

Fruit recognition from images using deep learning

Definition

Project Overview

This project comes from the following Kaggle Fruits dataset : <https://www.kaggle.com/moltean/fruits>

There are an increasing number of robotics applications aimed at detecting fruits from images or videos. Although various research efforts have been made in this field, challenges still remain for complex scenes with varying lighting conditions, low contrast between fruits and leaves, foreground occlusions and cluttered backgrounds. Most of these applications have been to find the fruits for automatic harvesting. A recently new direction is to find the fruits for plant breeding purposes to automatically recognize, count and measure the fruits in order to assess the differences in quality of the genetic material. When the measurements are made by a computer, this is often referred to as digital phenotype and the field is growing in importance.

Fruits have certain categories that are hard to differentiate, like the citrus genus, that contains oranges and grapefruits.

Historical information relevant to the project is that this project started with small dataset about five hundred images but now it become large about 50000 images, it also implemented by ACTA UNIV. SAPIENTIAE , INFORMATICA.

The project that has the target of obtaining a classifier that can identify a much wider array of objects from images. This fits the current trend of companies working in the augmented reality field. During its annual I/O

conference, Google announced that is working on an application named Google Lens which will tell the user many useful information about the object toward which the phone camera is pointing. First step in creating such application is to correctly identify the objects. The software has been released later in 2017 as a feature of Google Assistant and Google Photos apps. Currently the identification of objects is based on a deep neural network [[link above](#)].

Such a network would have numerous applications across multiple domains like autonomous navigation, modeling objects, controlling processes or human-robot interactions. The area we are most interested in is creating an autonomous robot that can perform more complex tasks than a regular industrial robot. An example of this is a robot that can perform inspections on the aisles of stores in order to identify out of place items or understocked shelves. Furthermore, this robot could be enhanced to be able to interact with the products so that it can solve the problems on its own. An other area in which this research can provide benefits is autonomous fruit harvesting. While there are several papers on this topic already, from the best of our knowledge, they focus on few species of fruits or vegetables.

Problem Statement

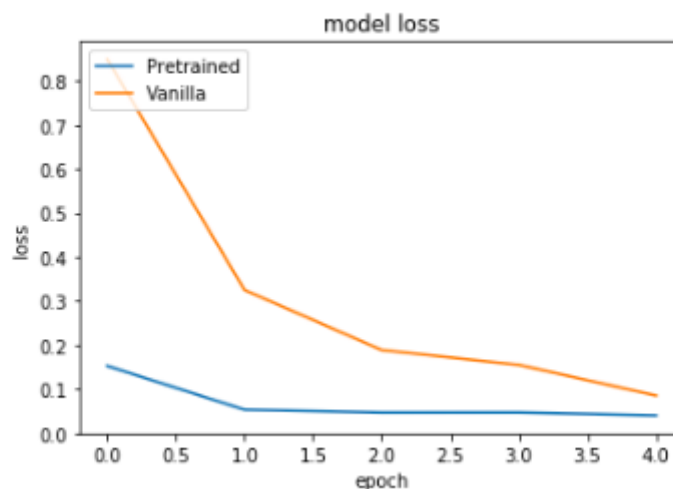
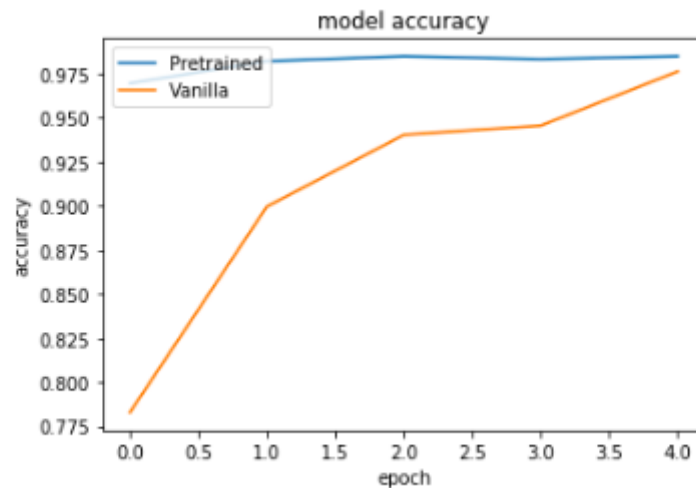
this project aims to classify the types of 81 different fruits by there images using convolution neural network (CNN). Thus we want to see how well can an artificial intelligence complete the task of classifying them. Another reason is that fruits are very often found in stores, so they serve as a good starting point for the previously mentioned project.

the solution is to use deep learning technique by using convolution neural network (CNN). Each level learns to transform its input data into a slightly more abstract and composite representation. deep neural networks specifically convolutional neural networks have been proved to obtain great results in the field of image recognition.

Metrics

This problem is a multi classification problem, Accuracy will be used as the evaluation metric between the solution model and the benchmark model. I depend on the result of the Evaluation metrics

Activity	Benchmark	Solution statement
Size of data	About 50000 images	About 53250 images
Quantify the performance of both	The accuracy reach to over 0.9454 %	Hope to reach the accuracy over 94%
Number of epochs	Use four epochs	Use about five epochs



Analysis

Data Exploration

The dataset has about 53250 images of fruits spread across 81 labels.

The data set is available on GitHub and Kaggle. The dataset is balanced and the project doesn't have any unbalanced dataset because of the equality of the data and each-other all types of data have the same number of images which enter the model to be trained and tested.

the dataset are available and open source on [[Fruit-360](#)] ,i can work on kaggle workbench, it gives 6 hours per session so can work on it and implement the dataset from it. The characteristics of the dataset that every type of the fruits have at least 420 images to train and 160 images to test from all directions and different kind of the same fruit. The dataset is the main component in the project it's divided in two sections train data and test data so their use is appropriate given the context of the problem.

Some historical information about dataset (The images were obtained by filming the fruits while they are rotated by a motor and then extracting frames. Fruits were planted in the shaft of a low speed motor (3 rpm) and a short movie of 20 seconds was recorded. Behind the fruits we placed a white sheet as background.

In the figure above :

Left-side: original image. Notice the background and the motor shaft.

Right-side: the fruit after the background removal and after it was scaled down to 100x100 pixels. The images then scaled them to 100x100 pixels images.



The labels and the number of images for training are given in Table1.

Table 1: Number of images for each fruit. There are multiple varieties of apples each of them being considered as a separate object. We did not find the scientific/popular name for each apple so we labeled with digits (e.g. apple red 1, apple red 2 etc).

Label	Number of training images	Number of test images
Avocado	427	143
Avocado ripe	491	166
Banana	490	166
Banana Red	490	166
Cactus fruit	490	166
Cantaloupe 1	492	164
Cantaloupe 2	492	164
Carambula	490	166
Cherry 1	492	164
Cherry 2	738	246

Cherry Rainier	738	246
Cherry Wax Black	492	164
Cherry Wax Red	492	164
Cherry Wax Yellow	492	164
Clementine	490	166
Cocos	490	166
Dates	490	166
Granadilla	490	166
Grape Pink	492	164
Grape White	490	166
Grape White 2	490	166
Grapefruit Pink	490	166
Grapefruit White	492	164
Guava	490	166
Huckleberry	490	166
Kaki	490	166
Kiwi	466	56
Kumquats	490	166
Lemon	492	164
Lemon Meyer	490	166
Limes	490	166
Lychee	490	166
Mandarine	490	166
Mango	490	166
Maracuja	490	166
Melon Piel de Sapo	738	246
Mulberry	492	164
Nectarine	492	164
Orange	479	160
Papaya	492	164
Passion Fruit	490	166
Peach	492	164
Peach Flat	492	164

Pear	492	164
Pear Abate	490	166
Pear Monster	490	166
Pear Williams	490	166
Pepino	490	166
Physalis	492	164
Physalis with Husk	492	164
Pineapple	490	166
Pineapple Mini	493	163
Pitahaya Red	490	166
Plum	447	151
Pomegranate	492	164
Quince	490	166
Rambutan	492	164
Raspberry	490	166
Salak	490	162
Strawberry	492	164
Strawberry Wedge	738	246
Tamarillo	490	166
Tangelo	490	166
Walnut	735	249

Exploratory Visualization

The relevant characteristic of the used dataset is all of them present the type of fruits and not any other object in the dataset exact fruits, secondly the feature or pattern in all dataset is that any image can contain only one fruit so it's easy to classify between them, third each type of fruits have a folder with it's images in the training set and test set, each folder of them contain the shape of the fruit from all direction to be completely described.

The third point is the most important on because this act the fruit like a 3D so it make the model be more accurate when detect the type of the fruit. so

if the plot is provided from any folder of the dataset it can clearly define the axis of any image are the same and the title of the fruit clearly.

Algorithms and Techniques

Deep learning

In the area of image recognition and classification, the most successful results were obtained using artificial neural networks. These networks form the basis for most deep learning models. Deep learning is a class of machine learning algorithms that use multiple layers that contain nonlinear processing units. Each level learns to transform its input data into a slightly more abstract and composite representation. Deep neural networks have managed to outperform other machine learning algorithms. They also achieved the first superhuman pattern recognition in certain domains. This is further reinforced by the fact that deep learning is considered as an important step towards obtaining Strong AI. Secondly, deep neural networks - specifically convolutional neural networks - have been proved to obtain great results in the field of image recognition.

Convolutional neural networks

Convolutional neural networks (CNN) are part of the deep learning models. Such a network can be composed of convolutional layers, pooling layers, ReLU layers, fully connected layers and loss layers. In a typical CNN architecture, each convolutional layer is followed by a Rectified Linear Unit (ReLU) layer, then a Pooling layer then one or more convolutional layer and finally one or more fully connected layer. A characteristic that sets apart the CNN from a regular neural network is taking into account the structure of the images while processing them. Note that a regular neural network converts the input in a one dimensional array which makes the trained classifier less sensitive to positional changes. Among the best results obtained on the MNIST dataset is done by using multi-column deep neural networks, they use multiple maps per layer with many layers of non-linear neurons. Even if the complexity of

such networks makes them harder to train, by using graphical processors and special code written for them. The structure of the network uses winner-take-all neurons with max pooling that determine the winner neurons.

Benchmark Model

I plan to compare the results of the CNN model which I implement with the result of the same project on kaggle ([linked to above](#)). I will compare the model accuracy/model loss and number of epochs used each other to see which is more effective, as well as compare the speed of the two techniques (after training, in the case of the CNN).

Methodology

Data preprocessing

The targeted plants are peppers with fruits of complex shapes and varying colors similar to the plant canopy. The aim of the application is to locate and count green and red pepper fruits on large, dense pepper plants growing in a greenhouse. The training and validation data used consists of 28000 images of over 1000 plants and their fruits. The used method to locate and count the peppers is two-step :

in the first step, the fruits are located in a single image and in a second step multiple views are combined to increase the detection rate of the fruits.

The approach to find the pepper fruits in a single image is based on a combination of (1) finding points of interest, (2) applying a complex high dimensional feature descriptor of a patch around the point of interest and (3) using a so-called bag-of-words for classifying the patch.

For this purpose the authors adapt a Faster Region-based convolutional network. The objective is to create a neural network that would be used by autonomous robots that can harvest fruits. The network is trained using RGB and NIR (near infra red) images. The combination of the RGB and

NIR models is done in 2 separate cases: early and late fusion. Early fusion implies that the input layer has 4 channels :

3 for the RGB image and one for the NIR image. Late fusion uses 2 independently trained models that are merged by obtaining predictions from both models and averaging the results. The result is a multi modal network which obtains much better performance than the existing networks.

Implementation

The first of all we import the required libraries to be used, secondly the phase of load the data so write function to load the data from their folders and load them in arrays to be used and separate them to train set and test set then create the CNN model. The model use sequential model of CNN that can be composed of convolutional layers, pooling layers, ReLU layers, fully connected layers and loss layers. CNN architecture, each convolutional layer is followed by a Rectified Linear Unit (ReLU) layer, then a Pooling layer then one or more convolutional layer and finally one or more fully connected layer.

The first layer is consisted of 16 dense the second is 32 and the third is 64, adding **pooling** layers which are used on one hand to reduce the spatial dimensions of the representation and to reduce the amount of computation done in the network. The other use of pooling layers is to control overfitting. The most used pooling layer has filters of size 2 x 2 with a stride 2. This effectively reduces the input to a quarter of its original size.

Fully connected layers are layers from a regular neural network. Each neuron from a fully connected layer is linked to each output of the previous layer. The operations behind a convolutional layer are the same as in a fully connected layer. Thus, it is possible to convert between the two.

Loss layers are used to penalize the network for deviating from the expected output. This is normally the last layer of the network. Various lossfunction exist: softmax is used for predicting a class from multiple

disjunct classes, sigmoid cross-entropy is used for predicting multiple independent probabilities (from the $[0, 1]$ interval). Then compile the model with loss (categorical cross entropy) , optimizer (Adam) and the metrics=['accuracy']. Then fit the model with the number of epochs which equal to 5. Finally get the accuracy of the project.

Refinement

the first solution contained about 20 epochs, when we train it find the accuracy of the solution isn't acceptable it was less than 90% because of the over-fitting son in the final model we reduce the number of epochs to 5 epochs which fit will and get a great accuracy. Another change while compile change the optimizer to adam which give the best fit and accuracy

Results

Model Evaluation and Validation

The final model will be the refined CNN model which performed well on the testing set.

Sensitivity analysis

To validate the robustness of this final model I computed the bottleneck features for the new modification in dataset and started training and testing on them. It got 0.9677 training accuracy score and 0.96771636 testing accuracy score which is suitable.

Justification

The solution model outperforms the benchmark model in accuracy of data with 9677% because the benchmark model accuracy is 9454% .

Conclusion

Free-Form Visualization

The dataset was split in 2 parts: training set - which consists of 37836 images of fruits and testing set - which is made of 12709 images.

The data was bundled into a TFRecords file (specific to TensorFlow).

The explanation above is the modification on dataset which I used in the project :









“ This is a binary file that contains protocol buffers with a feature map. In this map it is possible to store information such as the image height, width, depth and even the raw image. Using these files we can create queues in order to feed the data to the neural network. By calling the method `shuf_fle_batch` we provide randomized input to the network. The way we used this method was providing it example tensors for images and labels and it returned tensors of shape batch size x image dimensions and batch size x labels. This helps greatly lower the chance of using the same batch multiple times for training, which in turn improves the quality of the network.

On each image from the batch we applied some preprocessing in order to augment the data set. The preprocessing consists of randomly altering the hue and saturation, and applying random vertical and horizontal flips. For the hue and saturation we use the TensorFlow methods: `random hue` and `random saturation`. To further improve the accuracy of the network we converted each image from the batch to grayscale and concatenated it to the image. Thus the data that is fed into the network will have the size 100 x 100 x 4.”

In order to be able to detect fruits from images I used the previously¹⁸ described neural network which was trained over 50000 iterations with batches of 50 images selected at random from the training set. Every 50

steps we calculated the accuracy using cross-validation. This showed steady improving of the network until reaching 100% accuracy on cross-validation.

For the testing phase, we used the testing set and the calculated accuracy was 96.771636 % Some of the incorrectly classified images are given blow Some of the images that were classified incorrectly. On the top we have the correct class of the fruit and on the bottom we have the class that was given by the network and its associated probability

Apple Golden 2  Apple Golden 3 96.54%	Apple Golden 3  Granny Smith (Apple) 95.22%	Braeburn(Apple)  Apple Red 2 97.71%	Peach  Apple Red Yellow 97.85%
Pomegranate  Nectarine 94.64%	Peach  Apple Red 1 97.87%	Pear  Apple Golden 2 98.73%	Pomegranate  Braeburn(Apple) 97.21%

Reflection

The entire end-to-end problem solution:

1. Choose the problem and a valid dataset which is already divided among training and testing.
2. Do image preprocessing on the chosen dataset.
3. choose the benchmark model.
4. Extract the bottleneck features from the CNN model.

Improvement

I described a new and complex database of images with fruits. Also I made some numerical experiments by using TensorFlow library in order to classify the images according to their content. From my point of view one of the main objectives for the future is to improve the accuracy of the neural network. This involves further experimenting with the structure of the network. Various tweaks and changes to any layers as well as the introduction of new layers can provide completely different results.

Another option is to replace all layers with convolutional layers. This has been shown to provide some improvement over the networks that have fully connected layers in their structure. A consequence of replacing all layers with convolutional ones is that there will be an increase in the number of parameters for the network. Another possibility is to replace the rectified linear units with exponential linear units. this reduces computational complexity and add significantly better generalization performance than rectified linear units on networks with more that 3 layers. We would like to try out these practices and also to try to find new configurations that provide interesting results.

In the near future i plan to create a mobile application which takes pictures of fruits and labels them accordingly.

Another objective is to expand the data set to include more fruits.