensuring that there is a roughly equal representation for every particle.

The particles in this dataset differ in their fundamental properties and behavior:

- Electron: Electrons are elementary particles with a negative charge and a relatively low mass. In the LHC, electrons are typically detected through their interaction with electromagnetic fields, such as by measuring their energy deposition in calorimeters.
- Muon: Muons are similar to electrons, but are heavier and less common. They can penetrate matter more easily due to their higher mass. Detection of Muons in the LHC often involves specialized muon detectors that measure their trajectory and energy loss as they pass through various layers.
- Kaon: Kaons are mesons composed of a quark and an antiquark. They can have either positive or negative charges. Detecting kaons in the LHC typically involves analyzing their decay products, as they have relatively short lifetimes and quickly decay into other particles.
- Proton: Protons are composite particles made up of three quarks. They have a positive charge and are stable particles. Protons are the primary particles accelerated and collided in the LHC. Their detection involves measuring the energy deposition and trajectory of the particles resulting from proton-proton collisions.
- Pion: Pions are mesons composed of a quark and an antiquark. They can be positively charged, negatively charged, or neutral. Pions have relatively short lifetimes and quickly decay into other particles. Their detection in the LHC often involves identifying their decay products and measuring their energy deposition.
- Ghost: The term "Ghost" is not a specific particle type but is likely used as a label for unidentified or ambiguous particles in the dataset.

The LHC dataset with detailed description of all labels can be found on Kaggle [1]. In this work, we will use only the training file containing labels, since we concentrate on dimensionality reduction rather than constructing a prediction model. The training file contains relatively large amount of rows: 1.2 million, which is suitable for our experiment.

4 Feature Selection

On the original and normalized dataset we applied 3 feature selection techniques: PCA (30D and 20D), SelectKBest method from scikit learn library and correlation examination. In further chapters, we will compare visualizations of all 3 techniques.

4.1 PCA

PCA (Principal Component Analysis) is a dimensionality reduction technique used to simplify and analyze complex datasets. It works by identifying the directions (principal components) along which the data varies the most. These principal components are linear combinations of the original variables. The first principal component captures the maximum variance in the data, the second principal component captures the second maximum variance orthogonal to the first component, etc. Each principal component is a linear combination of the original variables, and they are uncorrelated with each other.

The principal components are computed such that the first component explains the largest possible amount of variance in the data, and each subsequent component explains the remaining variance in decreasing order. By choosing a subset of the principal components, it is possible to reduce the dimensionality of the dataset while retaining most of the information.

We are using PCA to decrease the dimensionality to 20 or 30 features. However, after visualize the dataset for a million records of only 20 features (figure 1), we can notice significant loss of information and no clusters built. For this reason, we will use only PCA with 30 features for further analysis.

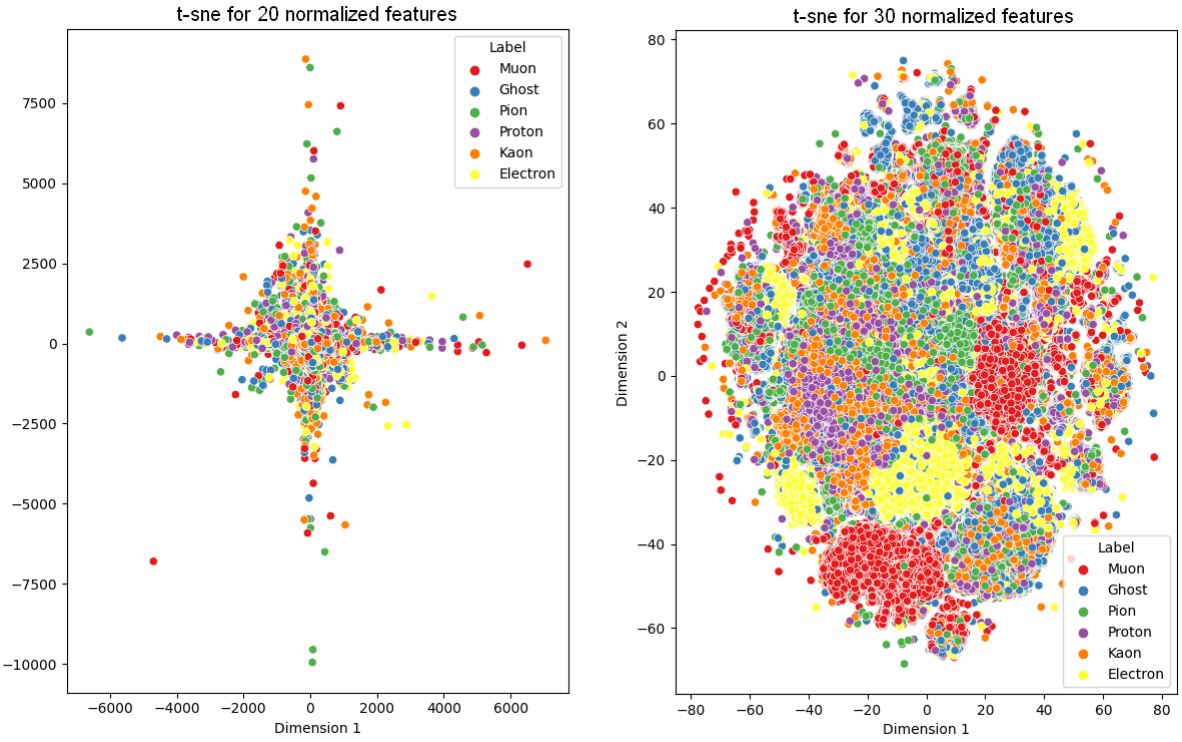

Figure 1: Visualization of t-sne for 20 and 30 features using PCA

4.2 SelectKBest

SelectKBest is a feature selection technique from scikit learn library used to select the K most informative features from a given dataset. It provides a simple and efficient approach to feature selection.

The SelectKBest algorithm ranks the features based on their individual relationship with the target variable, using statistical tests such as chi-squared test, ANOVA F-value, mutual information, or other appropriate measures. It evaluates each feature independently and assigns a score or p-value to indicate its relevance to the target variable. The algorithm selects the top K features with the highest scores or lowest p-values. By reducing the number of features,

For our feature selection with SelectKBest we used chi-squared test, and we were looking for k=35 best features.

4.3 Correlations

After analyzing correlations in this dataset, we discovered that a lot of features are highly correlated with each other.

The provided figure, labeled as Figure 2, illustrates the outcome achieved through an iterative examination of the correlation matrix. During this process, 26 pairs showing a correlation coefficient greater than 0.95 were identified. For the dimensionality reduction of the dataset, we deleted one feature of each highly correlated pair, getting as a result a dataset containing only 35 features.

```
SpdE - PrsDLLbeElectron: 0.9756308334052267
SpdE - PrsE: 0.9723478673936222
EcalDLLbeElectron - EcalDLLbeMuon: 0.9999672006405466
EcalDLLbeElectron - Calo2dFitQuality: 0.9900582976531298
DLLmuon - DLLelectron: 0.9974454646466033
DLLmuon - DLLkaon: 0.9694642660760205
DLLmuon - DLLproton: 0.9691475883360345
EcalDLLbeMuon - Calo2dFitQuality: 0.9903748918854208
DLLelectron - DLLkaon: 0.9730957787147295
DLLelectron - DLLproton: 0.9729644579704981
DLLkaon - DLLproton: 0.9931426482206003
PrsDLLbeElectron - PrsE: 0.9967984465662691
MuonLLbeBCK - MuonLLbeMuon: 0.9999983016968894
FlagHcal - HcalDLLbeElectron: 0.9544919641659796
FlagHcal - HcalDLLbeMuon: 0.9544377029907446
HcalDLLbeElectron - HcalDLLbeMuon: 0.9999911693629961
RICH_DLLbeBCK - RICH_DLLbeKaon: 0.9984869132357482
RICH_DLLbeBCK - RICH_DLLbeElectron: 0.9916648745154857
RICH_DLLbeBCK - RICH_DLLbeMuon: 0.9934253528322043
RICH_DLLbeBCK - RICH_DLLbeProton: 0.9991831719177067
RICH_DLLbeKaon - RICH_DLLbeElectron: 0.9910224982167757
RICH_DLLbeKaon - RICH_DLLbeMuon: 0.9927481061904997
RICH_DLLbeKaon - RICH_DLLbeProton: 0.998355257178084
RICH_DLLbeElectron - RICH_DLLbeMuon: 0.9964002295439103
RICH_DLLbeElectron - RICH_DLLbeProton: 0.9910102529327913
RICH_DLLbeMuon - RICH_DLLbeProton: 0.9927706323764351
```

Figure 2: Pairs showing a correlation coefficient greater than 0.95

5 Visualizations

In order to utilize the feature selection methods mentioned in chapter 4, we conducted visualizations for each method to determine the best approach to identify defined clusters. Standardization was used for all feature selection techniques, only for PCA we kept the original values.

After selection of features, we showed visualizations for different values of selected parameters of the techniques. We omitted parameters where the difference of visualization for their different values was small or non-existent.

5.1 t-SNE (Rapids-AI)

The Figure 3 shows visualizations for various feature selection methods. At the very beginning, the unnormalized features for PCA can be discarded, as it can be seen that the rest of the methods better separate points into clusters.
In the end, we decided that the visualization for 30 normalized features by PCA came out best, where you can see how electrons and muons separate from the rest of the molecules.

For the various metrics in the figure 4, we did not notice major differences, but we can still see the grouping of electrons and muons.

In Figure 5, you can see how the perplexity parameter affects the visualization of the set. A higher value encourages the algorithm to focus on preserving local structures, while a lower value emphasizes preserving global structures.

Additionally, we check the visualizations for various values of learnign_rate and n_iter, but we couldn't notice any difference between visualizations.

5.2 UMAP (Rapids-AI)

The Figure 6 shows visualizations for various feature selection methods. Unlike t-sne, you can't see much difference between the methods. In addition, only muons gather in one place.
In the end, we decided that the visualization for normalized features selected by Select K-Best came out best, where most more muons are clustered in one place.

In the figure 7 we can see the influence of the number of neighbors. Increasing the value results in more accurate local structure preservation. However, if their number is too large, then there is no separation of points, much less their clustering.

The impact of min_dist is shown in the figure 8. A smaller value forces the algorithm to focus on preserving fine-grained local structures, potentially leading to tighter clusters. On the other hand, a larger min_dist value encourages UMAP to maintain broader global structures.

Additionally, we check the visualizations for various metrics, but we couldn't notice any difference between visualizations.

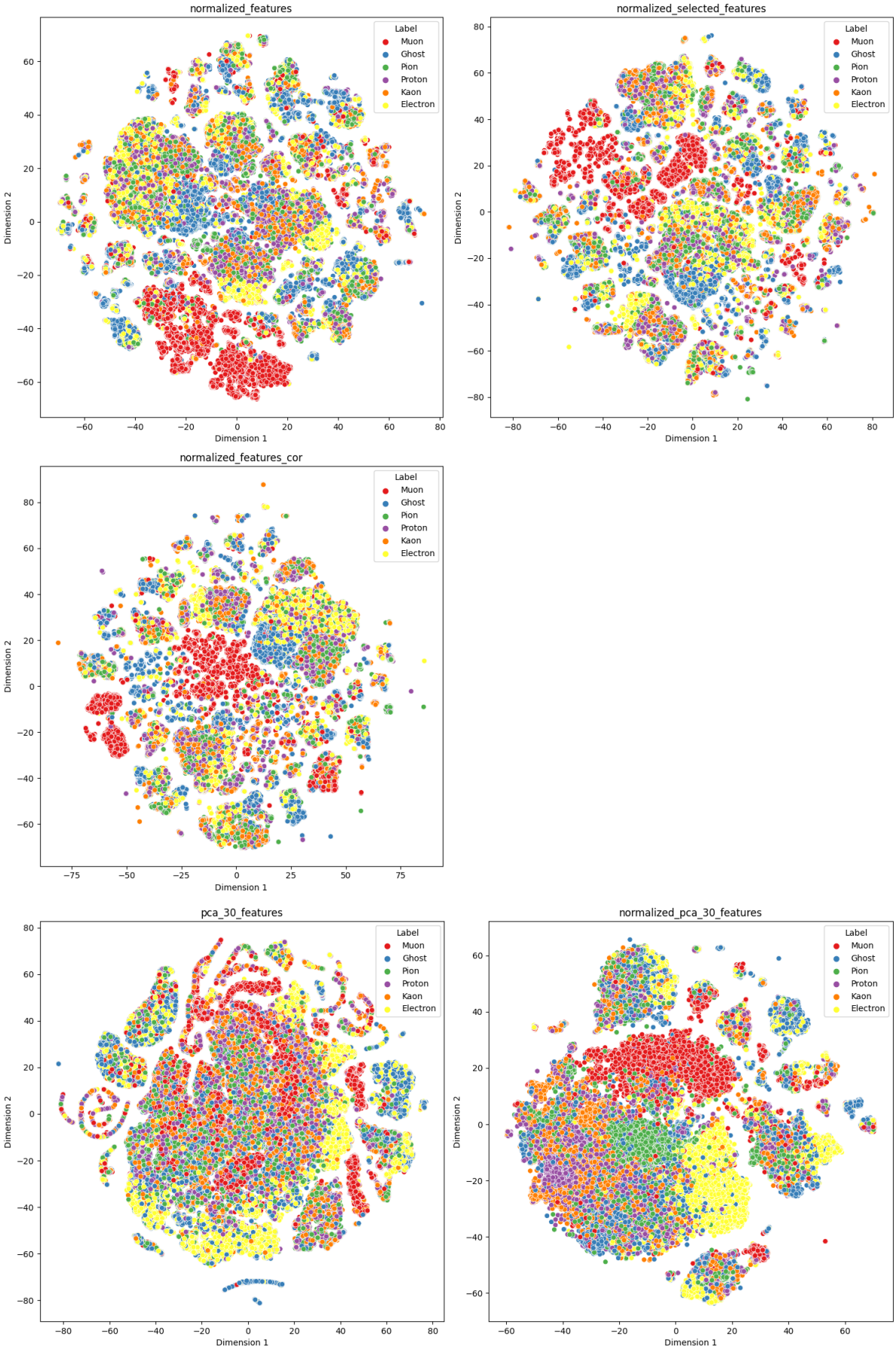

Figure 3: Visualization for various feature selection methods using t-SNE.

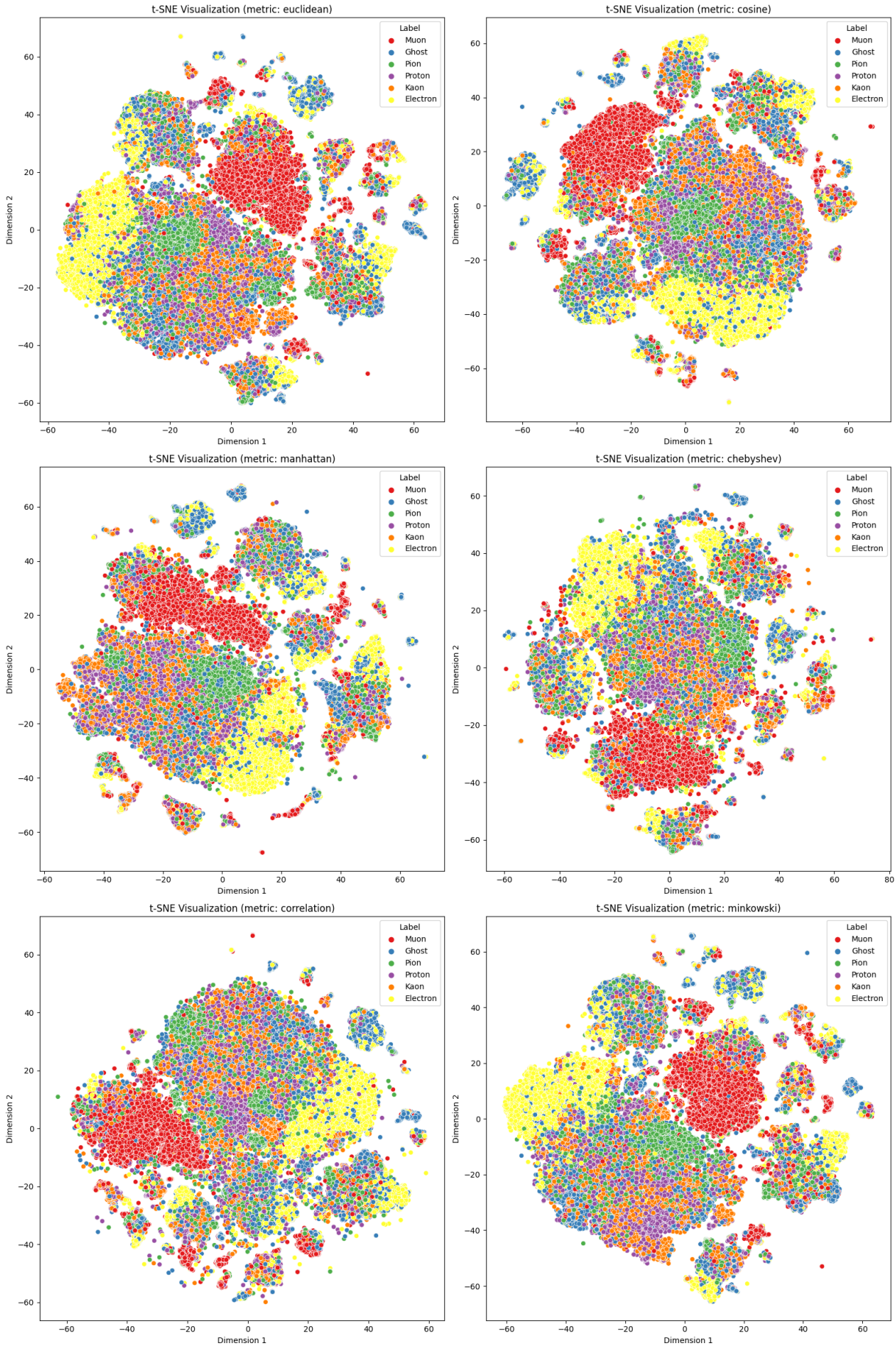

Figure 4: Visualization for different metrics using t-SNE.

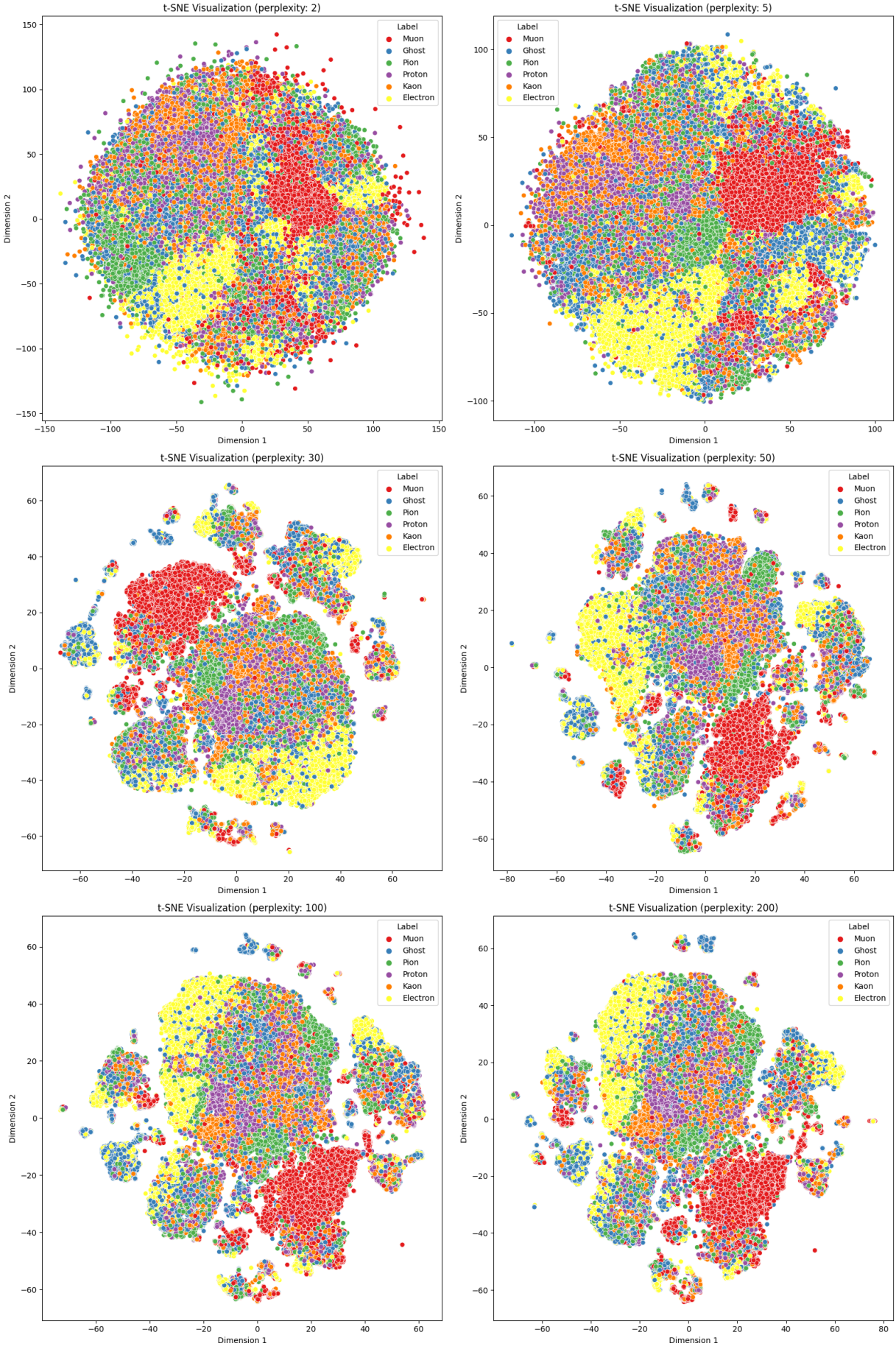

Figure 5: Visualization for different values of perplexity using t-SNE.

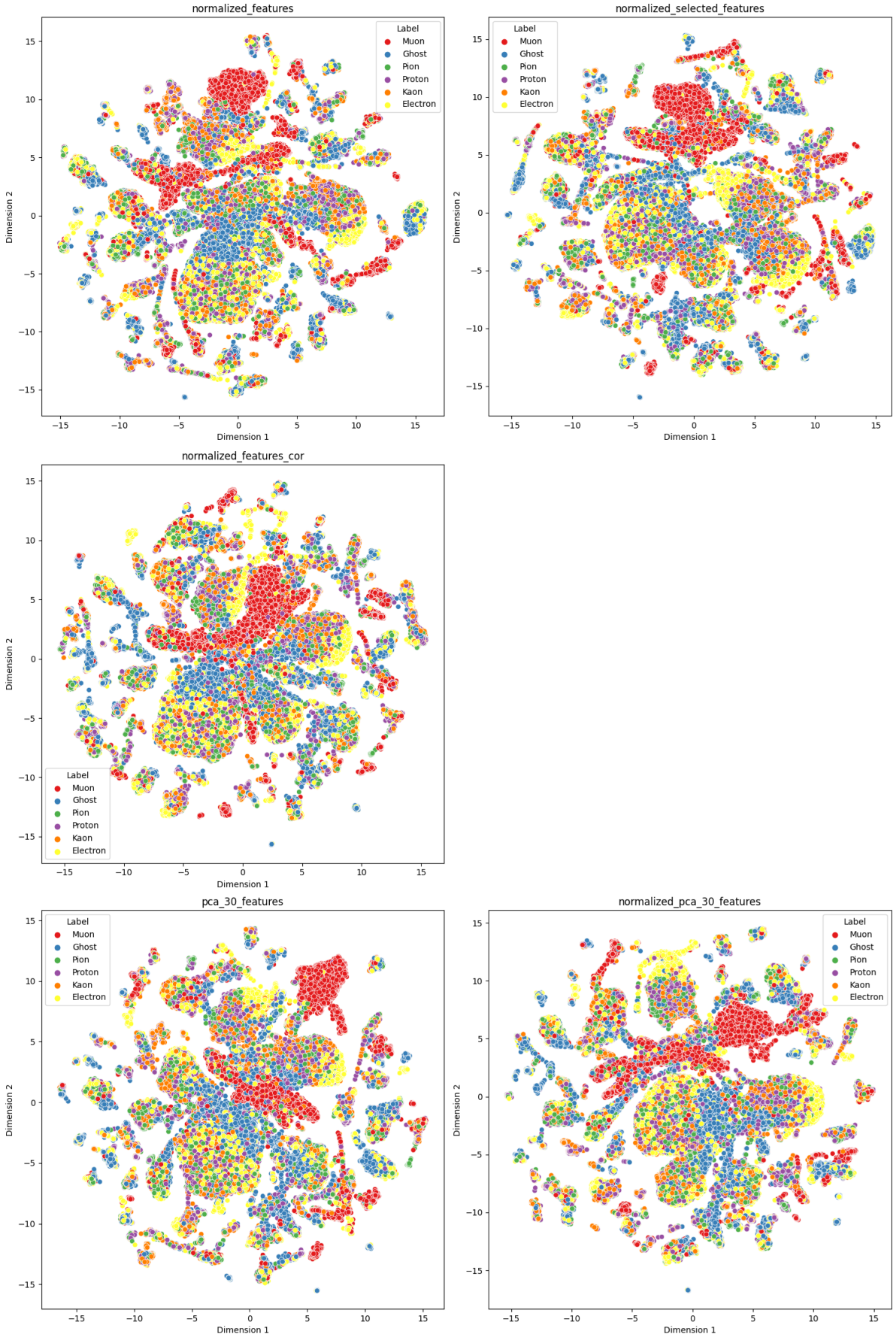

Figure 6: Visualization for various feature selection methods using UMAP.

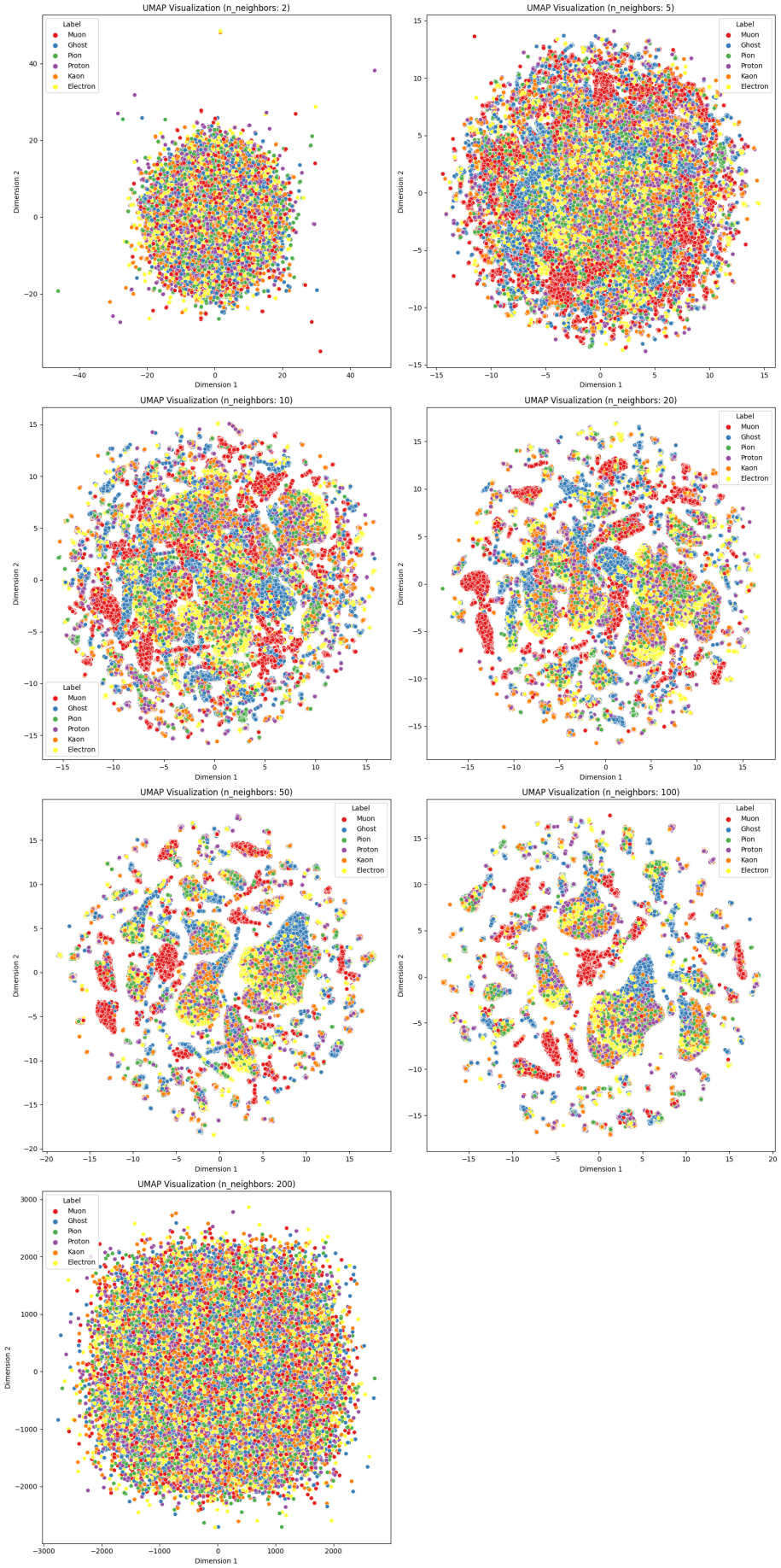

Figure 7: Visualization for different values of n_neighbors using UMAP.

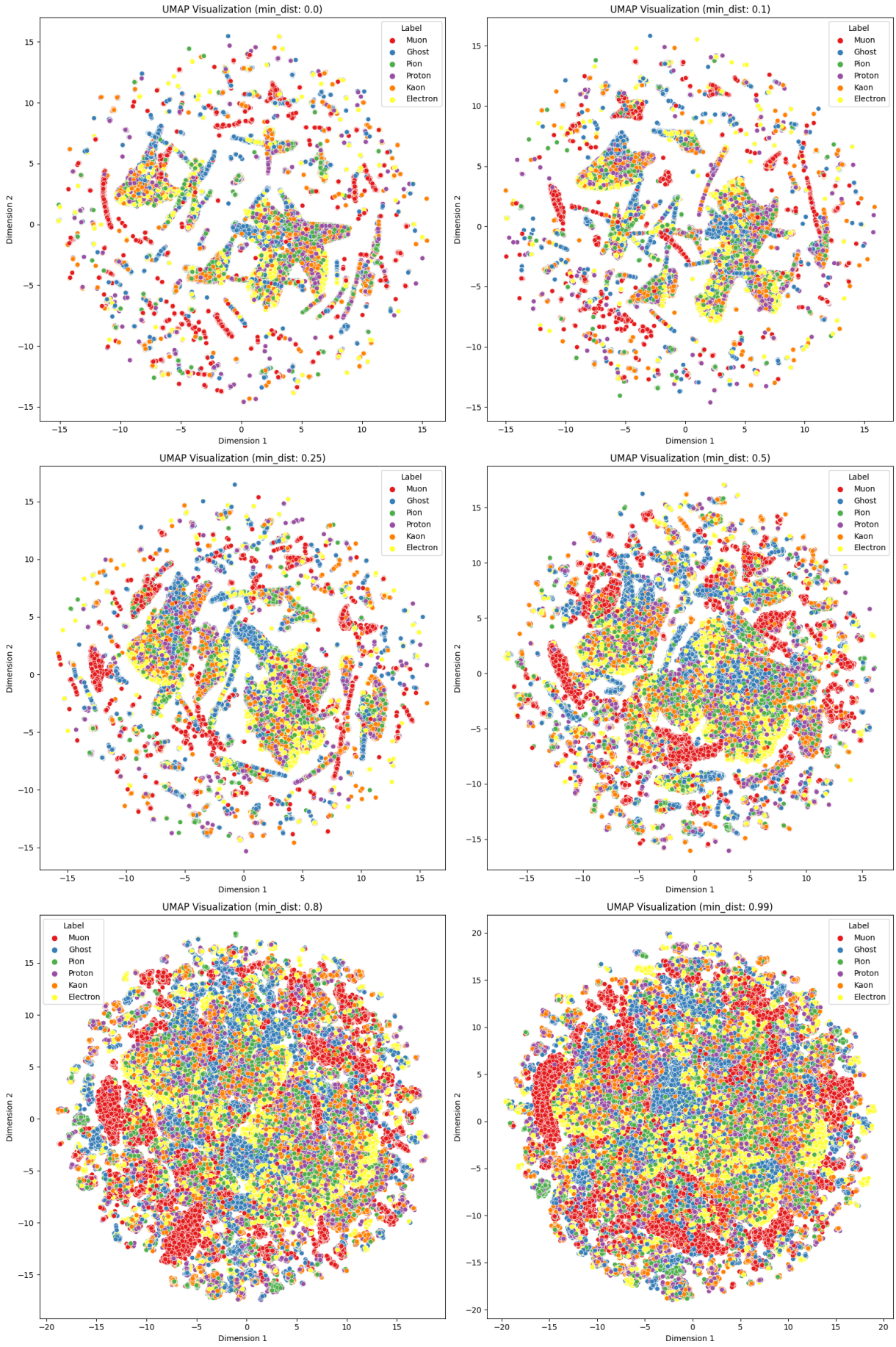

Figure 8: Visualization for different values of min_dist using UMAP.

5.3 PaCMAP

In PaCMAP we may change the initialization using the parameter 'init' in fit and fit_transformation function. The value of it may be 'pca' and 'random', which the last is the default value.
However, for various feature selection methods, we didn't notice any differences between visualizations, so we decided to choose 30 normalized features selected by PCA for parameters visualizations.

The figure 9 shows visualizations for different numbers of neighbors. Increasing the value allows for more local information to be captured. However, when the number is too large, no clusters are visible, unlike smaller values. Additionally, we can notice that not only electrons and muons are clustered in one place, but also pions slightly converge.

In the figure 10 we can notice that higher MN_ratio emphasizes global structure preservation, while a lower value focuses more on local structure.

Additionally, we check the visualizations for various values of fp_ratio, but we couldn't notice any difference between visualizations.

5.4 TriMap

Visualizations for various feature selection methods in the figure 11 are the most different of all the techniques we used. However, none is distinguished by better clustering. We decided to choose normalized features selected by correlation analysis.

For 3 of the 4 metrics shown in Figure 12, there is no clear difference between the visuals. Only the 'hamming' metric shows a significant deterioration in clustering.

In the figure 13 and 14 we can notice that increasing the value of n_inliers and n_random captures more local relationships and leads to a more comprehensive representation of the data.

Additionally, we check the visualizations for various values of n_outliers, but we couldn't notice any difference between visualizations.

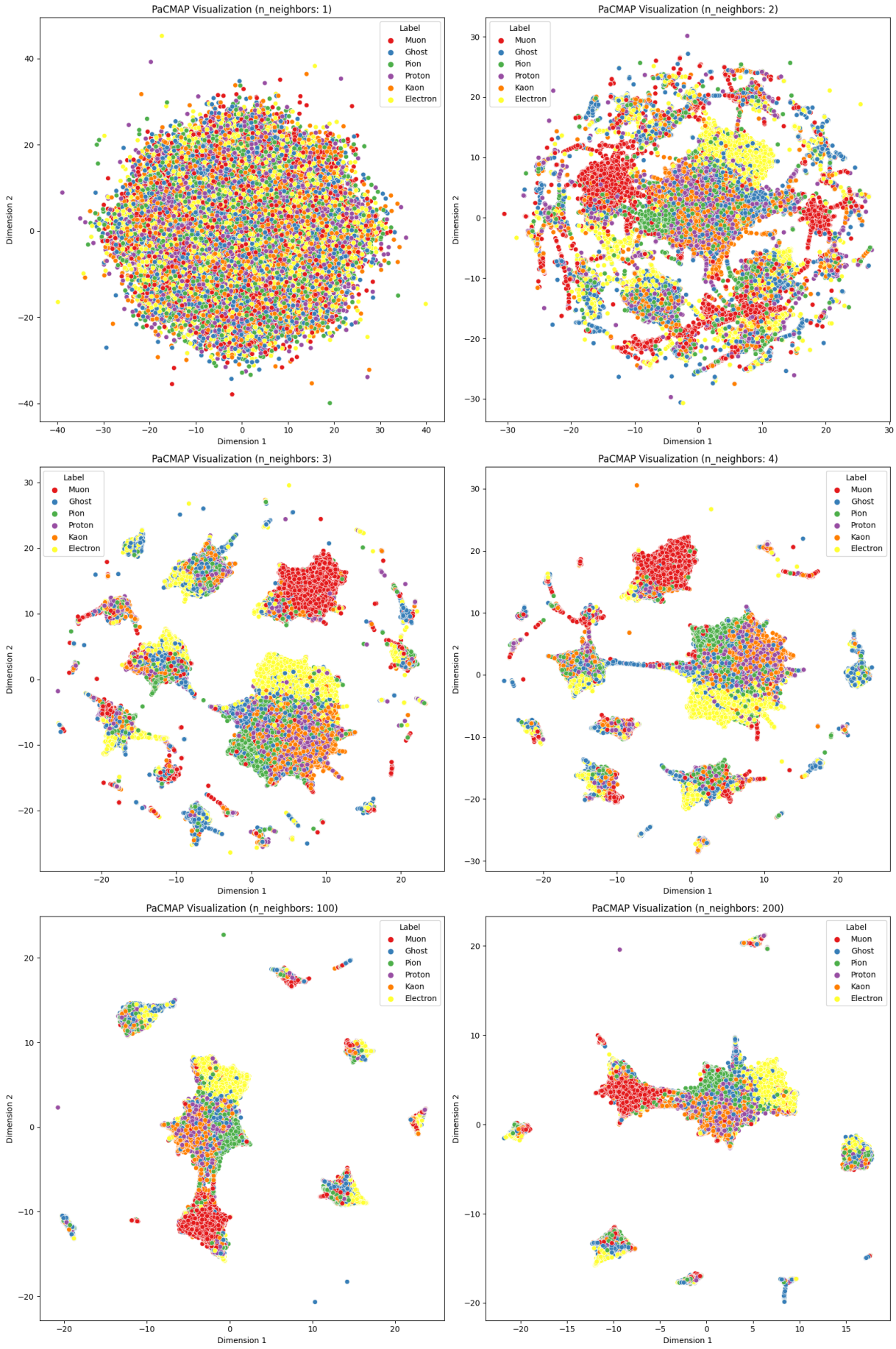

Figure 9: Visualization for different values of n_neighbor using PaCMAP.

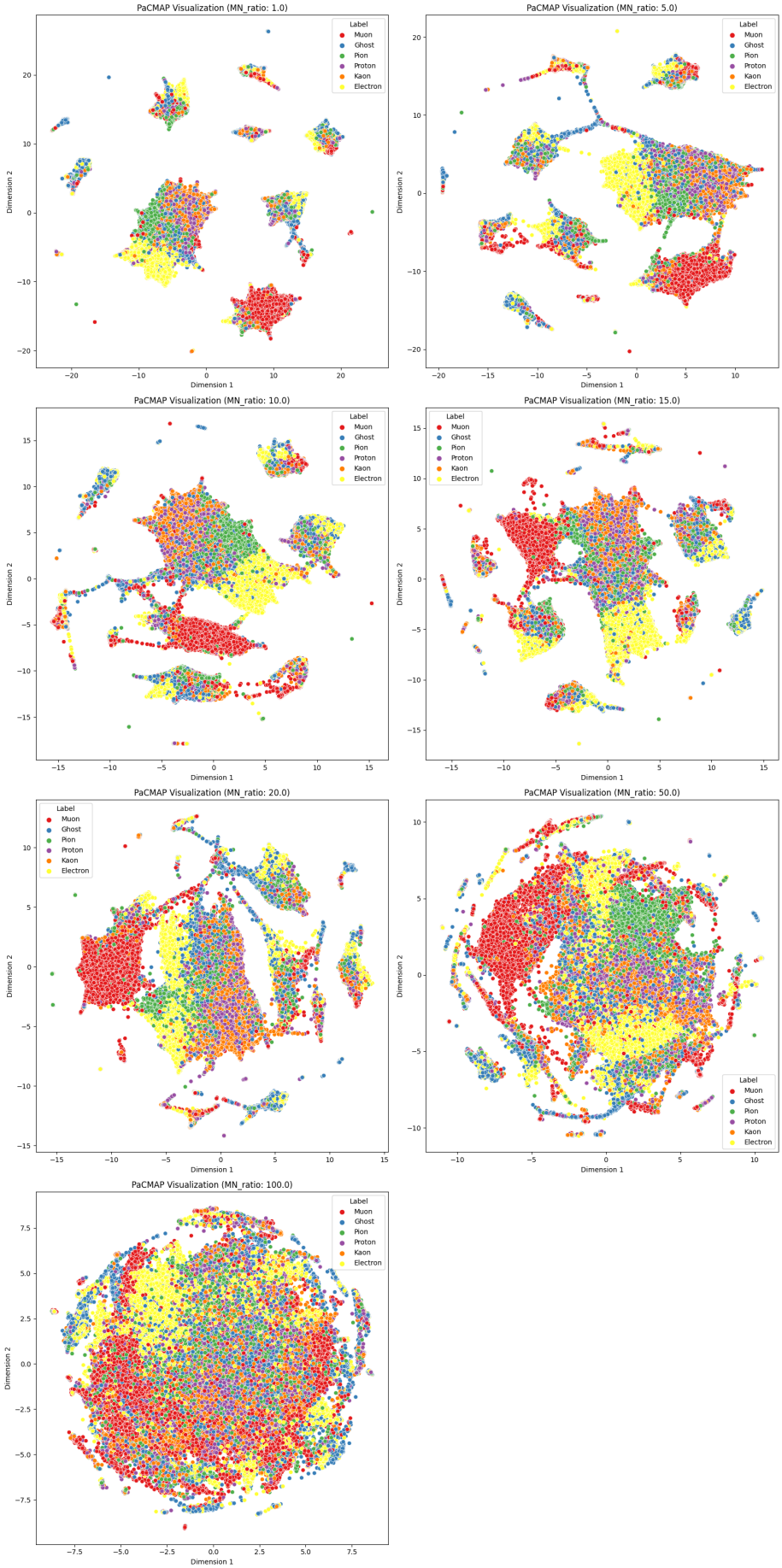

Figure 10: Visualization for different values of mn_ratio using PaCMAP.

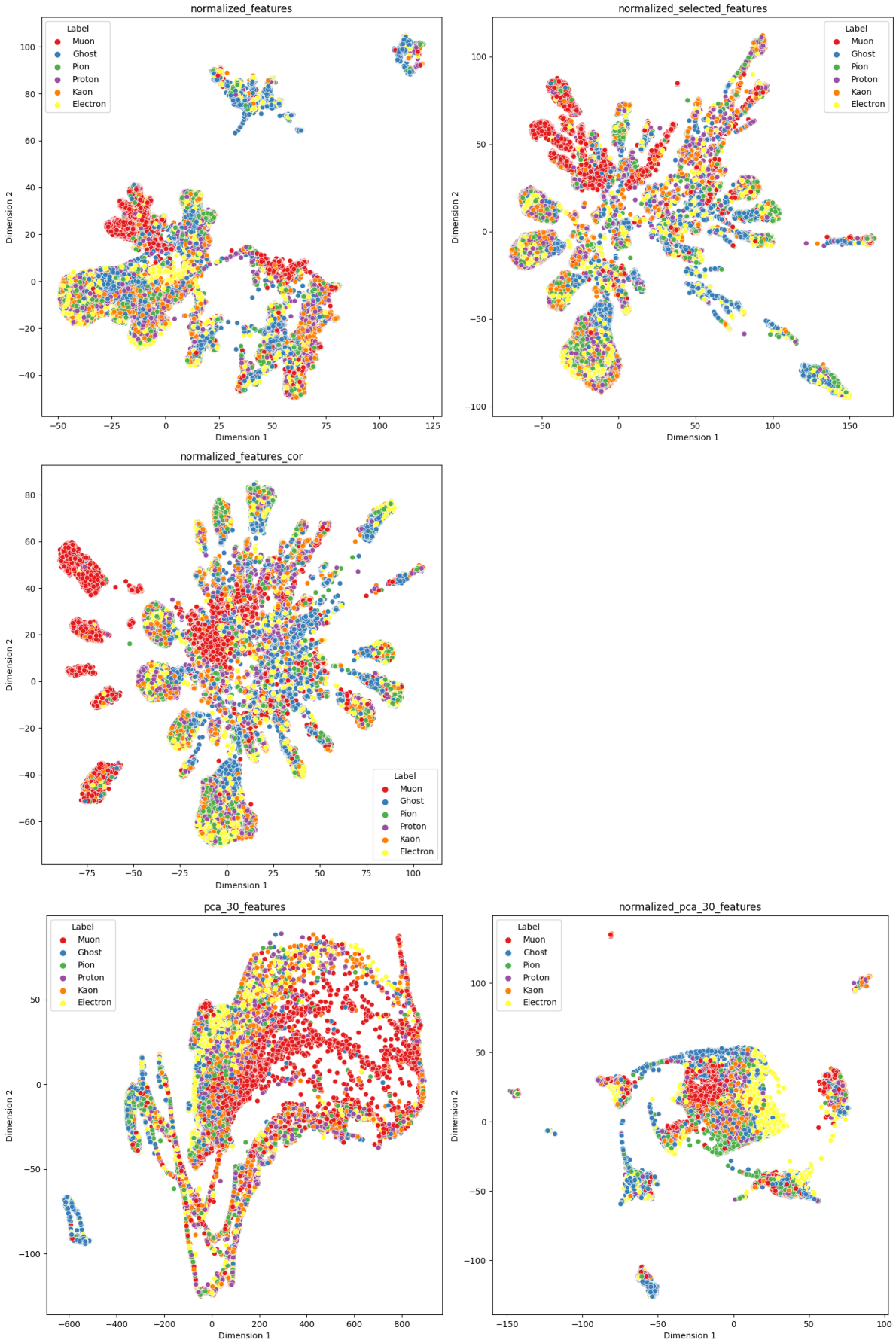

Figure 11: Visualization for various feature selection methods using TriMap.

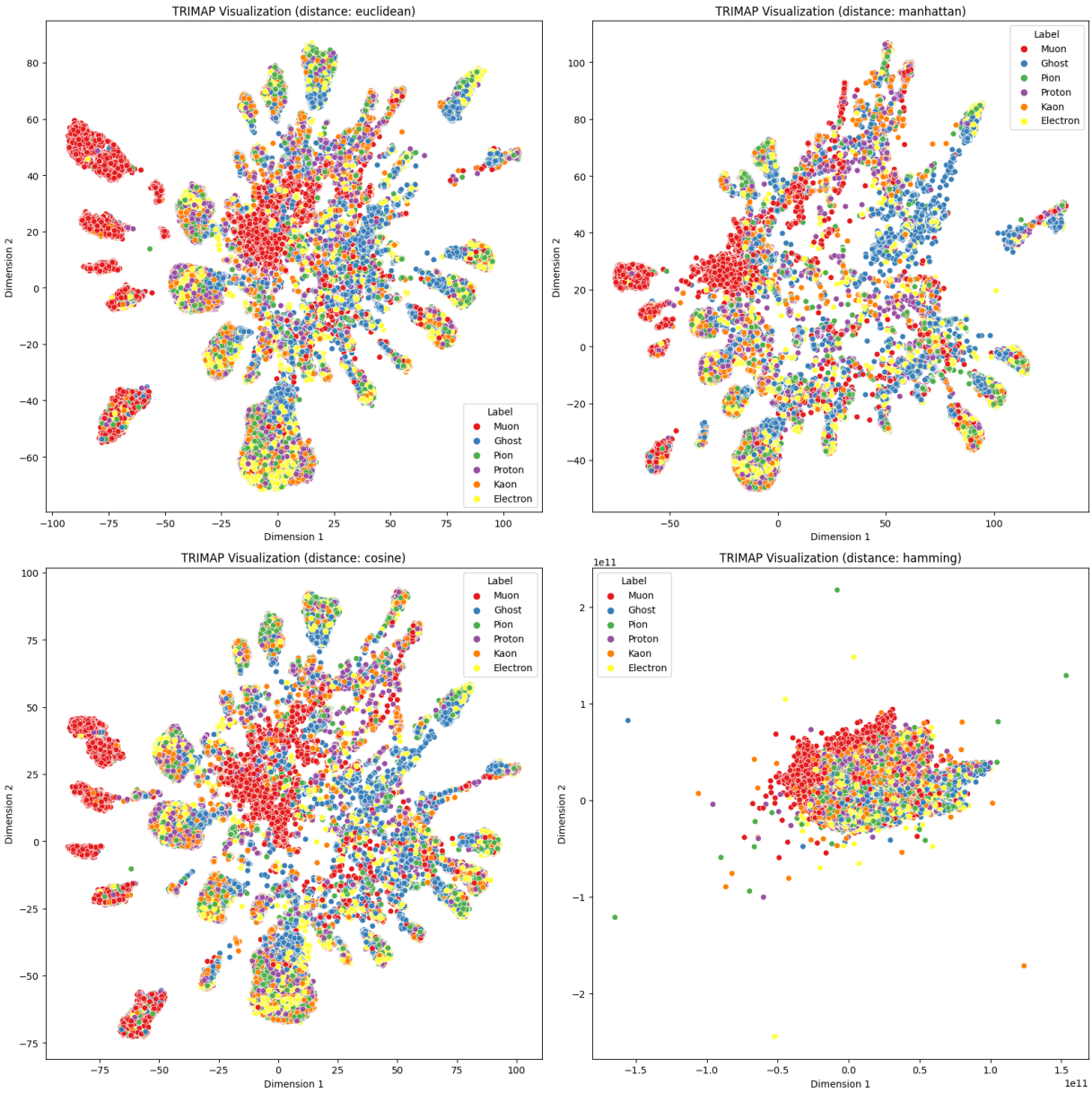

Figure 12: Visualization for different distance using TriMap after correlation analysis.

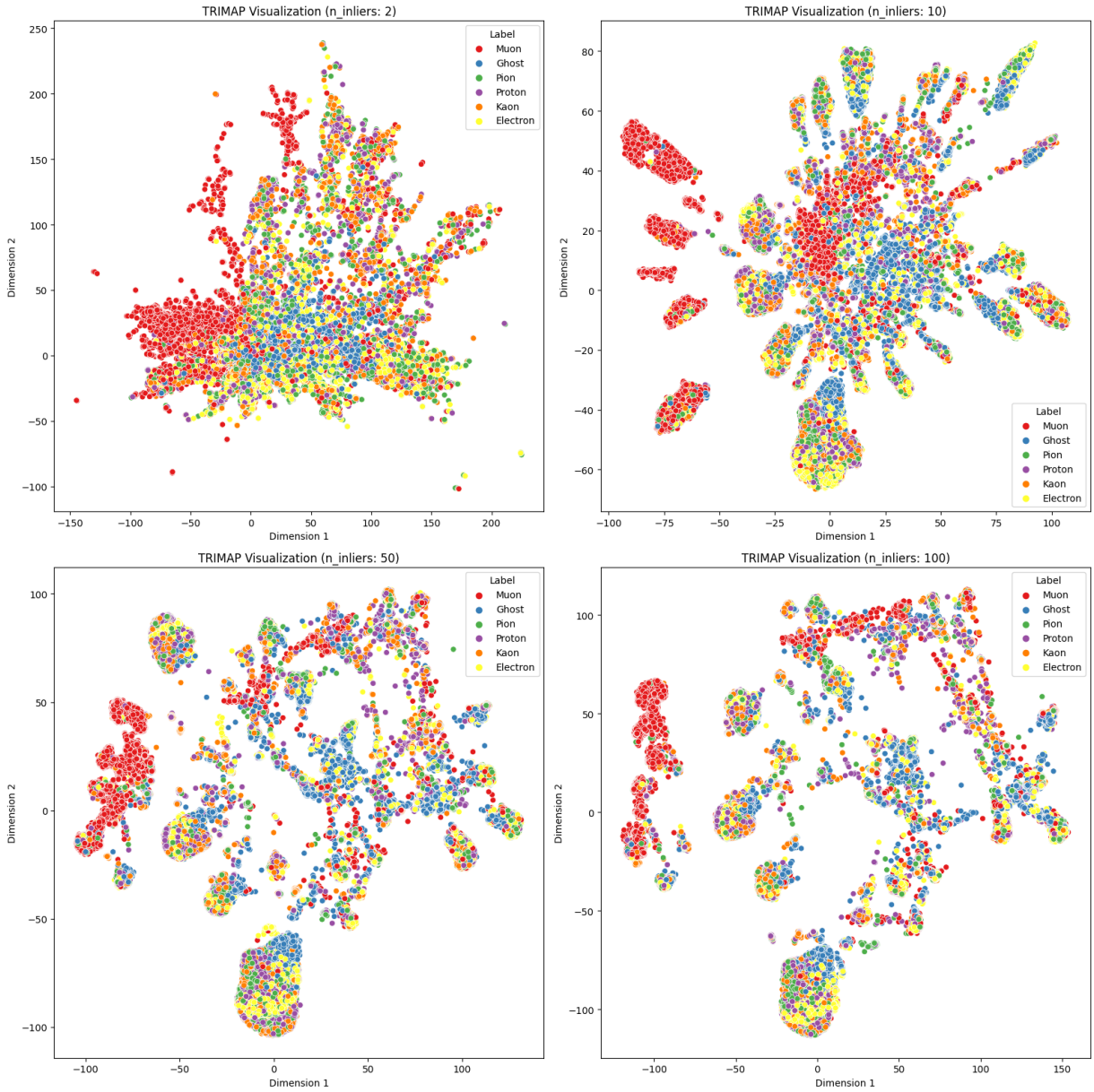

Figure 13: Visualization for different values of n_inliers using TriMap after correlation analysis.

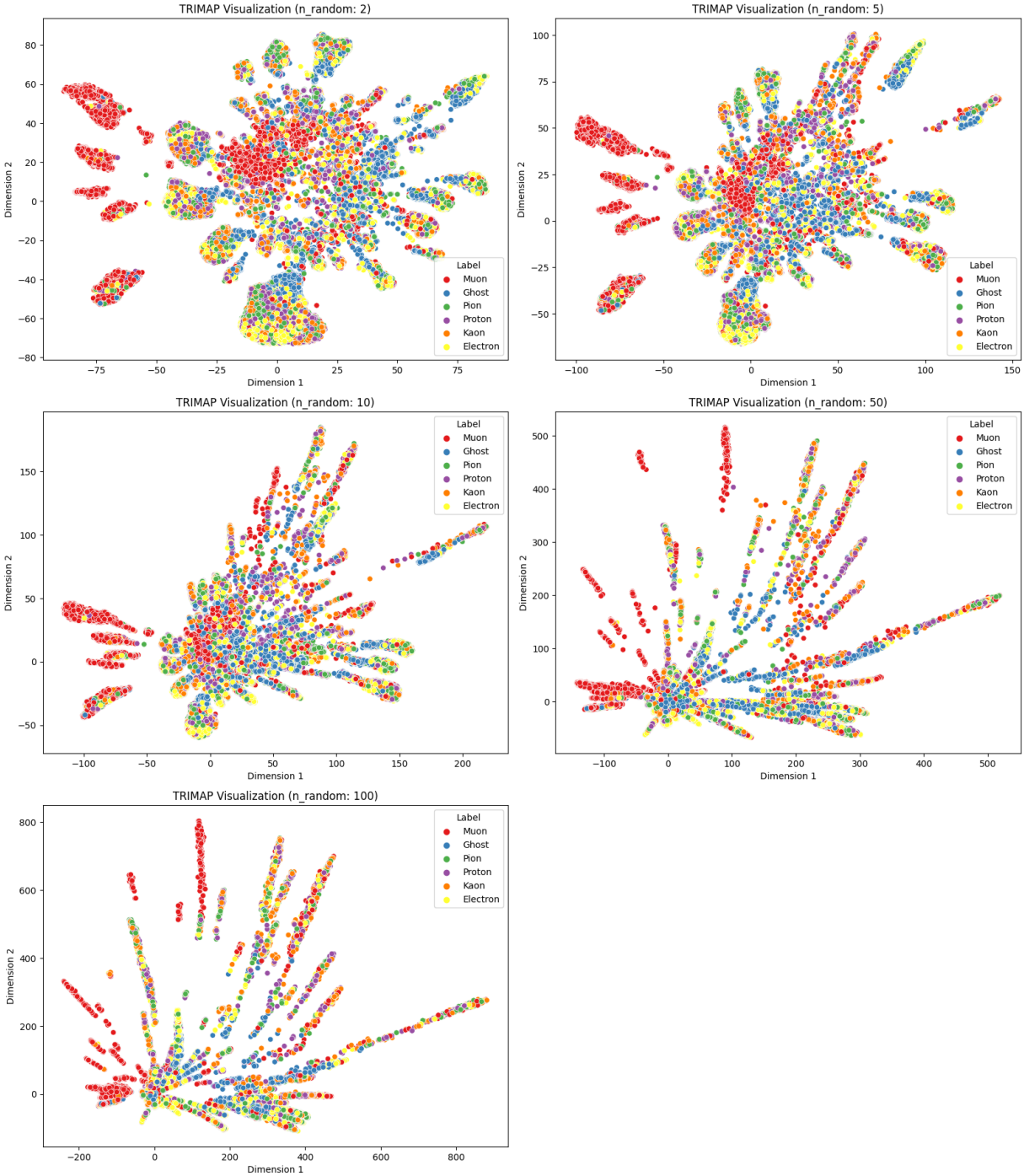

Figure 14: Visualization for different values of n_random using TriMap after correlation analysis.

6 Cluster analysis

Analyzing the results of all dimensionality reduction techniques, we notice interesting patterns in which the clusters are build:

- Muons (red dots) can be easily separated from the other particles and clustered, most likely due to the fact that only this particle penetrates matter so easily that it can be detected in the outermost layer of the detector (Muon chamber).
- Electrons (yellow dots) generally create various clusters.
- Other particles are very hard to separate, especially Kaons (orange) and Protons (purple). Pions (green) tend to form only sporadic clusters in the visualizations.

To analyze these patterns, we will take a closer look at the features for two mixed clusters shown in the Figure 15. We want to investigate which features remain similar in one cluster and differ in another. This analysis aims to provide insights into the reasons why particles tend to form mixed clusters. By investigating the features and patterns exhibited within these clusters, we can gain a better understanding of the factors driving their formation and composition.

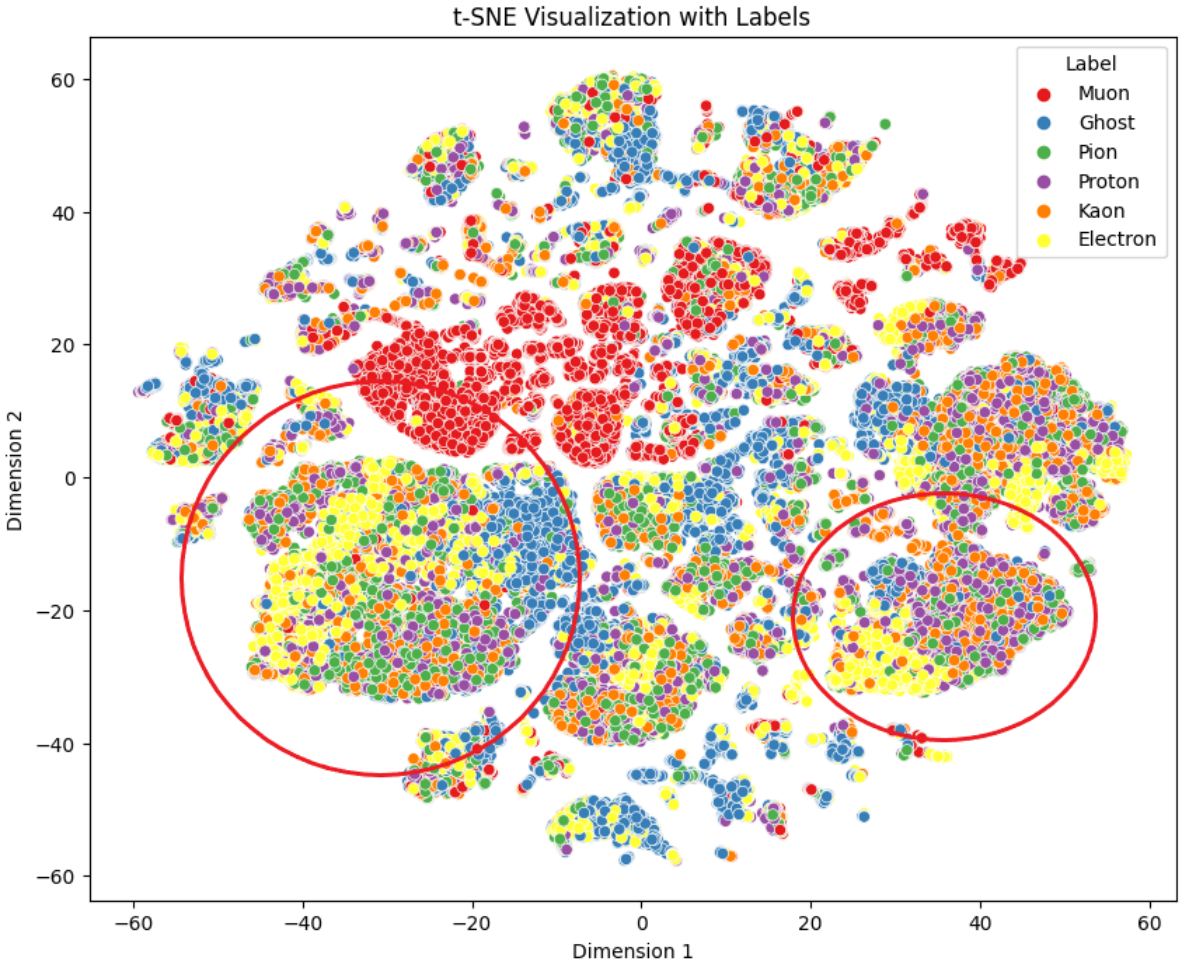

Figure 15: T-SNE for data after SelectKBest feature selection. 2 clusters were chosen for cluster analysis.

In the Figure 16 we pictured statistics for 3 selected features, which showed large difference between 2 clusters:

- TrackP - particle momentum
- TrackPt - particle transverse momentum
- EcalE - energy deposit associated to the track in the electromagnetic calorimeter (Ecal)

In both clusters we compared the values of TrackP for all particles and showed in the Figure 17. These histograms reveal similar distributions of TrackP among different particles within the same cluster. This observation strongly suggests that the mixed clusters originate from Pions, Kaons, and Protons with comparable energies and momenta.

Furthermore, we performed a comparison of the EcalE, TrackP, and TrackPt values for electrons in both clusters, as illustrated in Figure 18. There we observed distinct distributions for all three features in separate clusters.

These findings show that the origin of the clusters lies in the similarities of energy and momentum values among the particles, surpassing the significance of their individual characteristics. This explains why the dimensionality reduction techniques in this study fail to separate the particles based on their labels, but instead result in the formation of diverse mixed clusters.

Cluster 1:

	TrackP	EcalE	TrackPt
count	2374.000000	2374.000000	2374.000000
mean	5002.424464	3656.323199	422.110972
std	1659.101456	1883.797105	290.930393
min	2296.370133	-999.000000	47.174171
25%	3657.235108	2413.183047	217.928243
50%	4664.459957	3302.664058	329.566736
75%	6057.175163	4504.897454	537.562203
max	10832.030283	22475.523441	2152.685791

Cluster 2:

	TrackP	EcalE	TrackPt
count	873.000000	873.000000	873.000000
mean	13298.422911	10811.537750	1019.799820
std	2702.837969	5582.807094	532.046837
min	8470.580056	1299.167967	283.165751
25%	10959.780273	7209.368163	630.792408
50%	12985.759752	9026.358409	873.246030
75%	15347.530264	12515.328135	1278.468031
max	20869.150405	48539.148438	3521.158682

Figure 16: Statistics for selected features: EcalE, TrackP and TrackPt in 2 clusters showed in the Figure 15

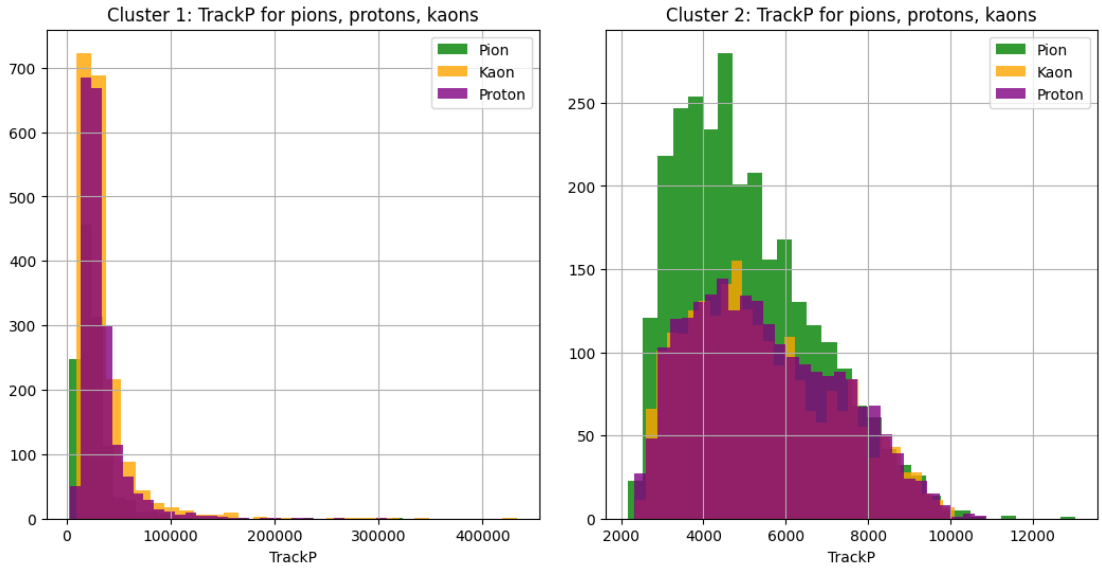

Figure 17: T-SNE for data after SelectKBest feature selection. 2 clusters were chosen for cluster analysis.

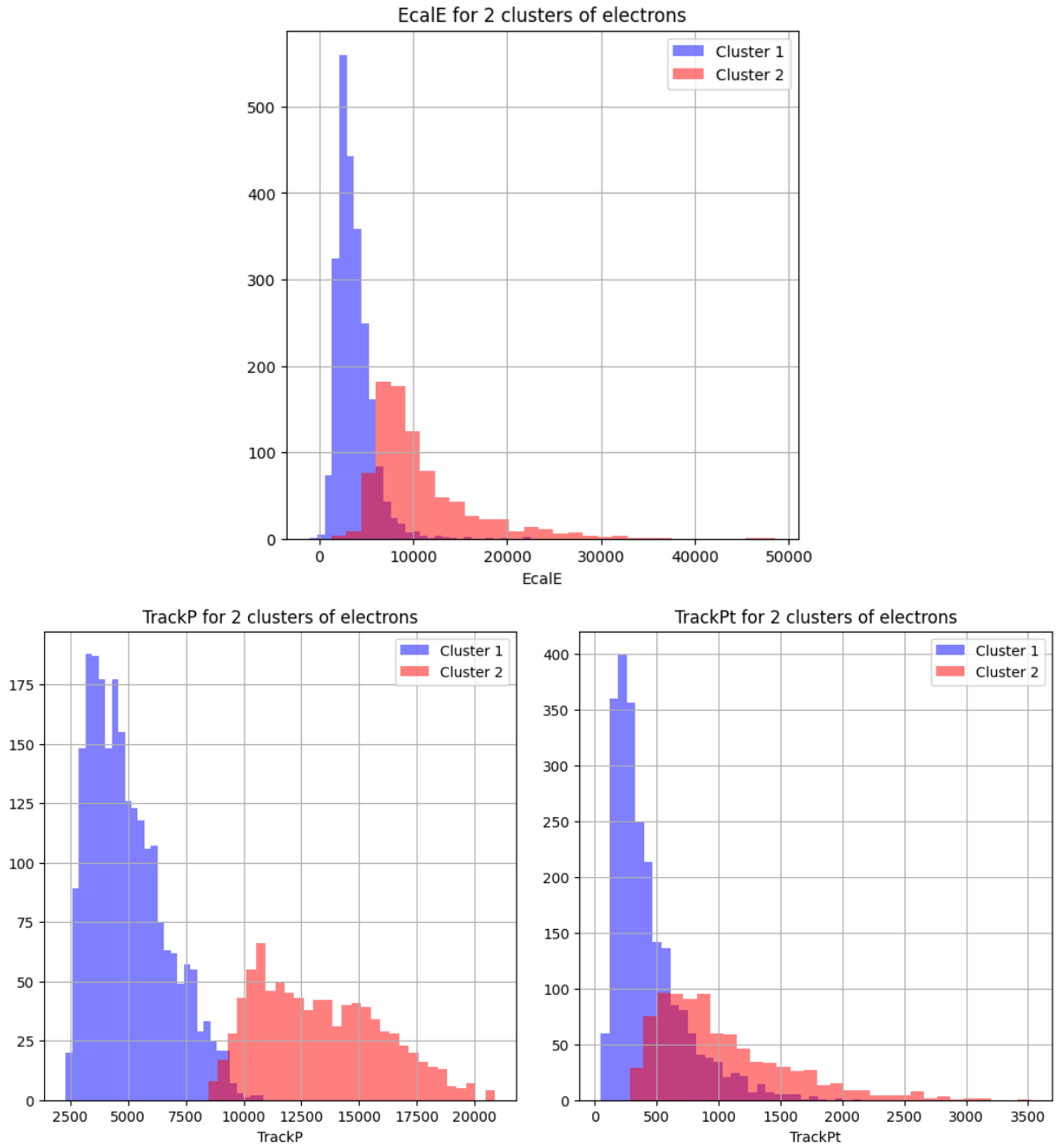

Figure 18: Comparison of selected features: EcalE, TrackP and TrackPt for electrons in separate clusters.

7 Performance Calculations

To compare the performance of dimensionality reduction techniques, we used 1 million rows of our dataset for each method. Our objective was to assess how these methods handle large-scale dataset. The result of our experiment can be found in the Table 1.

Technique	Time [min]	Memory [MB]
t-SNE (Rap. AI)	27.83	8.22
UMAP (Rap. AI)	1.88	7.79
PaCMAP	52.30	336.14
TriMap	122.43	9.36

Table 1: Time and memory usage calculations for four dimensionality reduction techniques for a large-scale dataset (1 mln rows)

In the Figures 19, 20, 21 and 22 the visualizations for 1 million rows were shown. The hyperparameters for this experiment were chosen based on the analysis made in the "Visualizations" chapter:

- t-SNE: n_components=2, random_state=42, perplexity=30, metric="minkowski",
- UMAP: n_components=2, learning_rate=0.1, metric="cosine", min_dist=0.5, random_state=42, n_neighbors=50,
- PaCMAP: n_components=2, n_neighbors=10, MN_ratio=0.8, FP_ratio=2.5,
- TriMap: n_inliers=20, n_outliers=5, n_random=5, distance='manhattan'

and the rest parameters have default values.

Comparing the runtime and memory usage of all techniques, the UMAP method from Rapids AI performed the best in our case. However, using this method, we were unable to observe any well-defined clusters as pictured in the Figure 20. Only a slight accumulation of Muons on the outer side of the circle could be observed.

Another method from Rapids AI, t-SNE, required more time and memory resources but performed significantly better in separating clusters (Figure 22). Just as described in the previous chapter, „Cluster Analysis", we can observe clustering of Muons (red) and Electrons (yellow) and a mixed cluster of other particles: Pions, Kaons, and Protons.

Methods that do not utilize GPUs, such as Rapids AI, tend to have the longest runtime and highest memory usage. Among them, PaCMAP stands out as the most memory-consuming technique, utilizing approximately 0.33 GB. As a result (Figure 21) we've got one defined cluster of Muons (red) and some mixed clusters, where Electrons (yellow) and Pions (green) accumulate in specific regions, Notably, PaCMAP demonstrates the best ability to separate Pions within the data.

TriMap consumed significantly more memory than PaCMAP, but had the highest runtime of over 2 hours. In result, we also didn't achieve well-defined clusters as shown in the Figure 22.

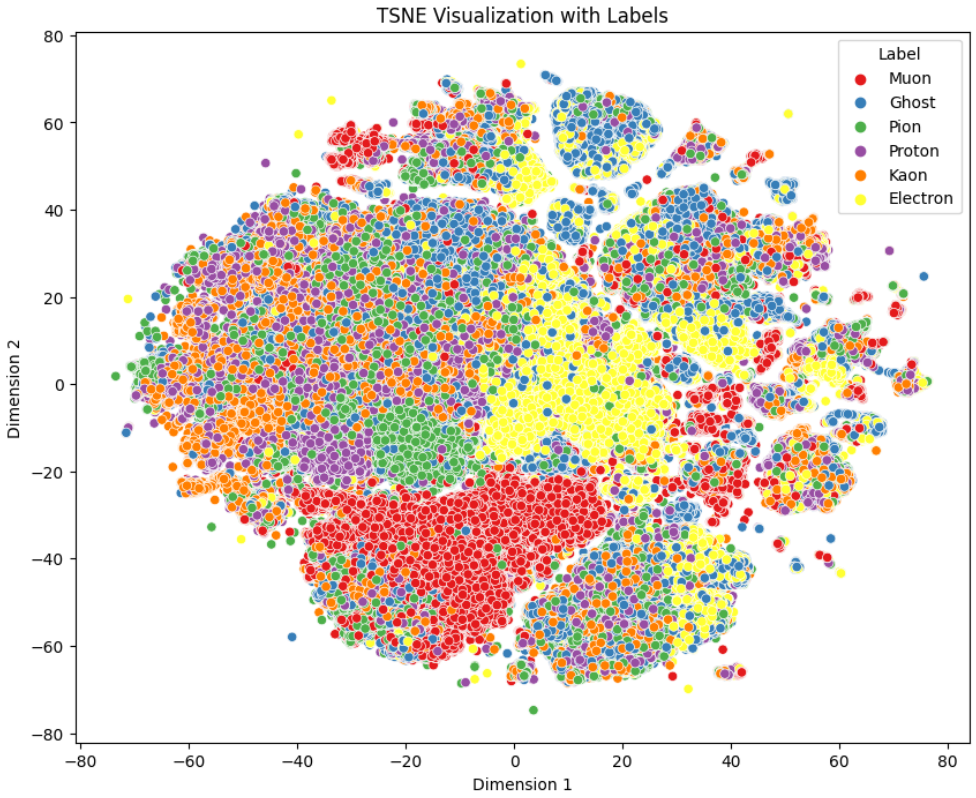

Figure 19: tSNE visualization for 1 millions records.

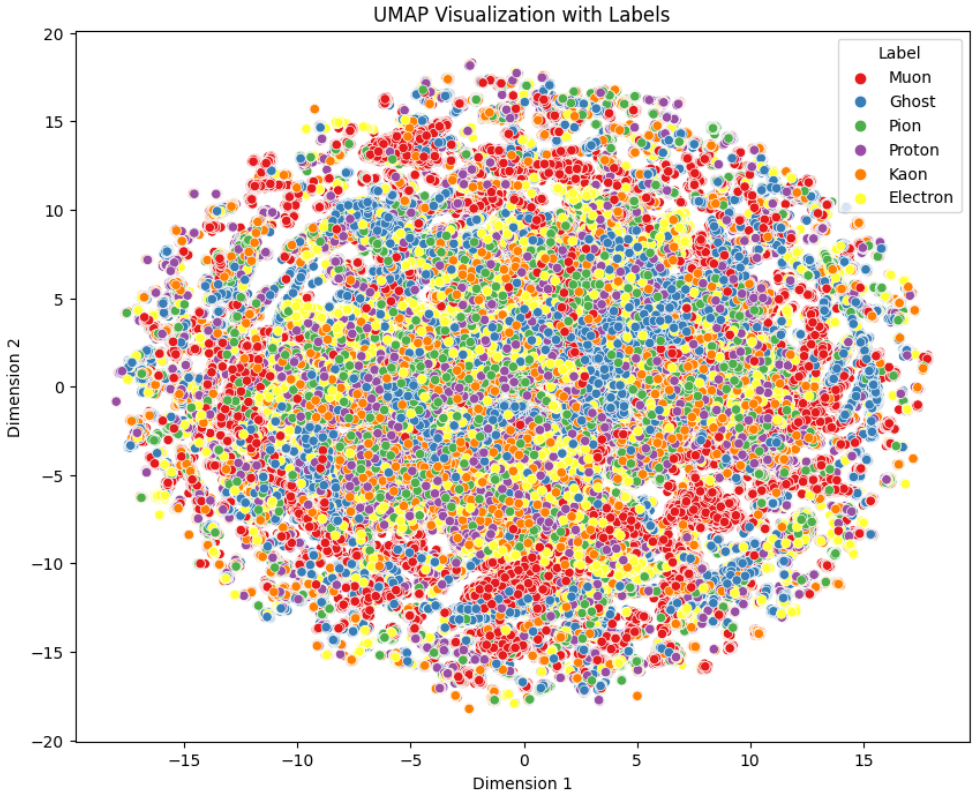

Figure 20: UMAP visualization for 1 millions records.

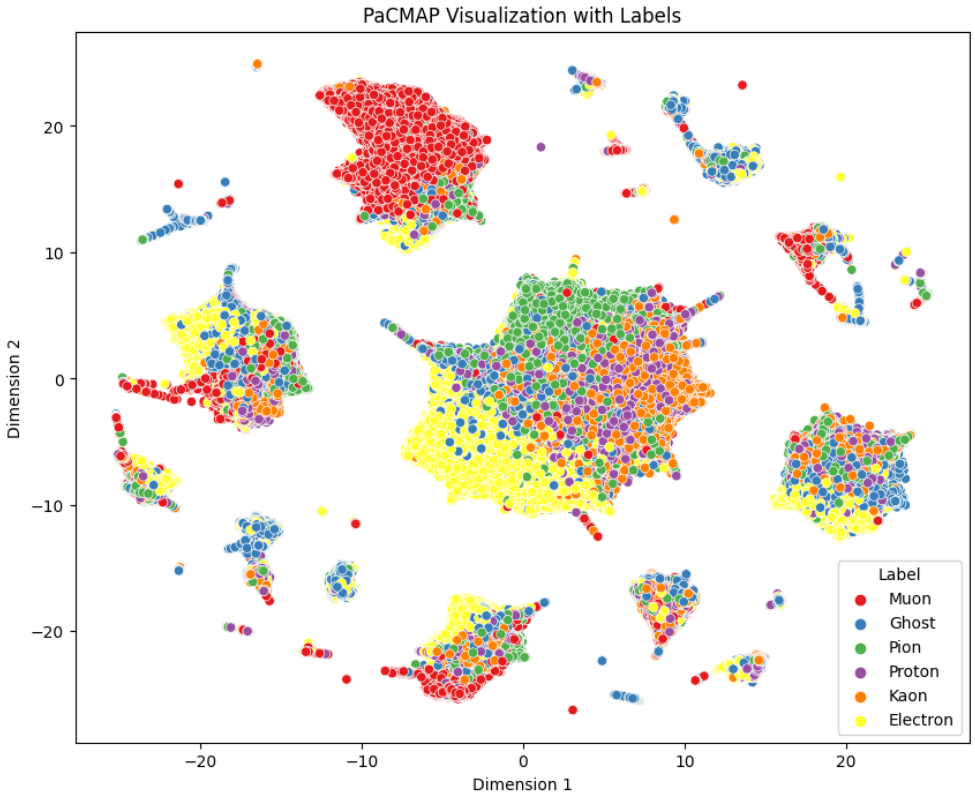

Figure 21: PaCMAP visualization for 1 millions records.

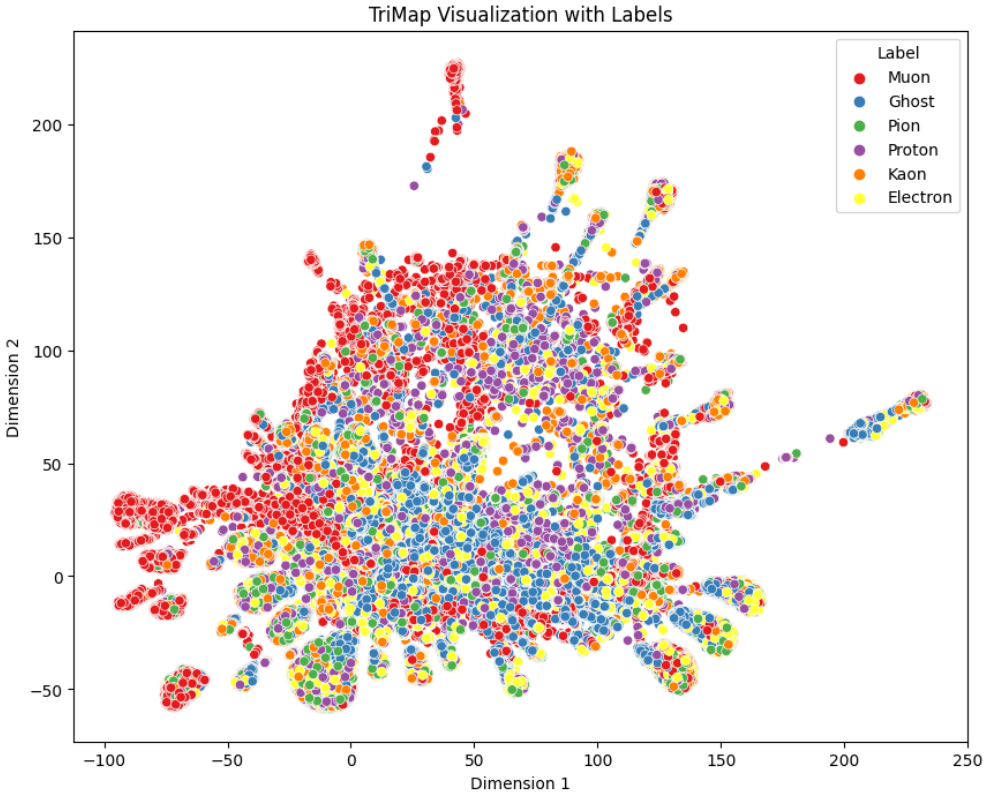

Figure 22: TriMap visualization for 1 millions records.

8 Summary

After conducting Principal Component Analysis (PCA) with 20D and 30D on a dataset consisting of 50 dimensions, we observed a significant loss of information. This loss of information made it challenging for t-SNE to generate well-defined clusters. For further feature selection (using SelectKBest and Correlation Analysis) 35 most important features were chosen.

For various visualization techniques, we were able to select the optimal features and hyperparameters for each respective technique. The selection process was conducted based on iterative visualizations of different feature and hyperparameter combinations for the subset of 100,000 data points. During this process, we selected feature and hyperparameter combinations that showed the most optimal results of clustering.

To gain deeper insights into the patterns in which the clusters are formed, we performed additional experiments. The analysis demonstrated that the formation of mixed clusters is primarily influenced by the similarities in energy and momentum values among particles, outweighing their individual characteristics. Consequently, the dimensionality reduction techniques employed in this study struggle to separate particles based on their labels, resulting in the emergence of diverse mixed clusters.

Finally, we performed the performance calculations for all techniques using 1 million rows of data. UMAP from Rapids AI demonstrated efficient resource usage, while t-SNE from Rapids AI provided better cluster separation. PaCMAP excelled at separating Pions, but had higher memory consumption. TriMap exhibited the longest runtime without achieving distinct clusters. Methods not utilizing GPUs (PaCMAP and TriMap), generally exhibited longer runtimes and higher memory usage.

References

[1] *Dark Matter / Electromagnetic Showerss*. 2019. URL: https://www.kaggle.com/datasets/lavanyashukla01/dark-matter-from-opera-experiments (visited on 06/23/2023).

[2] *PaCMAP*. URL: https://pypi.org/project/pacmap/ (visited on 06/24/2023).

[3] *Rapids AI*. URL: https://rapids.ai/ (visited on 06/24/2023).

[4] *TriMap*. URL: https://pypi.org/project/trimap/ (visited on 06/24/2023).