Fruit Classifier

Stefano Nicolis, Pietro Turco

A.Y. 2022-2023

17 settembre 2023

Università degli Studi di Verona **Computer Science Department**

Indice

1 Motivation and Rationale

The purpose of this project is to apply a subset of the various machine learning techniques learned during the ML and AI course to solve a particular problem and see them in practice.

It was chosen to create an image based fruit classifier.

This problem was chosen because it is simple enough to be approachable for beginners in this field but not simple enough not to pose a challenge.

2 State of the art

2.1 Today

Fruit recognition plays an important role in modern agriculture and is used for various tasks:

- 1.**Quality Control**: Fruit recognition allows for automated sorting of fruits based on their size, color, and other physical attributes. This ensures that only the highest quality fruits reach the consumers, enhancing customer satisfaction.
- 2.**Disease Detection**: Early detection of diseases in fruits can prevent widespread damage in orchards. Fruit recognition technology can identify disease symptoms that might be invisible to the naked eye.
- 3.**Inventory Management**: In retail settings, fruit recognition can help keep track of inventory and make the supply chain more efficient. It can also aid in automatic billing systems.
- 4.**Agricultural Planning**: By recognizing and classifying fruits, farmers can make informed decisions about harvesting and marketing their produce.
- 5.**Research and Development**: Fruit recognition plays a crucial role in botanical research, aiding in the classification and study of various fruit species.

2.2 Challenges

Like all image-based classifiers, a fruit classifier struggles with noise in the dataset: bad lightning, leaves occluding the fruit etc can greatly reduce the accuracy of such classifier. Not to consider the great variety of shapes, colors and textures that fruits of the same kind can have. For example, a ripe banana and an non-ripe banana have different colors, and a small pumpkin might look similar to an orange. Variety of the same fruit can also pose a challenge, for example a Beefsteak tomato is very different from a Cherry tomato.

3 Objectives

The objective of this project is to find an appropriate data set and classify all the sample in it. We chose the goal to classify fruits using different algorithms and methods. We thought to use both machine learning and deep learning method in order to show different kind of approaches for the same purpose.

4 Methodology

4.1 Dataset

The dataset used is a subset of a big fruit image dataset found on Kaggle. After some reduction it ended up with 25 classes, 400 images each, 40x40 RGB.

A 80%-20% split has been used divide the dataset into train and test data, respectively.

The discriminant features are the color, the shape and the texture of the fruit. In fact we have a lot of similarly shaped fruit, which only differ by colour and texture. For example, an orange and an apple have the same overall shape, but different color and texture.

A *.csv* file is used to store the features (pixels) and label of each image. The presence of the label means supervised learning methods will be used.

	A	B	C	D	E	F	G	H
1	label	0_R	0_G	0_B	1_R	1_G	1_B	2_R
2	0	244	255	246	245	255	246	248
3	0	244	255	249	244	255	249	246
4	0	246	255	249	246	255	249	247
5	0	246	255	252	246	255	252	247
6	0	246	255	252	246	255	252	247
7	0	243	255	252	243	255	252	246
8	0	243	255	252	243	255	252	246
9	0	246	255	252	246	255	252	247
10	0	243	255	252	243	255	252	246
11	0	243	255	252	243	255	252	246
12	0	243	255	252	243	255	252	246
13	0	245	255	255	248	255	255	252
14	0	246	255	252	246	255	252	247
15	0	243	255	252	243	255	252	246
16	0	243	255	249	243	255	249	246
17	0	243	255	252	243	255	252	246

4.2 Algorithms used

The image classifier has been implemented with 3 different methods:

- **K-NN** (K-Nearest Neighbors): It is a simple and widely used algorithm in machine learning for classification and regression tasks.

- **SVM** (Support Vector Machines): It is a machine learning model used for classification and regression. SVM tries to find a hyperplane in an N-dimensional space that clearly classifies the data points.

- **CNN** (Convolutional Neural Network): It is a type of deep neural network architecture specifically designed for processing and analyzing grid-structured data, such as images and videos. CNNs are widely used in artificial vision tasks, including image classification, object detection, image segmentation, and more. The key feature of CNNs is their ability to automatically learn hierarchical models and features from input data.

  The same three methods were also tested with a reduced feature space thanks to PCA.

- **PCA** (Principal Component Analysis): Also known as the Karhunen-Loève transform, it is a dimensionality reduction method used on the input features to make the problem more computationally manageable.

The effectiveness of **SVM** and **K-NN** was verified by looking at the the following statistics:

- **Accuracy**: measure of the overall correctness of the model. It is the ratio of the number of correct predictions to the total number of predictions. In other words, it's how many predictions are correct out of all predictions.

- **Precision**: measure of the correctness of the model's positive predictions. It is the ratio of the number of true positives (observations labeled as positive by the model that are actually positive) to the total number of observations labeled as positive by the model (true positives + false positives). In other words, it's a measure of how accurate the model is when it says an observation is positive.

- **Recall**: measure of the model's ability to find all possible positive predictions. It is the ratio between the number of true positives and the total number of actual positives (true positives + false negatives). In other words, it's a measure of how well the model can find all the positive observations.

The formulas for these three statistics are given below:

Key Metrics:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

5 Experiments and Results

5.1 SVM

The SVM algorithm is used with different kernels: Linear, Sigmoid, Polynomial and Radial.

SVM			
Kernel	**Accuracy**	**Precision**	**Recall**
Linear	1	1	1
Poly	1	1	1
Rbf	1	1	1
Sigmoid	1	1	1

We notice that SVM with different kernels doesn't change the results and it provides excellent results. That is due to the simplicity of the dataset which takes as main feature color and shape. In fact in our dataset there aren't a lot of similar fruits.

5.2 KNN

KNN is used in our project with following k values: $5, 17, 21, 51, 101$. We chose them in order to have k always lower than $\sqrt{\#\ samples}$. In particular 101 is exactly the next odd value after $\sqrt{10000}$ where 10000 is the number of samples of our dataset.

The Metrics used are Cosine, Manhattan and Euclidean. In our project they work giving almost equals results.

KNN				
Metric	**K**	**Accuracy**	**Precision**	**Recall**
Cosine	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.98	0.99	0.98
	51	0.96	0.97	0.96
	101	0.93	0.93	0.93
Manhattan	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.98	0.99	0.98
	51	0.96	0.97	0.96
	101	0.93	0.93	0.93
Euclidean	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.98	0.99	0.98
	51	0.96	0.97	0.96
	101	0.93	0.93	0.93

We chose odd values for the obvious reason of having always fair results. Even numbers could potentially result in a tie between two classes. Note how higher K values yield worse results, that is due to the fact that searching for more points around the one to be classified, will inevitably capture points in other classes.

5.3 CNN

CNNs work very well in image recognition tasks, and indeed we got excellent results in our fruit classifier. As activation function we tried with RELU, ELU and TanH. Epochs were limited to 3 because convergence was reached.

CNN		
Activation func	**Epochs**	**Accuracy**
RELU	3	99.8
ELU	3	100
TanH	3	100

We expected in this case excellent results, because CNN is perfect for tasks like image classification. The sequence of layers used is the following:

- **Conv1**: This is the first convolutional layer. It takes as input a color image with 3 channels (RGB) and applies 32 filters of size 3x3. This layer is responsible for learning low-level features of the image such as edges and gradients.
- **Pool1**: This is the first pooling layer, which reduces the spatial dimensions of the image while retaining the most important features. It uses a kernel of size 2x2.
- **Conv2**: This is the second convolutional layer. It takes as input the features learned from the first convolutional layer and applies 64 filters of size 3x3. This layer learns higher-level features based on the low-level features learned from the first convolutional layer.
- **Pool2**: This is the second pooling layer, which further reduces the spatial dimensions of the image.
- **Flatten**: This layer flattens the output of the second pooling layer into a one-dimensional vector. This is necessary because fully connected (Fc) layers require one-dimensional input.
- **Fc1**: This is the first fully connected (Fc) layer. It takes as input the flattened vector and transforms it into a vector of size 128.
- **Fc2**: This is the second (and final) fully connected layer. It takes as input the output from the first FC layer and transforms it into the final number of classes, which represent probabilities for each class.

5.4 PCA

PCA (Principal Component Analysis)is a method of dimensionality reduction often used in machine learning. We tested the variance of the dataset under five different component values:

```
Start test PCA
Dataset variance, PCA = 7: 0.714373342981796
Dataset variance, PCA = 21: 0.8579296580496243
Dataset variance, PCA = 70: 0.9377112703284214
Dataset variance, PCA = 100: 0.9525678491892725
Dataset variance, PCA = 700: 0.9930366287308121
End test PCA
```

The high value of representation means that the principal components capture a large amount of the variability in the data. In other words, they represent a large portion of the information contained in the original features.

5.5 SVM with PCA

With PCA = 100, SVM maintained very good results.

SVM, PCA = 100			
Kernel	Accuracy	Precision	Recall
Linear	1	1	1
Poly	0.99	0.99	0.99
Rbf	1	1	1
Sigmoid	0.94	0.94	0.94

5.6 KNN with PCA

With PCA = 100, KNN maintained very good results. We notice with 101 a little decrement of accuracy with Cosine and Manhattan, but a little increase for Euclidean.

KNN, PCA=100				
Metric	**K**	**Accuracy**	**Precision**	**Recall**
Cosine	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.99	0.99	0.99
	51	0.95	0.95	0.95
	101	0.91	0.91	0.91
Manhattan	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.99	0.99	0.99
	51	0.96	0.97	0.97
	101	0.91	0.93	0.91
Euclidean	5	0.99	0.99	0.99
	17	0.99	0.99	0.99
	21	0.99	0.99	0.99
	51	0.97	0.97	0.97
	101	0.94	0.94	0.94

6 Conclusions

Machine learning techniques such as K-Nearest Neighbors (KNN), Principal Component Analysis (PCA), Support Vector Machines (SVM), and Convolutional Neural Networks (CNN) were used to create a classifier capable of accurately classifying a wide range of fruits with a high degree of accuracy.

Fruit classification may seem like a simple task at first glance, but it's actually a complex problem that we facilitated for ourselves by choosing a simple dataset. The simple dataset allowed us to focus on learning the actual machine learning algorithms and workflow needed in these scenarios.

7 References

https://ieeexplore.ieee.org/document/8442331/

$https : //link.springer.com/content/pdf/10.1007/978 - 0 - 387 - 77251 - 6_18.pdf$

https://viso.ai/applications/computer-vision-in-agriculture/

https://www.mdpi.com/1424-8220/16/8/1222

https://arxiv.org/abs/1712.00580

https://link.springer.com/article/10.1007/s11042-022-12868-2