**FINAL**

**REPORT**

**ON**

# Fruit recognition from images using CNN Technique


**Name: Rahul Duggal**

**RegNo: 11716807**

**Roll No: 61**

**Course Code: INT248**

**Programme Name: B.Tech CSE**

**Submitted To: Mr. Sanjay Kumar Singh**

**School of Computer Science & Engineering**

**Lovely Professional University, Phagwara**

**(September, 2020)**

## Abstract

In this paper we introduce a new, high-quality, dataset of images containing fruits. We also present the results of some numerical experiment for training a neural network to detect fruits. We discuss the reason why we chose to use fruits in this project by proposing a few applications that could use such classifier.



**Keywords: Deep learning, Object recognition, CNN, fruits dataset, image processing**


## 1. Introduction

The aim of this paper is to propose a new dataset of images containing popular fruits. The dataset was named Fruits-360 and can be downloaded from the addresses pointed by references [20] and [21]. Currently the set contains 82213 images of 120 fruits and vegetables and it is constantly updated with images of new fruits and vegetables as soon as the authors have accesses to them. The reader is encouraged to access the latest version of the dataset from the above indicated addresses. As the start of this project we chose the task of identifying fruits for several reasons. On one side, fruits have certain categories that are hard to differentiate, like the citrus genus, that contains oranges and grapefruits. 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 paper is structured as follows: in the first part we will shortly discuss a few outstanding achievements obtained using deep learning for fruits recognition, followed by a presentation of the concept of deep learning. In the second part we describe the Fruits-360 dataset: how it was created and what it contains. In the third part we will present the framework used in this project - TensorFlow[32] and the reasons we chose it. Following the framework presentation, we will detail the structure of the neural network that we used. We also describe the training and testing data used as well as the obtained performance. Finally, we will conclude with a few plans on how to improve the results of this project.

## 2. Deep learning

In the area of image recognition and classification, the most successful results were obtained using artificial neural networks [6, 30]. 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 [26]. Each level learns to transform its input data into a slightly more abstract and composite representation [6]. Deep neural networks have managed to outperform other machine learning algorithms. They also achieved the first superhuman pattern recognition in certain domains [5]. 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. In the rest of this section we will briefly describe some models of deep artificial neural networks along with some results for some related problems.

## 3.1 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 [34]. 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 [13] dataset is done by using multi-column deep neural networks. As described in paper [7], 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. Another paper [16] further reinforces the idea that convolutional networks have obtained better accuracy in the domain of computer vision. In paper [29] an all convolutional network that gains very good performance on CIFAR-10 [12] is described in detail. The paper proposes the replacement of pooling and fully connected layers with equivalent convolutional ones. This may increase the number of parameters and adds inter-feature dependencies however it can be mitigated by using smaller convolutional layers within the network and acts as a form of regularization. In what follows we will describe each of the layers of a CNN network.

### 3.1.1 Convolutional layers

Convolutional layers are named after the convolution operation. In mathematics convolution is an operation on two functions that produces a third function that is the modified (convoluted) version of one of the original functions. The resulting function gives in integral of the pointwise multiplication of the two functions as a function of the amount that one of the original functions is translated [33]. A convolutional layer consists of groups of neurons that make up kernels. The kernels have a small size but they always have the same depth as the input. The neurons from a kernel are connected to a small region of the input, called the receptive field, because it is highly inefficient to link all neurons to all previous outputs in the case of inputs of high dimensions such as images. For example, a 100 x 100 image has 10000 pixels and if the first layer has 100 neurons, it would result in 1000000 parameters. Instead of each neuron having weights for the full dimension of the input, a neuron holds weights for the dimension of the kernel input. The kernels slide across the width and height of the input, extract high level features and produce a 2 dimensional activation map. The stride at which a kernel slides is given as a parameter. The output of a convolutional layer is made by stacking the resulted activation maps which in turned is used to define the input of the next layer. Applying a convolutional layer over an image of size 32 X 32 results in an activation map of size 28 X 28. If we apply more convolutional layers, the size will be further reduced, and, as a result the image size is drastically reduced which produces loss of information and the vanishing gradient problem. To correct this, we use padding. Padding increases the size of a input data by filling constants around input data. In most of the cases, this constant is zero so the operation is named zero padding. ”Same” padding means that the output feature map has the same spatial dimensions as the input feature map. This tries to pad evenly left and right, but if the number of

columns to be added is odd, it will add an extra column to the right. ”Valid” padding is equivalent to no padding. The strides causes a kernel to skip over pixels in an image and not include them in the output. The strides determines how a convolution operation works with a kernel when a larger image and more complex kernel are used. As a kernel is sliding the input, it is using the strides parameter to determine how many positions to skip. ReLU layer, or Rectified Linear Units layer, applies the activation function max(0, x). It does not reduce the size of the network, but it increases its nonlinear properties.

### 3.1.2 Pooling layers

Pooling layers 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.

### 3.1.3 Fully connected layers

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.

### 3.1.4 Loss layers

Loss layers are used to penalize the network for deviating from the expected output. This is normally the last layer of the network. Various loss function 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).

## 4 Fruits-360 data set

In this section we describe how the data set was created and what it contains. 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 of paper as background. However due to the variations in the lighting conditions, the background was not uniform and we wrote a dedicated algorithm which extract the fruit from the background. This algorithm is of flood fill type: we start from each edge of the image and we mark all pixels there, then we mark all pixels found in the neighborhood of the already marked pixels for which the distance between colors is less than a prescribed value. we repeat the previous step until no more pixels can be marked. All marked pixels are considered as being background (which is then filled with white) and the rest of pixels are considered as belonging to the object. The maximum value for the distance between 2 neighbor pixels is a parameter of the algorithm and is set (by trial and error) for each movie. Fruits were scaled to fit a 100x100 pixels image. Other datasets (like MNIST) use 28x28 images, but we feel that small size is detrimental when

you have too similar objects (a red cherry looks very similar to a red apple in small images). Our future plan is to work with even larger images, but this will require much more longer training times. To understand the complexity of background-removal process we have depicted in Figure 1 a fruit with its original background and after the background was removed and the fruit was scaled down to 100 x 100 pixels. The resulted dataset has 82110 images of fruits and vegetables spread across 120 labels. Each image contains a single fruit or vegetable. Separately, the dataset contains another 103 images of multiple fruits. The data set is available on GitHub [20] and Kaggle [21].

## 5 The structure of the neural network used in experiments

For this project we used a convolutional neural network. As previously described this type of network makes use 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. Note again that a characteristic that sets apart the CNN from a regular neural network is taking into account the structure of the images while processing them. A regular neural network converts the input in a one dimensional array which makes the trained classifier less sensitive to positional changes.

• The first layer (Convolution #1) is a convolutional layer which applies 16 5 x 5 filters. On this layer we apply max pooling with a filter of shape 2 x 2 with stride 2 which specifies that the pooled regions do not overlap (Max-Pool #1). This also reduces the width and height to 50 pixels each.

• The second convolutional (Convolution #2) layer applies 32 5 x 5 filters which outputs 32 activation maps. We apply on this layer the same kind of max pooling(Max-Pool #2) as on the first layer, shape 2 x 2 and stride 2.

• The third convolutional (Convolution #3) layer applies 64 5 x 5 filters. Following is another max pool layer(Max-Pool #3) of shape 2 x 2 and stride 2.

• The fourth convolutional (Convolution #4) layer applies 128 5 x 5 filters after which we apply a final max pool layer (Max-Pool #4).

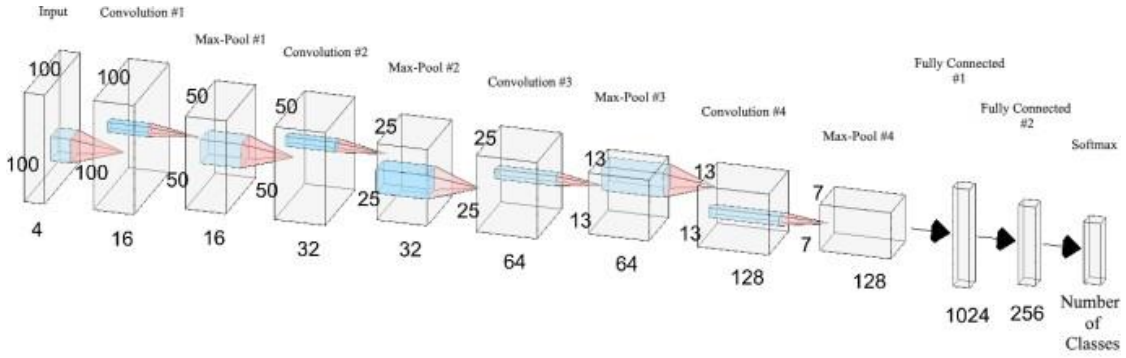


Figure 2: Graphical representation of the convolutional neural network used in experiments.

• Because of the four max pooling layers, the dimensions of the representation have each been reduced by a factor of 16, therefore the fifth layer, which is a fully connected layer(Fully Conne cted #1), has 7 x 7 x 16 inputs.

• This layer feeds into another fully connected layer (Fully Connected #2) with 1024 inputs and 256 outputs.

• The last layer is a softmax loss layer (Softmax) with 256 inputs. The number of outputs is equal to the number of classes.

## 6. Result :

I have used CNN model. After training the model I got approx. 96% accuracy in my model.
For more…. visit : https://github.com/11716807/Fruits-360-CNN-model

## 7. Conclusion

We described a new and complex database of images with fruits. Also we made some numerical experiments by using TensorFlow library in order to classify the images according to their content. From our 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 [29]. Another possibility is to replace the rectified linear units with exponential linear units.