تمرین سوم
ریحانه اسماعیلی زاده
هوش مصنوعی-دکتر فدایی
۸۱۰۸۰۰۰۰۴

First step)

This is a VGG16 structure which is a cnn model. Here we only use the feature extraction part. It also worths noting that the model in keras library has been trained on nearly one million data and the weights are already tuned.

```
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPool2D , Flatten
model = Sequential()
model.add(Conv2D(input_shape=(224,224,3),filters=64,kernel_size=(3,3),padd
ing="same", activation="relu"))
model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same",
activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=128, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=256, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(Conv2D(filters=512, kernel_size=(3,3), padding="same",
activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Flatten())
model.summary()
```

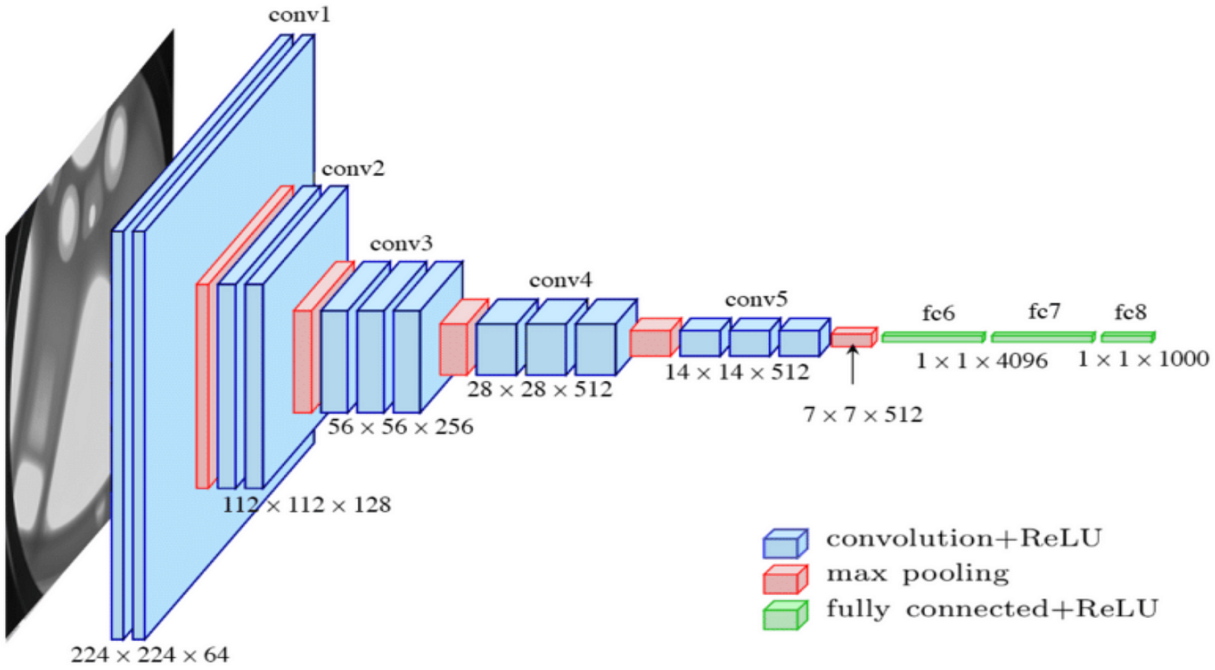
تمرین سوم
ریحانه اسماعیلی زاده
هوش مصنوعی-دکتر فدایی
۸۱۰۸۰۰۰۰۴

We now load the images and true labels

1) Reading the pixels of an image directly without passing it through a model like VGG16 can be useful in certain scenarios, but it also has limitations:

   Limited Feature Extraction: Reading pixel values directly does not extract high-level features from the image. Features extracted by models like VGG16 are learned from vast amounts of data and can capture complex patterns and relationships in images.
   Semantic Understanding: Models like VGG16 are pre-trained on large datasets such as ImageNet, which allows them to understand the semantic content of images. This means they can recognize objects, shapes, textures, and other visual elements, which may not be possible by simply reading pixel values.
   Generalization: Pre-trained models like VGG16 have learned representations that generalize well across different tasks and datasets. They can capture abstract features that are useful for a wide range of image-related tasks, such as classification, object detection, and image generation.
   Dimensionality Reduction: Features extracted by models like VGG16 typically have lower dimensionality compared to raw pixel values. This can lead to more efficient processing and better performance in downstream tasks.

تمرین سوم
ریحانه اسماعیلی زاده
هوش مصنوعی-دکتر فدایی
۸۱۰۸۰۰۰۰۴



VGG16 Illustration

2) Histogram of Oriented Gradients (HOG):
Technique: HOG computes the distribution of gradient orientations in localized portions of an image. It divides the image into small cells, computes gradient orientations within each cell, and then generates a histogram of gradient orientations.
Explanation: HOG captures the local shape and texture information in an image by quantifying the intensity gradients. It is commonly used in object detection and pedestrian detection tasks.
Applications: Object detection, pedestrian detection, human pose estimation.

Local Binary Patterns (LBP):
Technique: LBP encodes the local texture patterns of an image by comparing the intensity of a central pixel with its neighboring pixels. It generates a binary pattern for each pixel based on whether its intensity is greater or less than the intensity of the central pixel.

Explanation: LBP captures the texture information of an image by analyzing the local contrast patterns. It is robust to changes in illumination and provides discriminative features for texture classification tasks.
Applications: Texture classification, face recognition, image retrieval.

Scale-Invariant Feature Transform (SIFT):
Technique: SIFT detects and describes local feature points in an image that are invariant to scale, rotation, and illumination changes. It identifies keypoints based on the presence of local extrema in scale-space and then computes descriptors around each keypoint.
Explanation: SIFT extracts distinctive features from images that are robust to various transformations. It is widely used in image matching, object recognition, and panoramic image stitching.

3) We normalize the feature vectors which have been exracted out of the VGG16

```
scaler = StandardScaler()
feature_vectors = scaler.fit_transform(feature_vectors)
```

4) K-means:

Algorithm: K-means is a partitioning clustering algorithm that aims to partition n data points into k clusters. It iteratively assigns each data point to the nearest cluster centroid and then updates the centroids based on the mean of all points assigned to the cluster.
Brief Explanation: Initially, K cluster centroids are randomly chosen. Then, each data point is assigned to the nearest centroid, forming k clusters. The centroids are updated by recalculating the mean of the data points in each cluster. This process is repeated until convergence, where centroids no longer change significantly or after a predefined number of iterations.
Use Case: K-means is widely used for clustering tasks when the number of clusters is known or can be estimated, and the clusters are globular in shape.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise):
Algorithm: DBSCAN is a density-based clustering algorithm that groups together closely packed points based on two parameters: epsilon (ε), which defines the radius of neighborhood around a point, and minPts, which specifies the minimum number of points required to form a dense region (core point).
Brief Explanation: DBSCAN starts with an arbitrary point and identifies all points within its ε-neighborhood. If the number of points in the neighborhood exceeds minPts, it forms a dense region (core point) and expands the cluster by recursively adding neighboring points. Points that are not core points but are within the ε-neighborhood of a core point are considered part of the cluster (border points). Points that do not belong to any cluster are considered outliers (noise points).
Use Case: DBSCAN is useful for discovering clusters of arbitrary shapes and handling noise in the data. It does not require the number of clusters to be specified in advance and is robust to outliers.

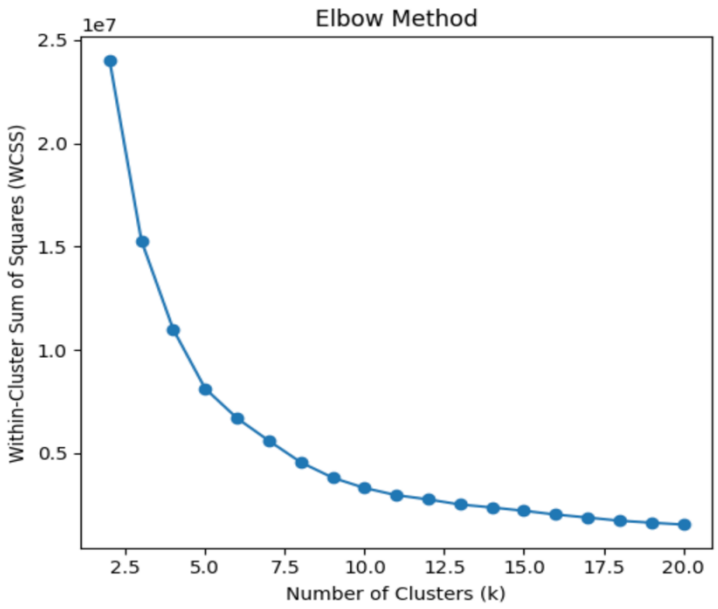
5) Run K-means for different values of k: First, you run the K-means algorithm for a range of k values, typically starting from 1 up to a certain maximum number of clusters. For each value of k, the algorithm clusters the data points into k clusters.

   Calculate the Within-Cluster Sum of Squares (WCSS): For each value of k, calculate the sum of squared distances between each data point and its corresponding centroid within the cluster. This is also known as the Within-Cluster Sum of Squares (WCSS). Mathematically, WCSS is the sum of squared distances from each point to its assigned centroid, summed over all clusters.

   Plot the WCSS against the number of clusters (k): Plot a line graph where the x-axis represents the number of clusters (k) and the y-axis represents the corresponding WCSS value for each value of k.

   Identify the "elbow" point: The "elbow" point is the point on the plot where the rate of decrease in WCSS starts to slow down significantly. Visually, this point resembles an elbow in the plot. The idea is that this point represents

the optimal number of clusters, as adding more clusters beyond this point may not significantly reduce the WCSS.



of

of

Select the optimal k: Based on the plot, you select the value k corresponding to the elbow point as the optimal number clusters for your data.

Here the k=10 is the optimal answer because after that the WCSS differences are negligible. In other words the derivate of the elbow function starts to become zero after the point k = 10.

6) The feature vector size is so large that the visualization is impossible, instead we resort to the homogeneity scores for two different approaches:

تمرین سوم
ریحانه اسماعیلی زاده
هوش مصنوعی-دکتر فدایی
۸۱۰۸۰۰۰۰۴

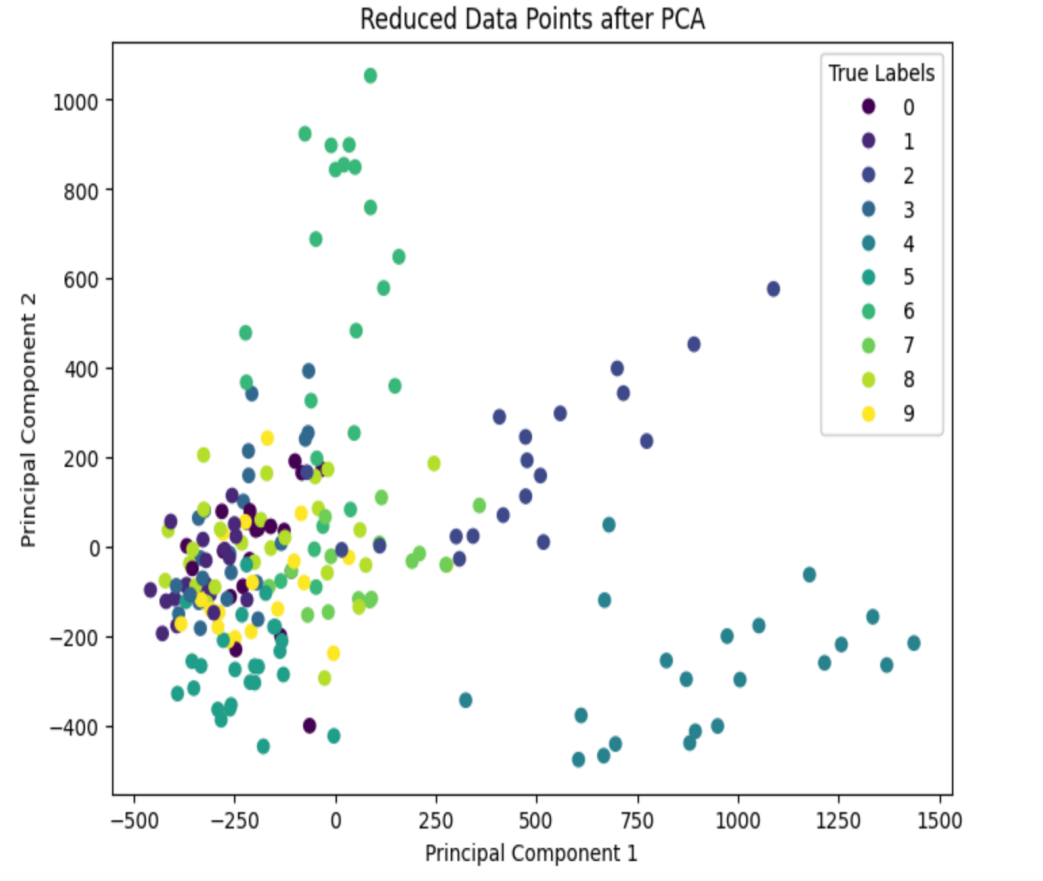
Homogeneity Score for kmeans: 0.515470587075723

Homogeneity Score for DBSCAN: 0.0

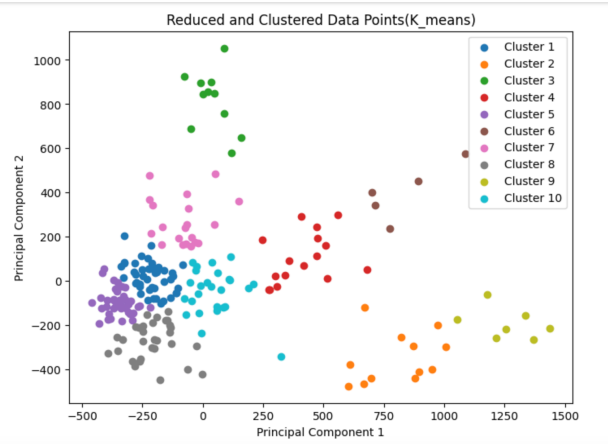
As the space is so large, DBSCAN algorithm perform poorly on clustering, However, we will see improvement using PCA in the upcoming section.

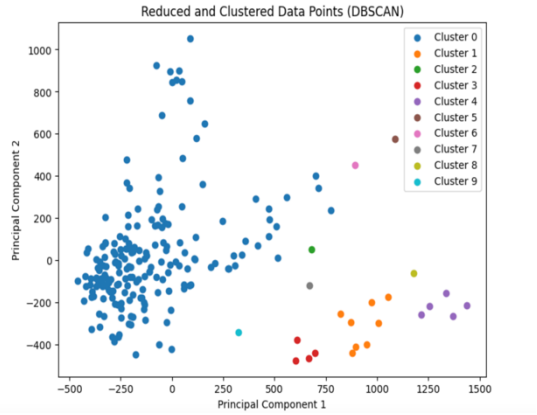
7) Dimensionality Reduction: PCA is primarily used for reducing the dimensionality of a dataset while preserving most of its variance. It transforms the original high-dimensional dataset into a new lower-dimensional space, called the principal components, where each dimension (principal component) is a linear combination of the original features.
   Maximize Variance: PCA aims to find the directions (principal components) along which the variance of the data is maximized. The first principal component captures the most variance in the data, the second principal component captures the second most variance, and so on.
   Orthogonal Transformation: PCA achieves dimensionality reduction through an orthogonal linear transformation. The principal components are orthogonal to each other, meaning they are linearly independent and uncorrelated.
   Eigenvalue Decomposition or Singular Value Decomposition (SVD): PCA can be implemented using eigenvalue decomposition of the covariance matrix of the data or singular value decomposition (SVD) of the data matrix. These methods provide the principal components and corresponding eigenvalues or singular values, which indicate the amount of variance captured by each principal component.
   Variance Retention: PCA allows you to select the number of principal components to retain based on the amount of variance you want to preserve in the data. By choosing fewer principal components, you can achieve dimensionality reduction while retaining most of the variance in the original dataset.
   Applications: PCA is widely used for various purposes, including data visualization, noise reduction, feature extraction, and data compression. It's

commonly applied in fields such as image processing, signal processing, genetics, and finance.



تمرین سوم
ریحانه اسماعیلی زاده
هوش مصنوعی-دکتر فدایی
۸۱۰۸۰۰۰۰۴





As we can see, DBSCAN has clustered the data points based on how much data points are close to each other. In contrast, KMEANS linearly separate the datapoint based on the iteratively updating average and variance of each cluster.

8) Silhouette Score:

The silhouette score measures how similar an object is to its own cluster (cohesion) compared to other clusters (separation).

For each sample:

Compute the average distance from the sample to all other points in the same cluster (a).

Compute the average distance from the sample to all points in the nearest neighboring cluster (b).

The silhouette score for the sample is then calculated as (b - a) / max(a, b).

The silhouette score ranges from -1 to 1:

A score close to +1 indicates that the sample is far away from neighboring clusters.

A score of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters.

A score close to -1 indicates that the sample may have been assigned to the wrong cluster.

Homogeneity Score:

The homogeneity score measures the homogeneity of the clusters, meaning the extent to which each cluster contains only samples from a single class.
It is computed based on the knowledge of the true labels of the data (ground truth).
The homogeneity score ranges from 0 to 1:
A score of 1 indicates perfect homogeneity, meaning each cluster contains only samples from a single class.
A score close to 0 indicates poor homogeneity, meaning clusters contain samples from multiple classes.

9)

DBSCAN:

```
Homogeneity Score: 0.151617594401874

Silhouette Score: 0.28693086
```

KMEASN:

```
Homogeneity Score: 0.5149077079343679

Silhouette Score: 0.376237
```

As can be seen, applying PCA has affected the final results and the scores have gotten better compared to the time when the feature vectors were not reduced. Why is that?

Curse of Dimensionality: As the dimensionality of the feature space increases, the density of data points in that space decreases exponentially. This can lead to sparsity, making it difficult for clustering algorithms to find meaningful clusters.
Increased Computational Complexity: High-dimensional data requires more computational resources and time to process. K-means and DBSCAN have to calculate distances between data points, and with high-dimensional data, the number of distance calculations increases significantly.
Noise Sensitivity: High-dimensional data tends to have a higher proportion of irrelevant or noisy features, which can negatively impact the clustering performance of algorithms like K-means and DBSCAN.

Inefficiency in Distance-Based Metrics: Both K-means and DBSCAN rely on distance-based metrics to define clusters. In high-dimensional spaces, the notion of distance becomes less meaningful, as data points can be arbitrarily far apart or close together in high-dimensional space due to the curse of dimensionality.

10) Feature Selection or Dimensionality Reduction: Reduce the dimensionality of the feature space using techniques like PCA (Principal Component Analysis) or feature selection methods to eliminate irrelevant or redundant features. This can help improve the clustering performance by reducing the curse of dimensionality and noise.
Normalization or Standardization: Scale the features to have similar ranges or distributions, especially if the features have different units or scales. Normalization or standardization can help algorithms like K-means converge faster and produce more meaningful clusters.
Optimizing Hyperparameters: Tune the hyperparameters of DBSCAN and K-means algorithms, such as the epsilon (ε) and minimum samples (minPts) parameters for DBSCAN, and the number of clusters (k) for K-means. Grid search or cross-validation can be used to find the optimal values for these parameters.
Choosing Suitable Distance Metrics: Select appropriate distance metrics based on the characteristics of the data. For example, Euclidean distance is commonly used, but other metrics like Manhattan distance or cosine similarity may be more suitable for certain types of data.
Handling Outliers: Outliers can significantly affect the performance of clustering algorithms. Consider removing or identifying outliers before clustering, or use outlier detection techniques to robustly handle them during clustering.