Flowers clustering

# Problem spec

Given a dataset consisting of images of flowers, the data should be clustered using unsupervised learning. In this particular case, the actual labels for the flowers species exist, therefore the clustering results will be compared with the results of a supervised classification.

# Data

The labels are given in standard csv format where a row consists of a file name identifying the image and the label. The images are RGB and (mostly) of size 128x128 (the few examples of different shapes have been resized to 128x128). Examples:



The labels for the images above are, in order: 0, 0, 1 (denoting species phlox, phlox, rose).

Note: This dataset is currently expanding and offers two versions, a standard one of 210 images, or an extended one of 382 (containing the standard ones). The extended version has been used.

# Data representation

Features have been extracted from the images using a VGG16 model pretrained on ImageNet. This reduced the number of features from 49 512 to 8192. To reduce this number even further, PCA has been applied to the data and this resulted in 256 relevant features (with 95% variance explaining). After this preprocessing, TSNE has been used for visualization. This visualization shows only around 5-6 visible clusters, and this number of clusters was observed in the results as well. Therefore, the data doesn’t seem to be that easily separable into the 10 classes the set provides.

Note: data scaling was attempted before applying PCA, to bring the values between 0 and 1, however it has proven this generated worse results than using the features without scaling.

Chart, scatter chart

Description automatically generated

TSNE Visualization

# Models

After this preprocessing the data has been clustered using the KMeans, Spectral Clustering DBSCAN algorithms. Given that this dataset provided the ground truth labels, a confusion matrix and classification report is available for verifying the results. A comparation with supervised models is also available. For this approach, the features extracted by the PCA were fed to a neural network as one approach and as another, the raw data extracted by the VGG was fed to the same network for comparison. For setting a baseline a dummy random uniform classifier was used.

The models were tuned for hyperparameters, as much as possible, given the number of clusters for KMeans had to be set at 10, as we know the real labels. However, different values were tried for minimum number of samples and epsilon distance threshold in DBSCAN.

Chart, line chart

Description automatically generated

Sum of squared errors for Kmeans by Number of clusters

Visually, the “elbow” method seems to confirm our hunch that KMeans would best split the data into about 6 clusters.

Chart, line chart

Description automatically generated

DBSCAN number of clusters by epsilon (for min samples=5)

A picture containing shape

Description automatically generated

DBSCAN number of clusters by min\_samples(for eps=1000)

Chart, line chart

Description automatically generated

DBSCAN number of clusters by epsilon (for min\_samples=1)

# Results

Based on the hyperparameter search above, epsilon has been set to 1020 and minimum number of samples to 1. This way DBSCAN detects 11 clusters, but even so, it classifies almost all data into cluster 0 and all other clusters 1-10 only have one item. This effect happens even if the parameters are tweaked to change the number of clusters. Any other values would produce results in which all examples are labeled as noise. Therefore, DBSCAN results are not considered relevant.

The dummy random uniform classifiers obtains an accuracy score of 10%, as expected given there are 10 classes.

The neural network – having 3 Dense layers of sizes 256, 128, 10 – when applied to the VGG extracted features obtains an accuracy score of ~52%, while when applied to the same data, but only the features selected by PCA it obtains an accuracy score of ~70%.

The KMeans clustering obtains an accuracy score of ~37%. This is obviously better than the random classifier, but not quite close to a supervised method.

A picture containing text, remote, black

Description automatically generated

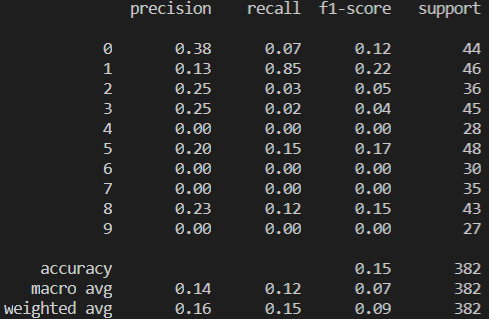
Classification report - KMEans

A picture containing text, electronics, keyboard

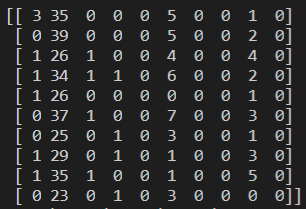
Description automatically generated

Confusion matrix – KMEans

Spectral clustering also succeeded in creating some clusters, but accuracy stands poorly at around ~15%.



Classification report - Spectral clustering



Confusion matrix - Spectral clustering

# Some cluster examples

## Cluster 0



## Cluster 1

Calendar

Description automatically generated

## Cluster 2

Graphical user interface, chart

Description automatically generated