K-means Clustering on Fruit Dataset

Applying k-means clustering to fruit dataset (k=10). Analysing the clusters and common properties found for each cluster.

Tarushi Kapoor
Machine Learning Engineer Intern
AI Tech Systems
www.ai-techsystems.com
Delhi, India
tarushikapoor.24@gmail.com

***Abstract* — In most real-world scenarios, data-scientists have to do analysis on data which is unlabelled. The K-means clustering algorithm attempts to split a given anonymous data set (a set containing no information as to class identity i.e. unsupervised) into a fixed number (*k*) of clusters (here, *k*=10). For this project, the dataset of 118 fruits and vegetables has been taken from kaggle.com. The objective was to find 10 clusters/groups in the given dataset, analyse them and find their common properties. The images were first trained on VGG16 Neural Network. Then the dimensions of the images were reduced using Principle Component Analysis (PCA) to visualise the clusters in 2-D.**

***Keywords* — Unsupervised learning, K-means, VGG16, PCA**

I. INTRODUCTION

Clustering finds patterns in data—whether they are there or not. It is a type of unsupervised learning method in which we draw references from datasets consisting of input data without labeled responses. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. K-means algorithm is an iterative algorithm that tries to partition the dataset into *k* pre-defined distinct non-overlapping clusters where each data point belongs to only one group. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as far as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous the data points are within the same cluster.

Here, we will perform clustering on the given dataset. Images are generally known as data points in Deep Learning and can considered same as the data points in Machine Learning. In this paper, we will have a set of fruit images. We will group them into 10 different clusters. For this, we will extract image features from a pre-trained Keras model —VGG16. Once we have the vectors, we will use PCA to reduce the dimensions of the data and then apply K-Means clustering over the data points. Given are the different types of fruits and vegetables in out dataset.

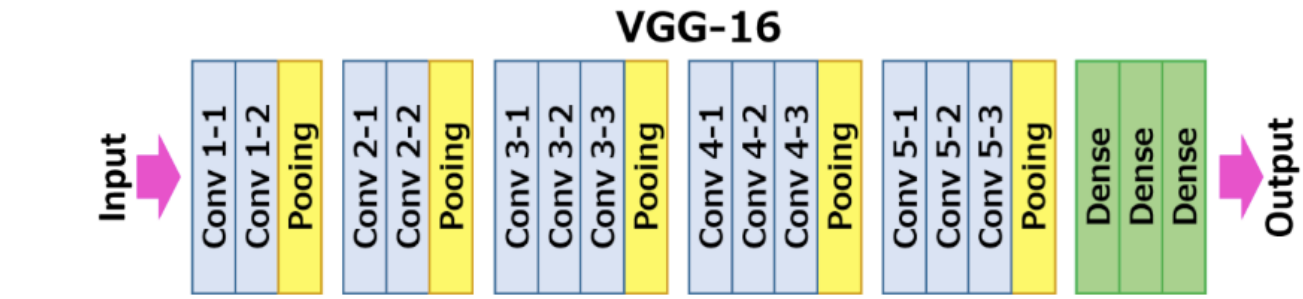
II. DATA PRE-PROCESSING

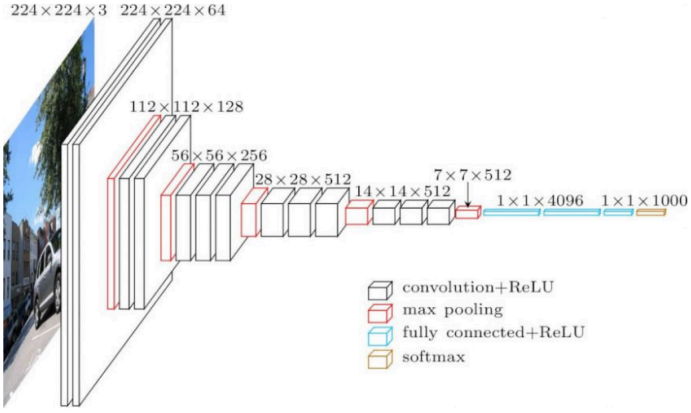
The dataset from kaggle.com consist of 59,328 images of fruits in the Training folder. These images are divided into 118 classes of fruits and each class contains approximately 400 images. Shape of every image is 100x100x3 pixels with 3 colour channels (RGB).

A. VGG16 : A pre-trained model from Keras to extract features from given images

VGG16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman. The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It makes the improvement over AlexNet by replacing large kernel-sized filters with multiple 3×3 kernel-sized filters one after another.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					



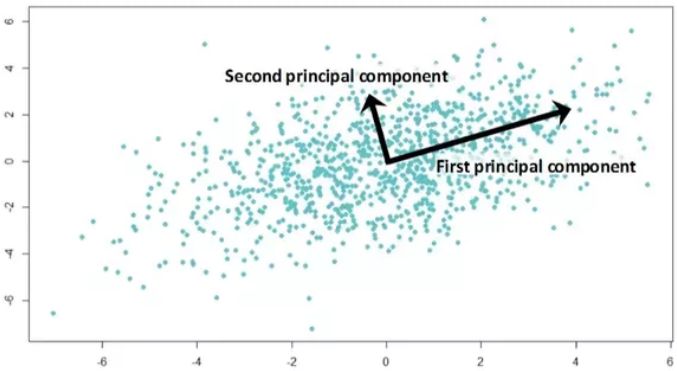


VGG16 Architecture

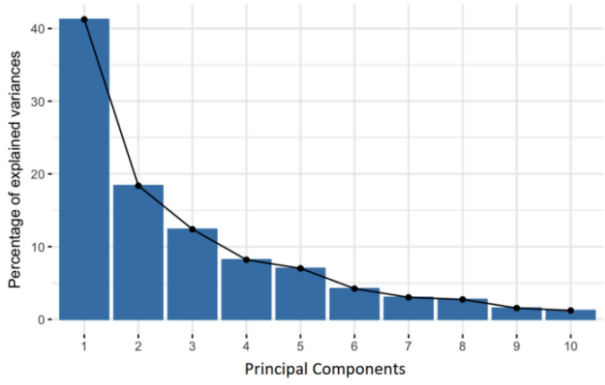
The input to conv1 layer is of fixed size 224 x 224 RGB image. The image is passed through a stack of convolutional layers, where the filters is used with a very small receptive field: 3×3. The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e., the padding is 1-pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers. Max-pooling is performed over a 2×2 pixel window, with stride 2. Three fully connected (FC) layers follow a stack of convolutional layers.

B. Principle Component Analysis (PCA)

Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualise and make analysing data much easier and faster for machine learning algorithms without extraneous variables to process. So to sum up, the idea of PCA is simple — reduce the number of variables of a data set, while preserving as much information as possible.



Principal components are new variables that are constructed as linear combinations or mixtures of the initial variables. These combinations are done in such a way that the new variables (i.e., principal components) are uncorrelated and most of the information within the initial variables is squeezed or compressed into the first components.



Percentage of variance (information) for by each PC

III. K-MEANS CLUSTERING ALGORITHM

Clustering is a method to divide a set of data into a specific number of groups. It's one of the popular method is k-means clustering. In k-means clustering, it partitions a collection of data into a k number group of data. It classifies a given set of data into k number of disjoint cluster. K-means algorithm consists of two separate phases. In the first phase it calculates the k centroid and in the second phase it takes each point to the cluster which has nearest centroid from the respective data point. There are different methods to define the distance of the nearest centroid and one of the most used methods is Euclidean distance. Once the grouping is done it recalculate the new centroid of each cluster and based on that centroid, a new Euclidean distance is calculated between each center and each data point and assigns the points in the cluster which have minimum Euclidean distance. Each cluster in the partition is defined by its member objects and by its centroid. The centroid for each cluster is the point to which the sum of distances from all the objects in that cluster is minimised. So K-means is an iterative algorithm in which it minimises the sum of distances from each object to its cluster centroid, over all clusters.

A. *Data Assignment Step*

The first thing k-means does, is randomly choose K examples (data points) from the dataset as initial centroids and that's simply because it does not know yet where the center of each cluster is. Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if C*i* is the collection of centroids in set *C*, then each data point *x* is assigned to a cluster based on

$$\underset{c_i \in C}{\arg\min}\ dist(c_i, x)^2$$

where *dist*(·) is the standard (*L*2) Euclidean distance. Let the set of data point assignments for each *ith* cluster centroid be *Si*.

B. *Centroid Update Step*

In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

$$c_i = \frac{1}{|S_i|}\sum_{x_i \in S_i} x_i$$

The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimised, or some maximum number of iterations is reached).

This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e., not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomised starting centroids may give a better outcome.

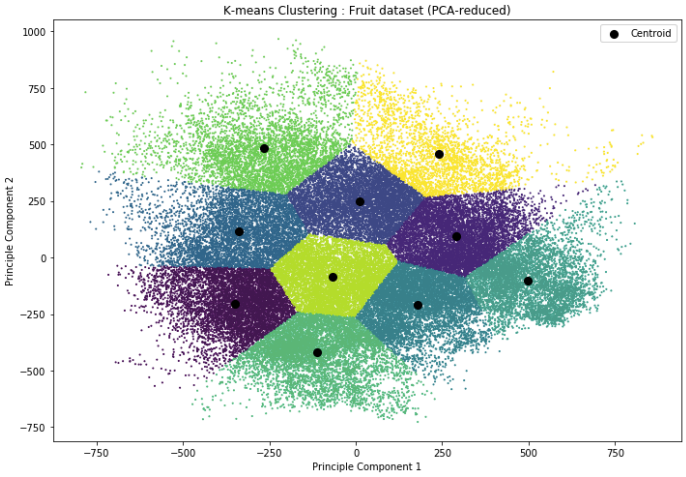
C. *Cluster Properties*

One of the clusters formed is shown below:

```
the number of images in 0 cluster is = 6443
the number of images in 1 cluster is = 7337
the number of images in 2 cluster is = 7265
the number of images in 3 cluster is = 5738
the number of images in 4 cluster is = 6909
the number of images in 5 cluster is = 5051
the number of images in 6 cluster is = 6469
the number of images in 7 cluster is = 4572
the number of images in 8 cluster is = 6857
the number of images in 9 cluster is = 2687
```

IV. CONCLUSION

Dimension is reduced to two using PCA and then k-means is applied to the reduced dataset. All the images are clustered in the 10 clusters on the basis of their shape and colour. Each cluster contains images of fruit of specific shape only. Graph is plotted using two reduced features named Principle Component 1 and Principle Component 2. Black dot represents the centroids.



REFERENCES

1. https://www.kaggle.com/moltean/fruits
2. https://www.geeksforgeeks.org/clustering-in-machine-learning/
3. https://www.datascience.com/blog/k-means-clustering
4. https://neurohive.io/en/popular-networks/vgg16/
5. https://towardsdatascience.com/a-step-by-step-explanation-of-principal-component-analysis-b836fb9c97e2