Practical Machine Learning – Project 2

Anghelina Ion-Marian, 407

Task Description

This project's task involves consists of two main parts. The first is using models for feature extraction from a complex dataset and the second is using unsupervised clustering methods on the feature vectors. The results should then be compared with the random baseline (which should be a strict lower bound) and a supervised approach (which should be a strict upper bound)

Dataset Description

The dataset was chosen from the Kaggle website. It was used in a classification task. The samples are RGB images of different shapes representing sports balls and the labels are the sport in which every particular ball is used. There are 15 labels for this task.

The dataset is accessible by the following link: https://www.kaggle.com/datasets/samuelcortinhas/sports-balls-multiclass-image-classification.

It is already separated into train and test. The test data (which is also labeled) will be used as validation

Data Preprocessing

Prior to the feature extraction process, all images were resized using the OpenCV library to a standard shape.

In the case of ViT feature extraction, the images were resized to 224x224 to match the pretraining image size.

In the case of HOG, experiments with more dimensions were conducted, the best results being obtained for an image size of 128x128.

The resize function provided by the OpenCV library conserves the important features of an image while rescaling its dimensions.

Prior to the supervised classifier training on the feature vectors, they were scaled, using the StandardScaler function, provided in the Scikit-Learn library. The vectors had all components in the range [0,1] which led to better numerical stability and a more accurate understanding of feature importance in the learning process.

Feature extraction methods

1. Histogram of Oriented Gradients (HOG)

Partial derivatives are computed for each pixel. The gradient magnitude and orientation are then computed. The pixels are then split into submatrix regions, and histograms are computed by grouping values of the magnitude into bins with respect to the values of orientation.

Then, a sliding window will extract information about the histograms in adjacent blocks and construct the final feature vector.

2. Vision Transformer Embeddings (ViT)

Vision Transformer architectures are inspired from Transformer models used in Natural Language Processing. Similar to how the Bag of Visual Words model works, the image is split into patches, each represented by a vector. Each patch is then represented by an embedding vector, with respect to its values and its position in the picture. The model, is, then, able to capture dependencies between all the patches in the picture to compute accurate embeddings.

The version used in this code uses a patch size of 16x16 and an image size of 224x224. It is pretrained on the ImageNet_21K dataset, a dataset containing various images, classified into more than 21 thousand classes.

Prior to extracting features, input images were scaled on each of their three color channels, using mean = 0.5 and std = 0.5. This standardization was also used in the pretraining step.

Random and Supervised Baselines

For the random baseline, I assigned the most frequent label to all the samples, using sklearn's DummyClassifier module.

For the supervise baseline, which should serve as an upper bound for the accuracy of the model, I used the Random Forest classifier with a number of 1000 estimators (the default value).

The supervised baselines were:

Model	Extractor	Accuracy	Precision	Recall	F1-Score
RandomForest(1000 estimators)	HOG	0.41	0.40	0.48	0.42
RandomForest(1000 estimators)	ViT	0.842	0.84	0.85	0.84

As for the random choice baseline:

Accuracy	Precision	Recall	F1-Score
0.087	0.07	0.01	0.01

Unsupervised methods

1. K-Medoids

This clustering method is similar to the K-Means method, however, the main difference is that each cluster's representative point (a medoid) is always one point in the set.

At the first step, each cluster is randomly selected a medoid. In subsequent steps, the algorithm finds the pair (x,y) where x is a cluster's medoid and y is not, which minimizes the cost function. If such a pair exists, the swap is made, it not, the algorithm terminates.

2. Agglomerative Clustering

This clustering methods starts with n clusters and merges the closest ones with respect to a predetermined distance metric.

Hyperparameter Tuning

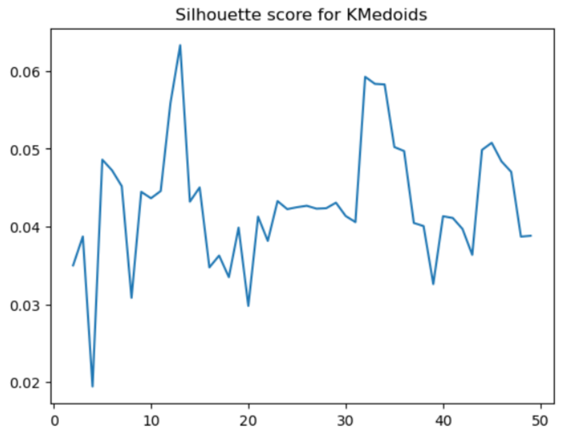
The number of clusters and other hyperparameters for each model was chosen by testing various values on the training set. The metrics used for testing are the following:

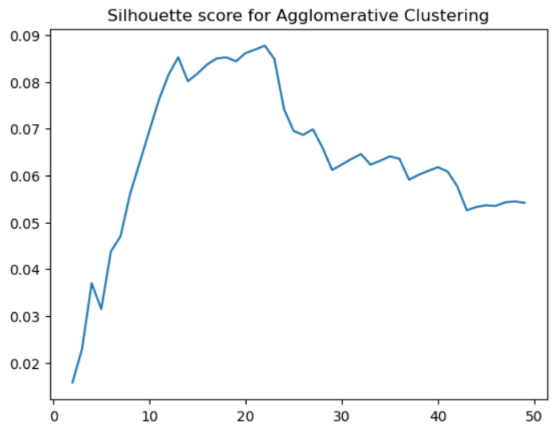
Silhouette Score

This metrics keeps track of the average distance between points inside each cluster and distances between closest different clusters. A bigger value of this index means the clusters are better separated and, thus, will lead to a better clustering.

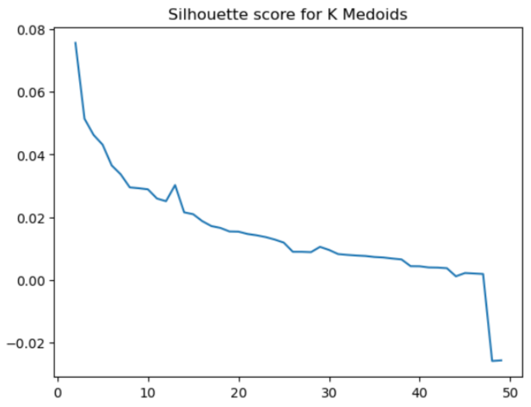
The values of the silhouette scores for different values of the number of clusters for each model.

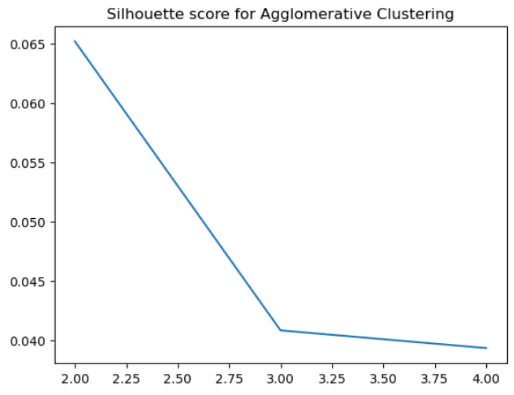
For ViT extraction:





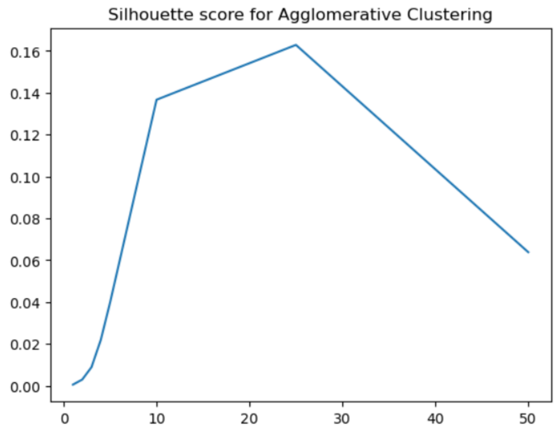
For HOG extraction:



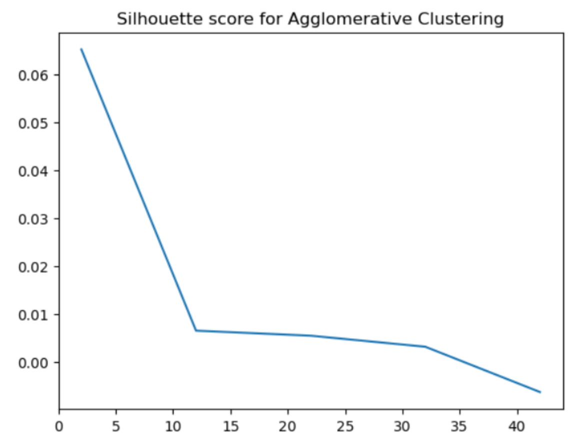


In the case of agglomerative clustering, the Silhouette Score can also be evaluated by tuning the minimum distance threshold for matching and with various distances.

For ViT model



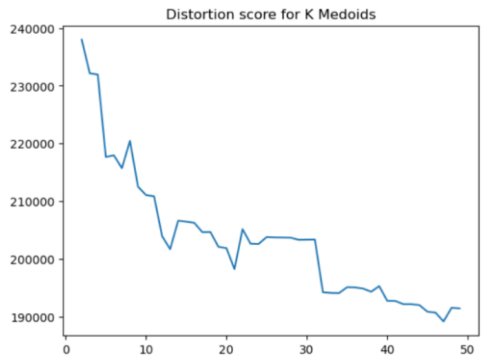
For HOG model



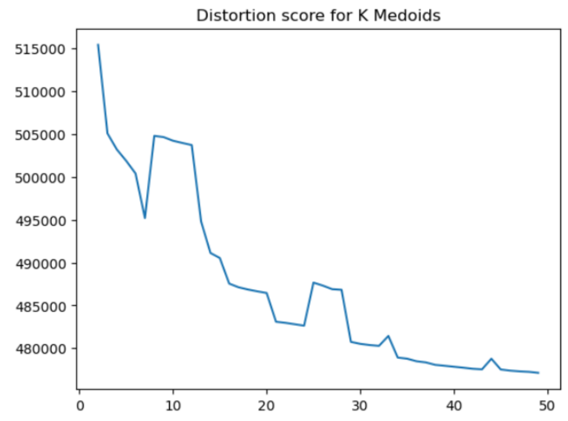
Distortion Score

This metric represents the sum of distances between each point and its cluster's representative point (in this case, a medoid). We can only use this method for K-Medoids.

For ViT Model:



For HOG Model:



The best hyperparameters obtained were evaluated on the test set using the same metrics, along with Rand Index, Adjusted Rand Index and Davies Bouldin Score.

Rand Index

Is a method of similarity of two clustering algorithms. It's computed taking into consideration pairs which share a cluster in both the algorithms' outputs, or not. Here it is used in contrast to the ground truth. A score closer to 1 signifies a better clustering.

Adjusted Rand Index

Works similarly to rand index, but, in addition, accounts for pairs which can be clustered together "by chance". It does this by considering multiple random clusterings. A score closer to 1 signifies a better clustering.

Davies Bouldin Score

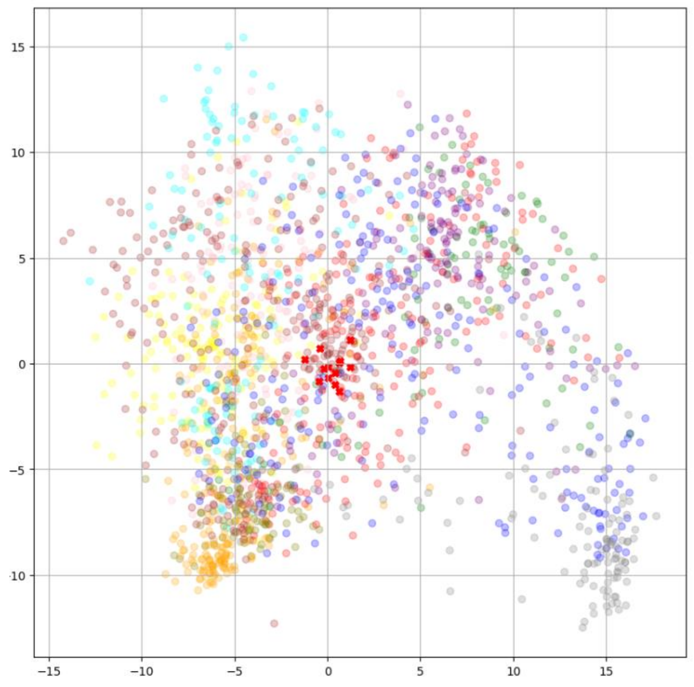
It computes similarity between each cluster and the cluster closest to it, in terms of intra-cluster and inter-cluster distances. A score closer to 1 signifies a better clustering.

The results on the test set are:

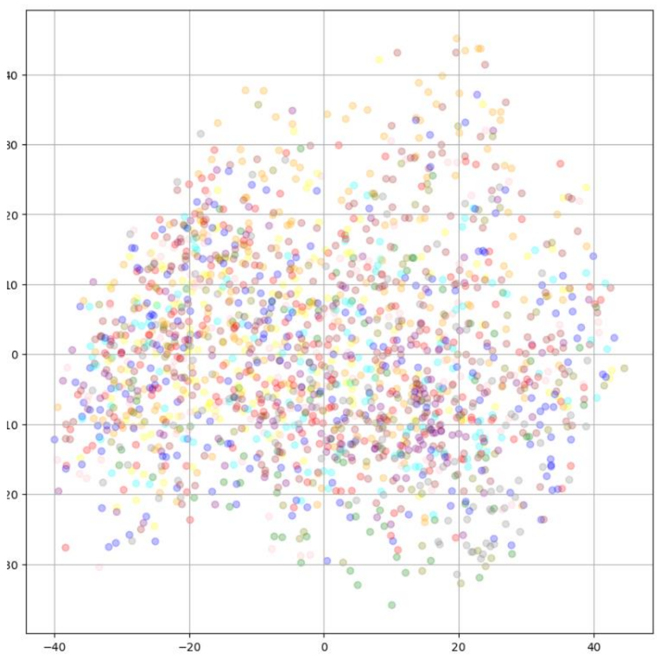
Features	Model	Hyperparameters	Rand Score	Adjusted Rand Score	Silhouette Score	Distortion	DavisBouldin
ViT	K Medoids	K = 13	0.884	0.27	0.03	52202.54	4.996
ViT	K Medoids	K = 47	0.9278	0.3334	0.002	47669.06	3.72
ViT	Aggl. Clustering	K = 22	0.931	0.403	0.096	-	3.36
ViT	Aggl. Clustering	Eps = 25	0.933	0.045	0.07	-	0.695
HOG	K Medoids	K=2	0.4957	0.0057	0.07	131581.36	3.478
HOG	K Medoids	K = 49	0.8937	0.016	-0.02	118362.25	4.647
HOG	Aggl. Clustering	K = 2	0.4579	0.0034	0.056	-	3.554

HOG	Aggl. Clustering	Eps =50	0.9314	0.0015	0.04	-	0.473

The clustering's plot for Kmedoids with k = 13 and ViT Extractor. The medoids are marked in bright red.



The clustering's plot for Agglomerative Clustering with K=49 and HOG Extractor



To compare the clusters to the supervised baseline, I had to find a way to map cluster indices to true labels. As there could be more clusters mapped to the same class, and vice versa, a straightforward matching algorithm cannot be possible.

As a heuristic, I assigned each cluster's mapping as the most frequent true label of that cluster's sample.

The data found was the following

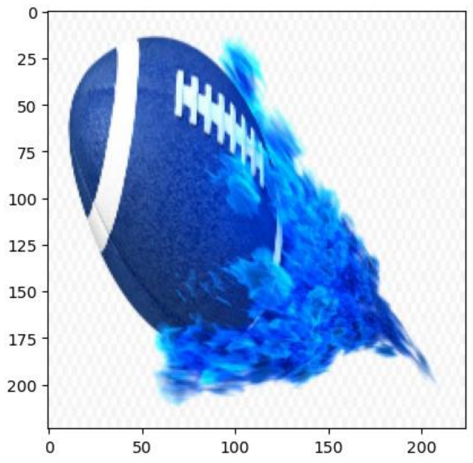
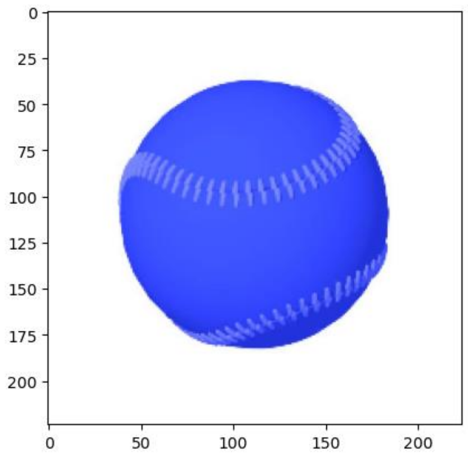
Model	Accuracy	Precission	Recall	F1-Score
ViT + Kmedoids + k = 13	0.4323	0.41	0.25	0.30
HOG + Aggl. Clustering + k = 49	0.2444	0.23	0.26	0.22

The results lie between the Random Baseline and the supervised baseline.

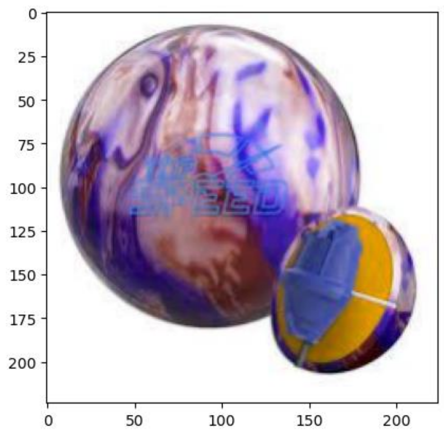
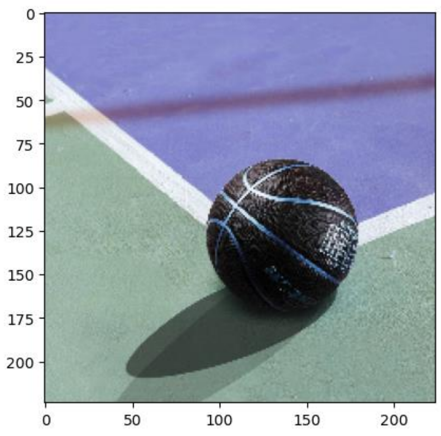
Cluster Interpretation

A first observation we could make is that the clusters are not perfectly when compared to their labels. This is a consequence of the high dimensionality of the feature vectors and also the high diversity between objects belonging to the same class.

The algorithm mistakes two items for being in the same cluster, based on their other features. For example, cluster 0 contains images of American Footballs and Baseballs, because of the similarity of their surface, sewing and color.

Cluster 1 contains Basketballs, Bowling Balls and Footballs, each having relatively large dimensions and being relatively round and multicolored.

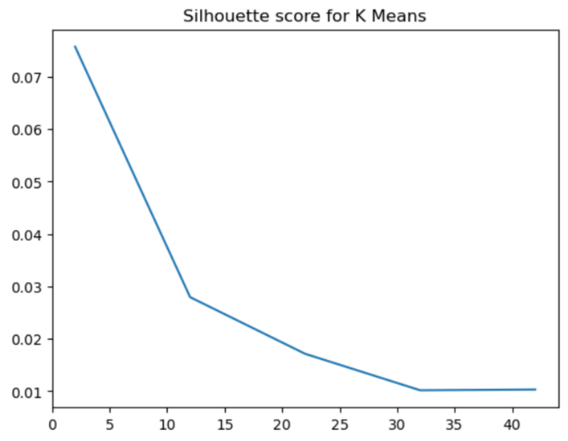
The clustering algorithms will make use of any feature to cluster objects, even if it is not linked to the task in hand.

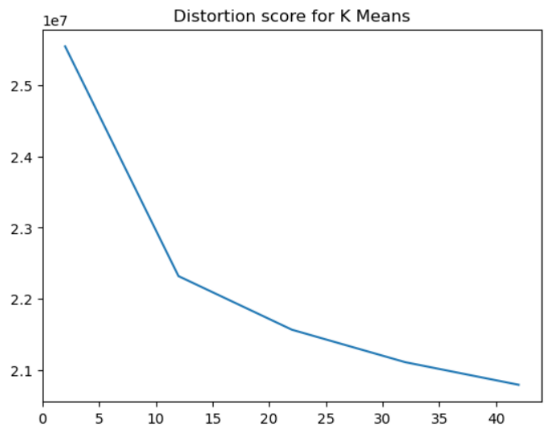
Bonus Model: K Means

Similar to K Medoids, but the representant of each cluster is not necessarily an instance within the cluster, but the mean of all the points in it.

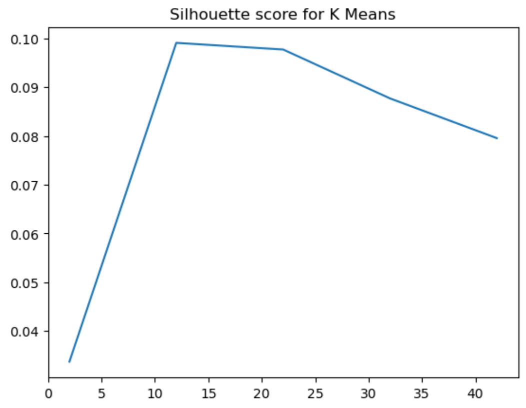
It is incrementally build by assigning points to clusters and recomputing centroids.

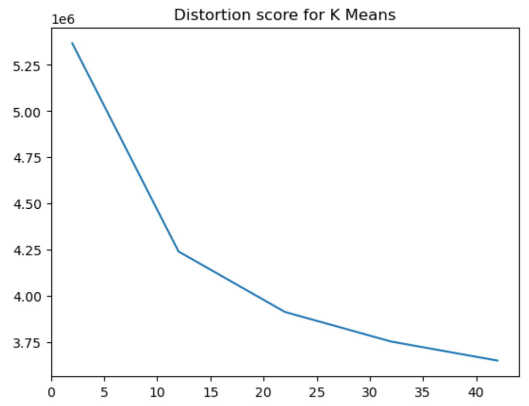
For the HOG Extractor





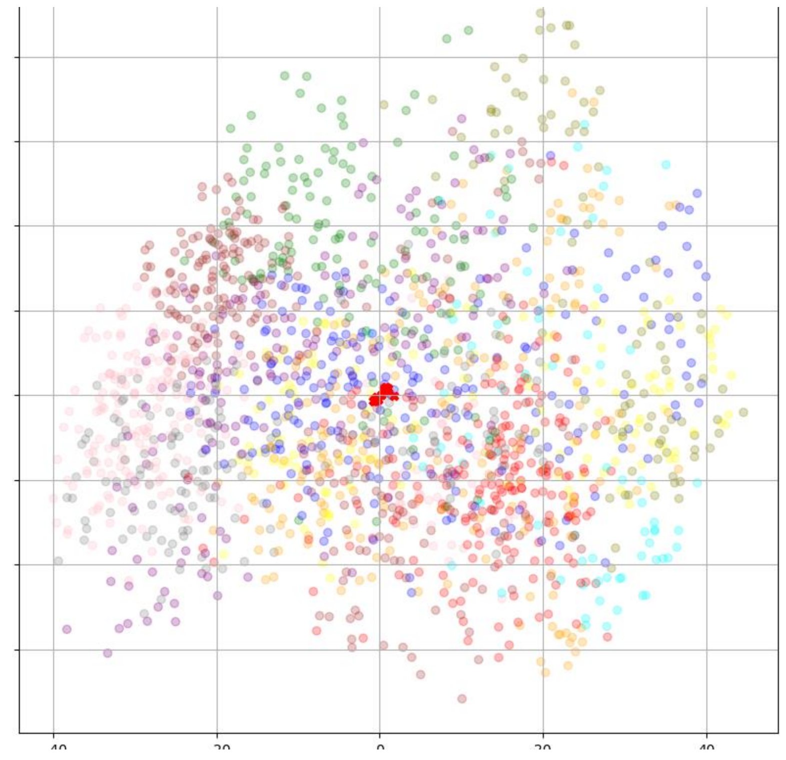
For the Vit Extractor



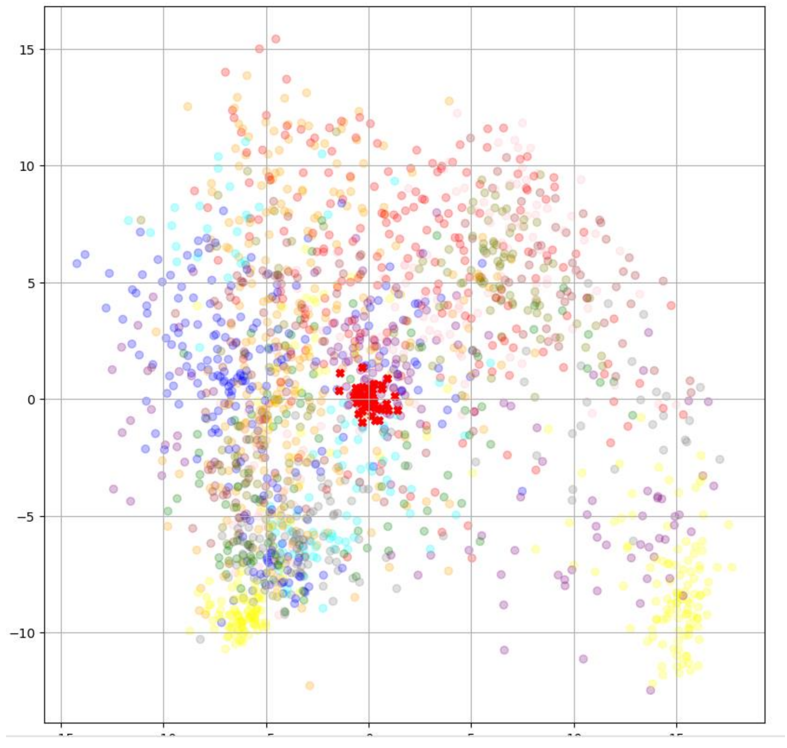


Features	Model	Hyperparameters	Rand Score	Adjusted Rand Score	Silhouette Score	Distortion	DavisBouldin
HOG	K Means	K = 2	0.50	0.004	0.074	6476006.5	3.43
HOG	K Means	K = 42	0.893	0.020	0.015	5197879.0	3.749
ViT	K Means	K = 12	0.8966	0.3371	0.099	1066006.0	3.052
ViT	K Means	K = 42	0.9369	0.344	0.07	911893.1875	3.008

Graph for HOG feature extractor and k = 42



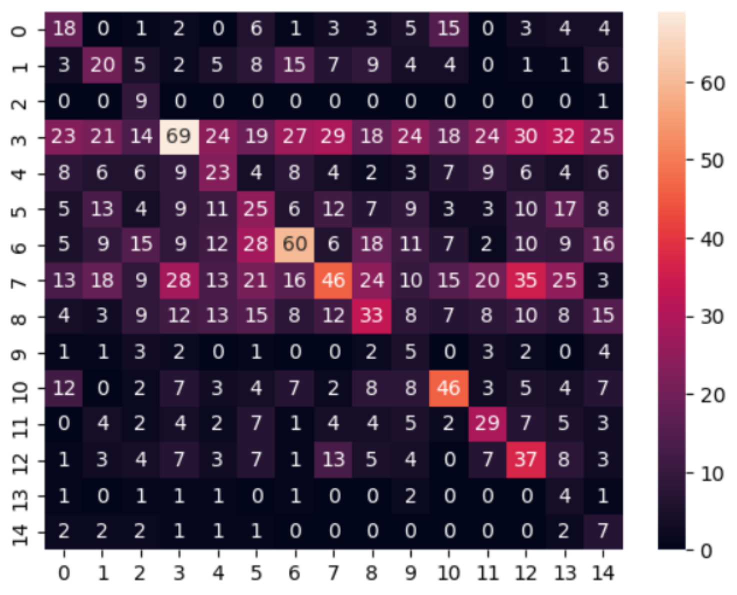
Graph for VIT feature extractor and k = 42



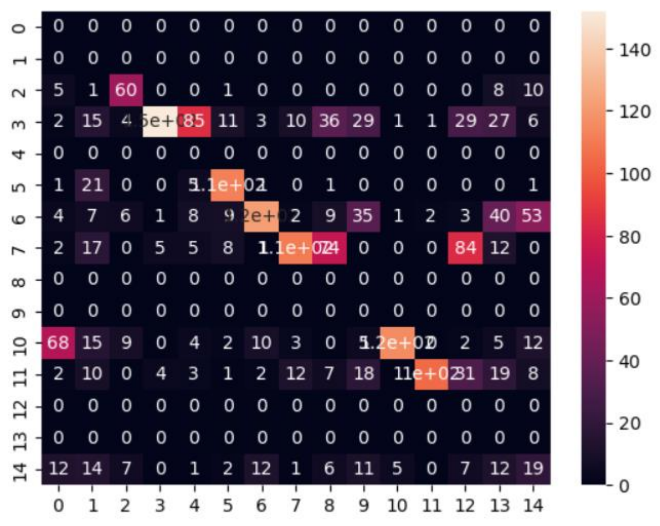
Model	Accuracy	Precission	Recall	F1-Score
HOG + Kmeans + k = 42	0.2189	0.21	0.23	0.20
Vit + KMeans+ k = 42	0.689	0.68	0.74	0.69

Confusion Matrices

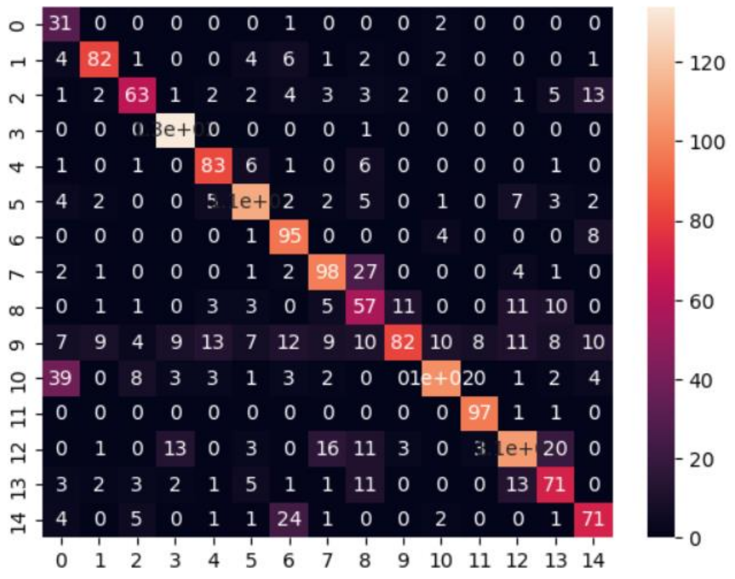
HOG + AgglomerativeClustering best model



VIT + KMedoids best model



VIT + Kmeans best model


0
20
40
60
80
100
120