Department of Computer & Software Engineering

Object Detection using Convolutional Neural Network

Shubham Jain

Supervisor- Michael Thornton

ABSTRACT

Convolutional neural networks (CNNs) are widely used in pattern- and image-recognition problems as they have a number of advantages compared to other techniques. This project will discuss the various applications of implementing a Convolutional Neural Network for Image Detection such as Facial Emotional Detection and Sight Prediction. The architecture of ConvNet and different layers would also be outlined. Data loss and efficiency while training will also be discussed. The applicability of final model is portrayed in a website where an interactive web application trains data and generates predictions.

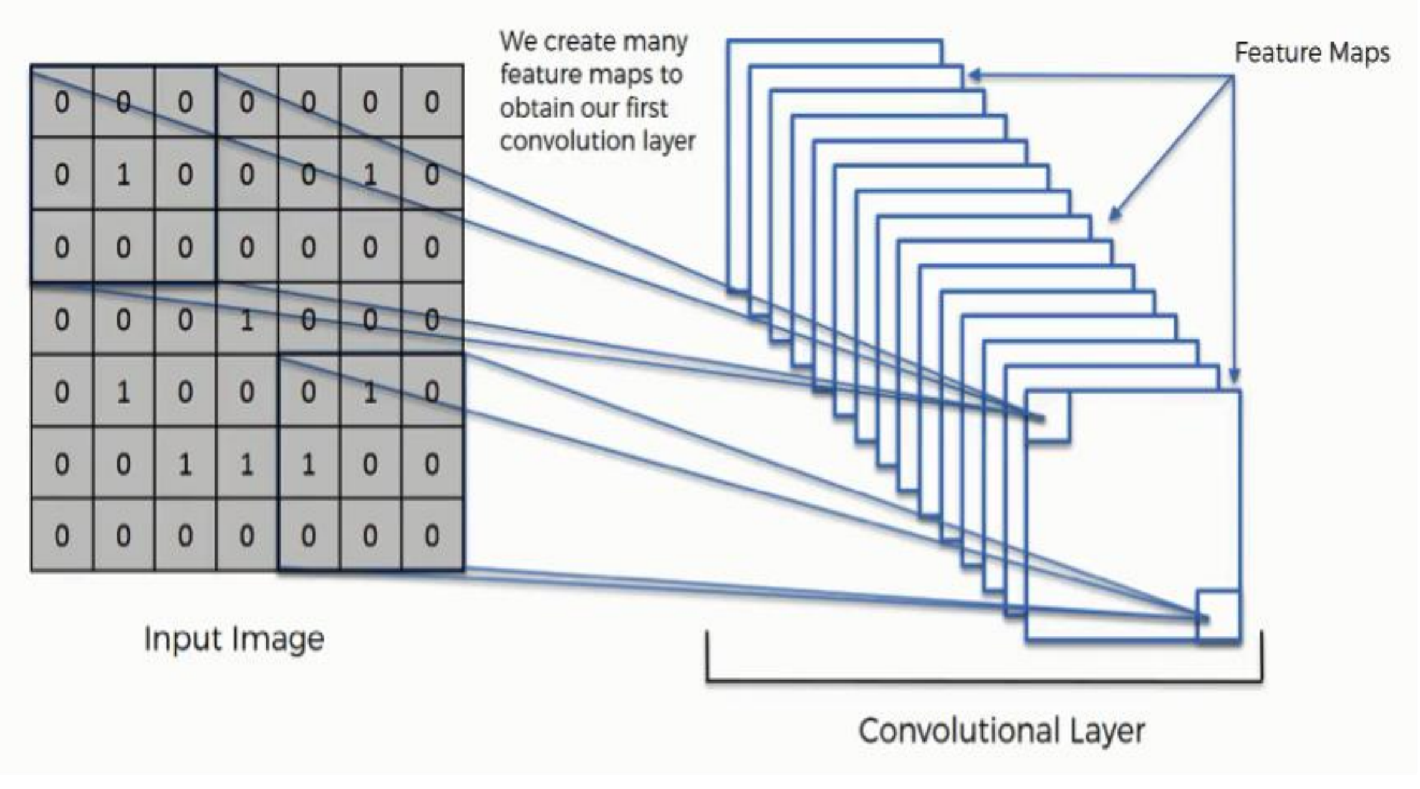
INTRODUCTION

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area. One of the most popular uses of this architecture is image classification.

METHODOLOGY

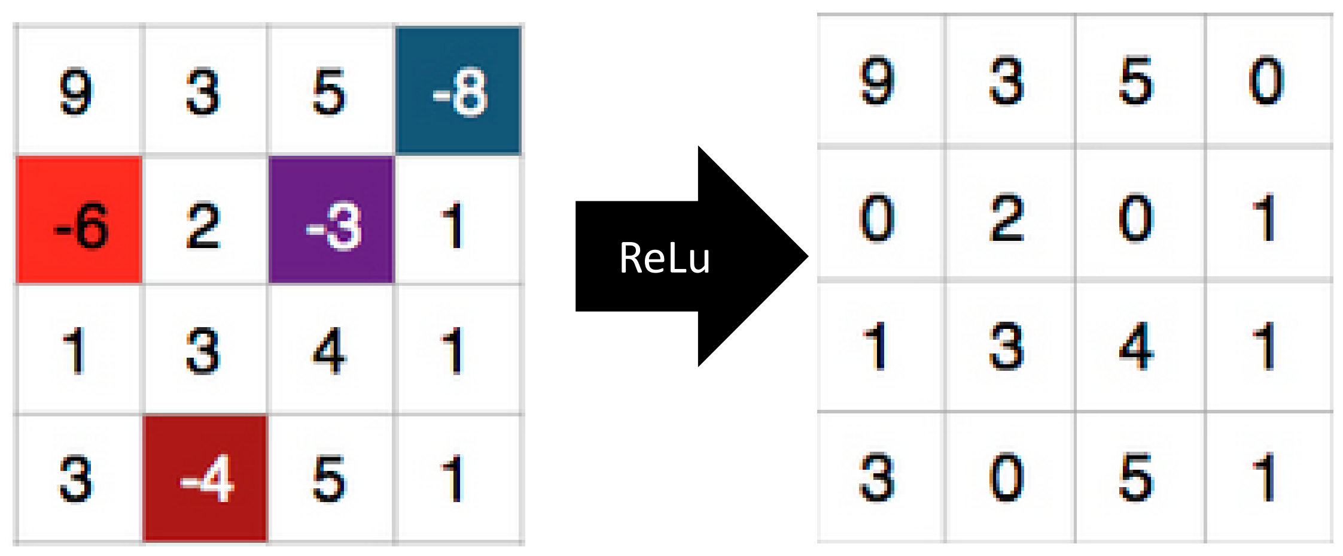
ConvNet mainly has four operations Convolution, Non Linearity (ReLU), Max Pooling or Sub Sampling and Classification (Fully Connected Layer). My project uses MobileNet tensorflow model for image classification which has 300 Million MACs and 3.47 Million parameters needed to perform inference on a single 224×224 RGB image. The first step is to have a depthwise conv of the image we capture through webcam.



0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

We create many feature maps to obtain our first convolution layer
Feature Maps
Input Image
Convolutional Layer

Figure 1: Convolve

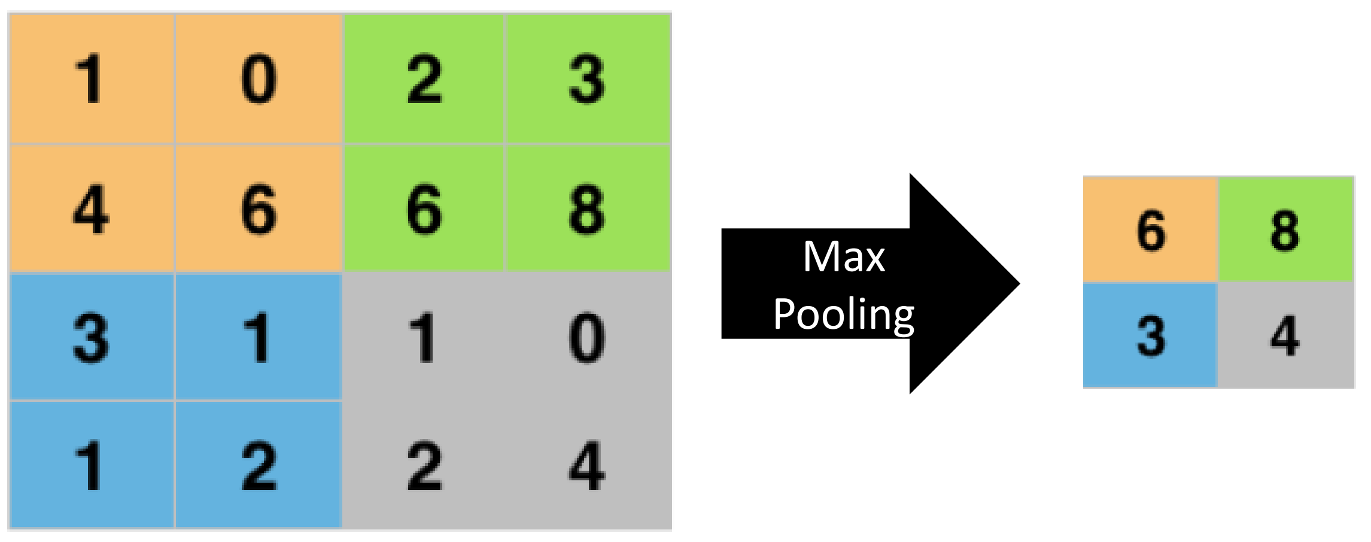


9	3	5	-8
-6	2	-3	1
1	3	4	1
3	-4	5	1

9	3	5	0
0	2	0	1
1	3	4	1
3	0	5	1

Figure 2: ReLu

ReLU a nonlinear Activation Function which makes it easy for the model to generalize or adapt with variety of data and to differentiate between the output is an element wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero.

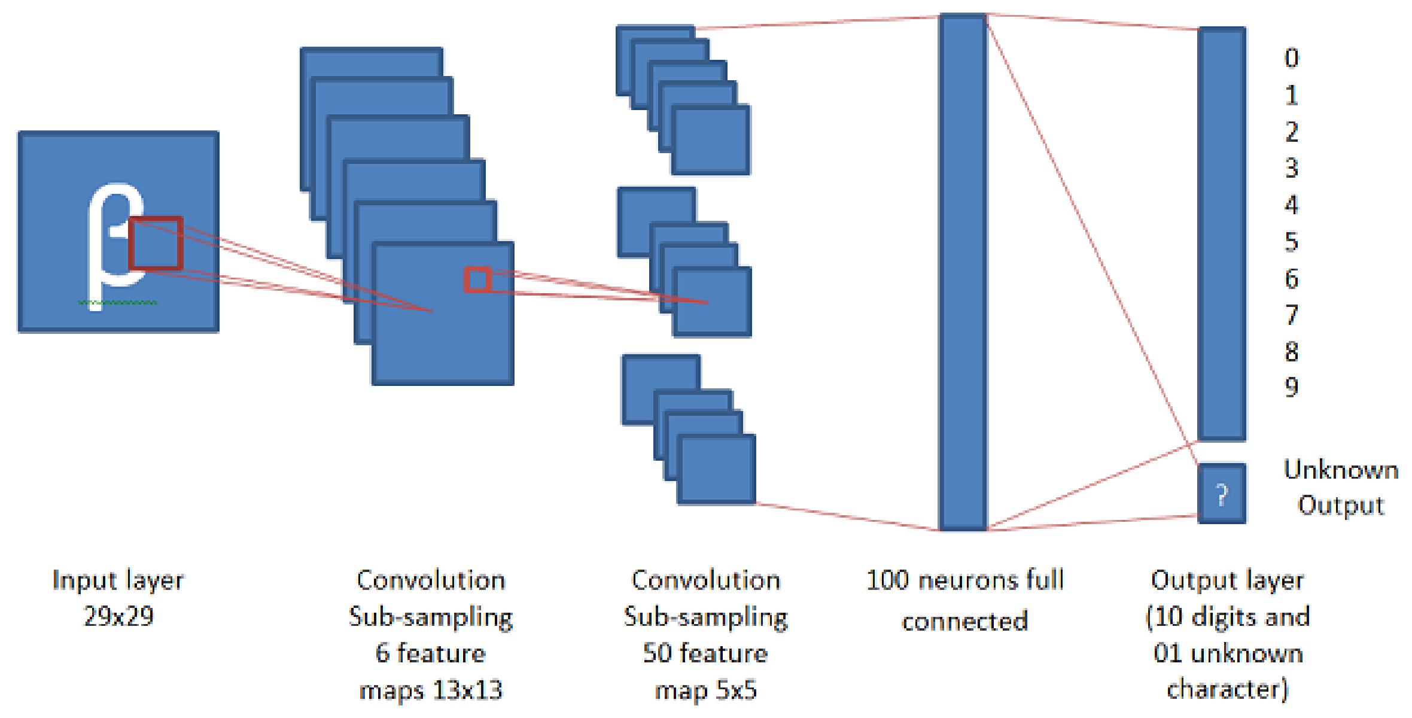


1	0	2	3
4	6	6	8
3	1	1	0
1	2	2	4

6	8
3	4

Figure 3: Max Pooling

The pooling layer works with width and height of the image and performs a downsampling operation on them. As a result the image volume is reduced. This means that if some features (as for example boundaries) have already been identified in the previous convolution operation, than a detailed image is no longer needed for further processing, and it is compressed to less detailed pictures.



β
0
1
2
3
4
5
6
7
8
9
?
Unknown Output
Input layer 29x29
Convolution Sub-sampling 6 feature maps 13x13
Convolution Sub-sampling 50 feature map 5x5
100 neurons full connected
Output layer (10 digits and 01 unknown character)

Figure 4: Fully Connected Layer

The Fully Connected layer is a traditional Multi Layer Perceptron that uses a softmax activation function in the output layer. The term “Fully Connected” implies that every neuron in the previous layer is connected to every neuron on the next layer. The output from the convolutional and pooling layers represent high-level features of the input image. The purpose of the Fully Connected layer is to use these features for classifying the input image into various classes based on the training dataset.

ARCHITECTURE

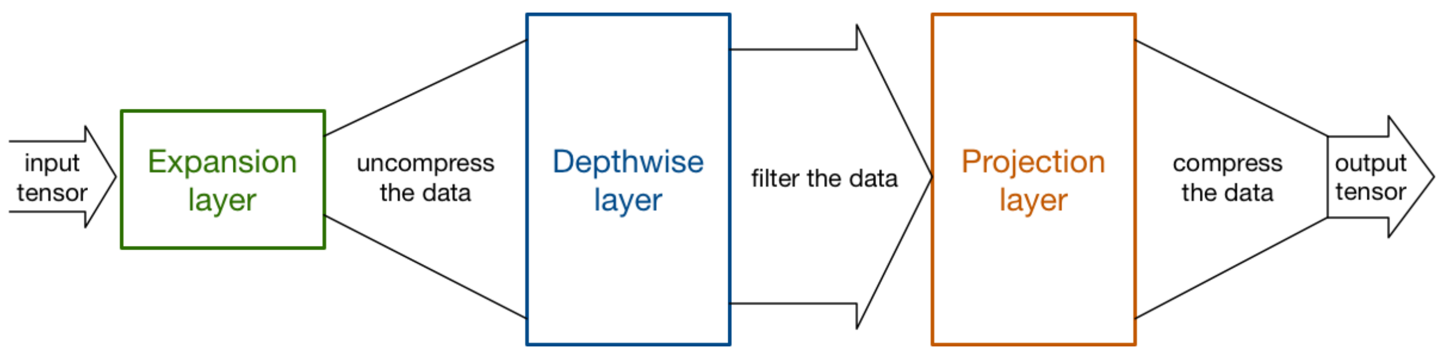
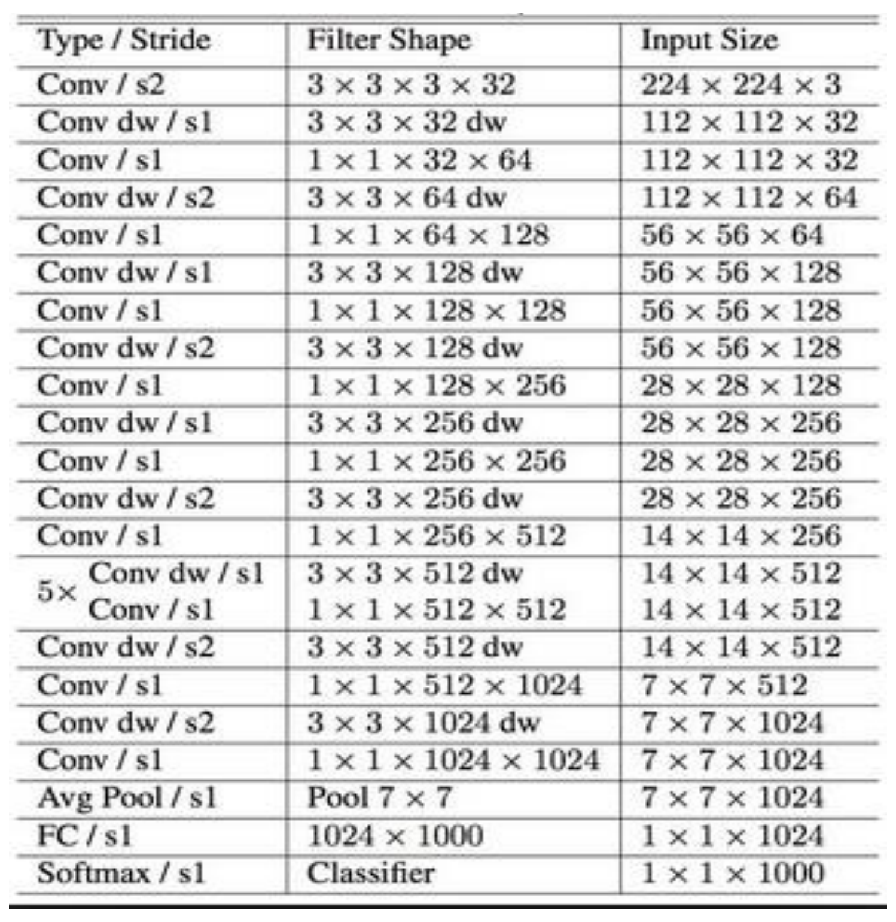


Figure 5: Architecture Depthwise ConvNet



Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1 Conv / s1	3 × 3 × 512 dw 1 × 1 × 512 × 512	14 × 14 × 512 14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

Figure 6: MobileNet Architecture

CONCLUSION

The model predicts emotions and sight with accuracy given good lighting conditions and background eliminations due to the state-of-the-art architecture of MobileNets which uses Depthwise Separable Convolution (DWConvolution) in place of standard Convolution to reduce the size of networks making it possible to work on low processing power devices. I have tested that this model works really well even with a small number of training data. I measured how the accuracy depends on the number of epochs in order to detect potential overfitting problem. I determined that 10 epochs are enough for a successful training of the model. My next step would be to try this model on more data sets such as fer2013, Japanese Female Facial Expression (JAFFE) database and Labeled Faces in the Wild (LFW) dataset for emotion detection. I want to work more on the visualizing the learned filters in depth, and in the future, we also want to take a semi-supervised approach by using the predictions made for the LFW images, to train the network with more data, more filters, and more depth.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, 2012.

[2] OpenSourceComputerVision. Face detection using haar cascades. URL http://docs.opencv.org/master/d7/d8b/tutorial_py_face_detection.html.

[3] TFlearn. Tflearn: Deep learning library featuring a higher-level api for tensorflow. URL http://tflearn.org/

Faculty of Engineering & Informatics