A Project report on

HANDWRITTEN DIGIT RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS

Submitted in partial fulfillment of the requirement for the award of degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

BY

A Bhavani(17JJ1A0502)

A Manogna(17JJ1A0503)

T Sushmitha(17JJ1A0550)

Under the Guidance of

Dr P.Sammulal

Professor, CSE Department

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(Accredited by NBA)

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

COLLEGE OF ENGINEERING

NACHUPALLY(KONDAGATTU), JAGTIAL DIST-505501, TELANGANA

2020-2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(Accredited by NBA)

CERTIFICATE

This is to certify that the major project named "**HANDWRITTEN DIGIT RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS**" is a bonafide work carried out by A.Bhavani(17JJ1A0502), A.Manogna(17JJ1A0503), T.Sushmitha(17JJ1A0550) in partial fulfilment of the requirements for the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** by the Jawaharlal Nehru Technological University, Hyderabad during the academic year 2020-2021.

The results embodied in this report have not been submitted to any other University or Institution for the award of any other degree or diploma.

Dr P.Sammulal	Dr T.Venugopal
Professor of CSE Department	Head and Professor of CSE
Project Guide	

External Examiner

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of the task would be put incomplete without the mention of the people who made it possible, whose guidance and encouragement crown all the efforts with success.

We wish to express our deep sense of gratitude to **Dr P Sammulal Garu**, Professor, CSE Department, our Project Guide for his able, vigorous guidance and useful suggestions, which helped us in completing the project work in time.

We show gratitude to our beloved Principal **Dr N.V. RAMANA**, Vice Principal **Dr S. VISWHANADHA RAJU**, for providing necessary infrastructure and resources for the accomplishment of our project report at JNTUHCEJ.

We also thank all the staff members of Computer Science and Engineering Department, JNTUHCEJ for their valuable support and generous advice.

DECLARATION

We do declare that the project work entitled "**HANDWRITTEN DIGIT RECOGNITION USING CONVOLUTIONAL NEURAL NEWTWORKS**" submitted by us in the Department of Computer Science and Engineering. JNTUH College of Engineering Jagtial, Telangana State in partial fulfilment of degree for the award of BACHELOR OF TECHNOLOGY is a bonafide work, which was carried out under the supervision of **Dr P Sammulal Garu**, Professor, CSE Department, JNTUH College of Engineering, Jagtial.

Also, we declare that the matter embedded in this thesis has not been submitted by us in full or partial thereof for the award of any degree/diploma of any other institution or University previously.

Place:

Date:

A Bhavani(17JJ1A0502)

A Manogna(17JJ1A0503)

T Sushmitha(17JJ1A0550)

ABSTRACT

Humans can see and visually sense the world around them by using their eyes and brains. Computer vision works on enabling computers to see and process images in the same way that human vision does. Several algorithms developed in the area of computer vision to recognize images. The Handwritten Digit Recognition problem becomes one of the most famous problems in machine learning and computer vision applications.

Handwritten digit recognition is the process of receiving and correctly interpreting a legible hand-drawn digit from an input source (paper or photographs) by comparing it with previously trained data. CNN is a type of Artificial Neural Network which is used in Image Recognition and Processing that is specially designed to process pixel data. CNN uses a system much like a multi-layer perceptron that has been designed for reducing processing requirements. The layers of CNN are input layer, hidden layer, output layer that include multiple convolutional layers, pooling layers, fully connected layers and Normalization layers.

The goal of our work will be to create a model that will be able to identify and determine the handwritten digit from its image with better accuracy. We aim to complete this by using the concepts of “Convolutional Neural Network” in Deep Learning. Images of digits were taken from a variety of scanned documents, normalized in size and centered. It is a digit recognition task. As such there are 10 digits (0 to 9) or 10 classes to predict. Using MNIST database and compiling with the CNN gives the basic structure of project development.

TABLE OF CONTENTS

1.Introduction

Objective

- To recognize handwritten digits correctly.
- To improve the accuracy of detection.
- To develop a method which is independent of digit size and style.

MNIST Dataset

Modified National Institute of Standards and Technology (MNIST) is a large set of computer vision dataset which is extensively used for training and testing different systems. It was created from the two special datasets of National Institute of Standards and Technology (NIST) which holds binary images of handwritten digits. The training set contains handwritten digits from 250 people, among them 50% training dataset was employees from the Census Bureau and the rest of it was from high school students [26]. However, it is often attributed as the first datasets among other datasets to prove the effectiveness of the neural networks.

The database contains 60,000 images used for training as well as few of them can be used for cross-validation purposes and 10,000 images used for testing [27]. All the digits are grayscale and positioned in a fixed size where the intensity lies at the center of the image with 28×28 pixels. Since all the images are 28×28 pixels, it forms an array which can be

flattened into 28*28=784 dimensional vector. Each component of the vector is a binary value which describes the intensity of the pixel.

Basic Definitions

Artificial Intelligence

In computer science, Artificial Intelligence (AI), sometimes called machine intelligence, is intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans.

Machine Learning

It is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence.

Deep Learning

Deep learning is a subset of machine learning where artificial neural networks, algorithms inspired by the human brain, learn from large amounts of data. Deep learning allows machines to solve complex problems even when using a data set that is very diverse, unstructured and inter-connected.

Convolution Neural Networks

Convolutional neural networks or CNNs, are a specialized kind of neural network for processing data that has a known grid-like topology. Examples include time-series data, which can be thought of as a 1-D grid taking samples at regular time intervals and image data, which can be thought of as a 2-D grid of pixels.

Deep Learning Jaragon

Perceptron

A Perceptron is a neural network unit that does certain computations to detect features or business intelligence in the input data. It is a function that maps its input x, which is multiplied by the learned weight coefficient, and generates an output value f(x).

Activation Function

To allow Neural Networks to learn complex decision boundaries, a nonlinear activation function is applied to some of its layers. Commonly used functions include sigmoid, tanh, ReLU (Rectified Linear Unit) and variants of these.

Feed Forward Neural Network

The feedforward neural network was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network. Compresses the input representation into a lower-dimensional representation.

Max Pooling

Max pooling is a sample-based discretization process. The objective is to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned.

Keras and Tensorflow

Keras is a neural network library while, TensorFlow is an open source library for a number of various tasks in machine learning. TensorFlow provides both high-level and low-level APIs while Keras provides only high-level APIs. Keras is built in Python which makes it way more user-friendly than TensorFlow.

Back Propagation

Backpropagation is an algorithm to efficiently calculate the gradients in a Neural Network, or more generally, a feedforward computational graph. It boils down to applying the chain rule of differentiation starting from the network output and propagating the gradients backward. The first uses of backpropagation go back to Vapnik in the 1960's but Learning representations by back-propagating errors is often cited as the source.

Aim

The goal of our work will be to create a model that will be able to recognize and determine the handwritten digits from its image. We aim to complete this by using the concepts of Convolution Neural Network. The aim of our study is to open the way to digitalization. Though the goal is to just to create a model which can recognize the digits but it can be extended to letters and then a person's handwriting. Through this work, we aim to learn and practically apply the concepts of Machine Learning and Neural Networks. Moreover, digit recognition is an excellent prototype problem for learning about neural networks and it gives a great way to develop more advanced techniques like deep learning.

2. System Requirements

The following are the hardware and software requirements that have used to implement the proposed system.The section of hardware is very important in the existence and proper working of any software. In the selection of hardware, the size and the capacity requirements are also important.

Hardware Requirement's:

- Processor : Minimum 1GHz; Recommended 2GHz or more
- RAM : Minimum 1GB; Recommended 4GB or more
- Hard Disk : Minimum 32GB; Recommended 64GB or more

Software Requirement's:

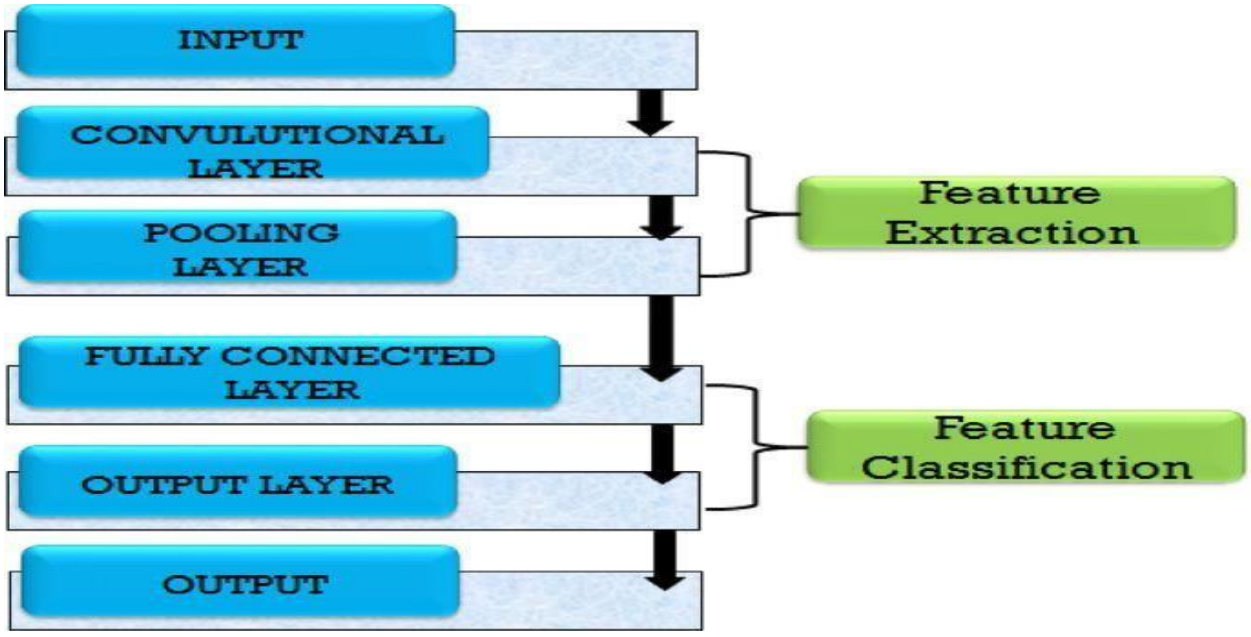
- Programming Language : Python
- OS : Windows/Linux
- Platform : Jupyter Notebook / Colab
- Other Libraries Used : Tensorflow,Keras,Opencv,Matplotlib, Numpy
- Input : Paint

3. System Design and Architecture

Convolutional Neural Network

- CNNs are a special type of neural networks.
- They can be divided into two parts: A feature learning part and a classification part.
- Each part consists of one or multiple layers.
- Feature learning is typically done by combining two types of layers: Convolution layers and pooling layers .
- Classification is then performed based on the learned features through dense layers, also known as fully connected layers.
- Additionally there is an input layer, containing the image data, as well as an output layer, containing the different classes we are trying to predict .

- **FlowChart for CNN**



The convolution neural network (CNN) is a class of **deep learning neural networks**. CNNs represent a huge breakthrough in image recognition. They're most commonly used to analyze visual imagery and are frequently working behind the scenes in image classification. They can be found at the core of everything from Face book's photo tagging to self-driving cars. They're working hard behind the scenes in everything from healthcare to security.

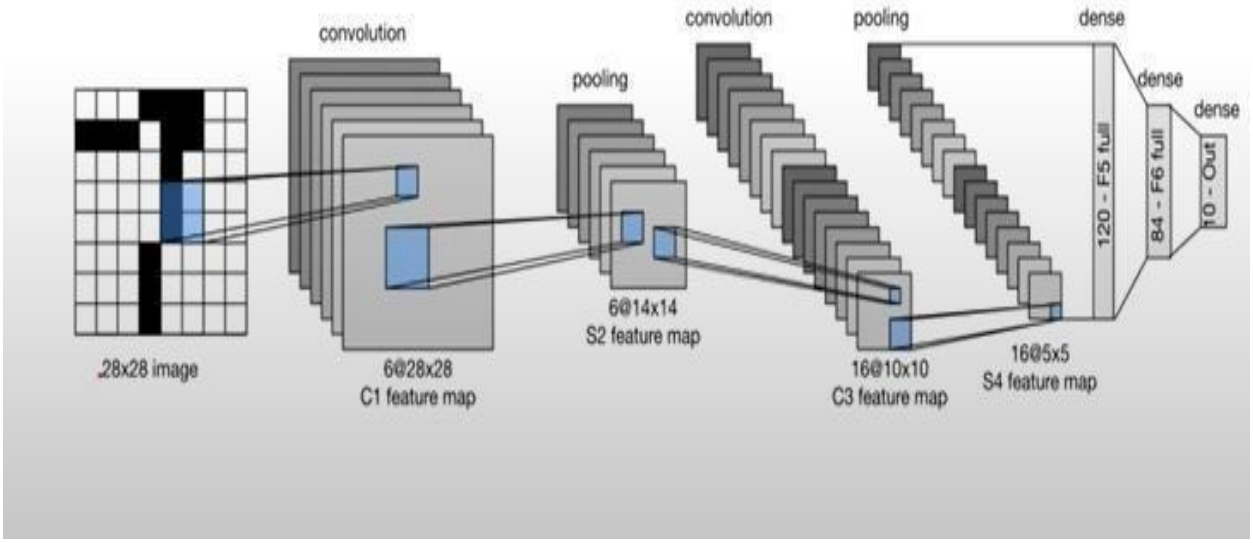
They're fast and they're efficient. Image classification is the process of taking an **input** (like a picture) and outputting a **class** (like "cat") or a **probability** that the input is a particular class ("there's a 90% probability that this input is a cat").

Convolution Neural Networks - Plan of Attack

These are the steps (our Plan of Attack) we'll follow to master Convolution Neural Networks and consequently Deep Learning:

CNNs aren't an easy feat to master. This is the first step in your journey - so get a grip on the basics before getting started.

A Convolution Neural Networks Introduction so to speak.



- Step 1: Convolution Operation

  The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped out.

- Step 1(b): ReLU Layer

  The second part of this step will involve the Rectified Linear Unit or ReLU. We will cover ReLU layers and explore how linearity functions in the context of Convolution Neural Networks. Not necessary for understanding CNN's, but there's no harm in a quick lesson to improve your skills.

- Step 2: Pooling

  In this part, we'll cover pooling and will get to understand exactly how it generally works. Our nexus here, however, will be a specific type of pooling; max pooling. We'll cover various approaches, though, including mean (or sum) pooling. This part will end with a demonstration made using a visual interactive tool that will definitely sort the whole concept out for you.

- Step 3: Flattening

  This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolution Neural Networks

- Step 4: Full Connection

  In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolution Neural Networks operate and

how the "neurons" that are finally produced learn the classification of images.

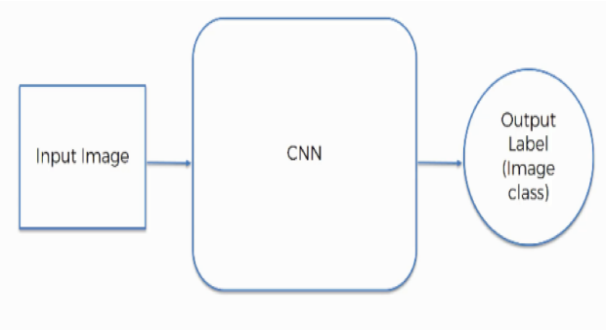
- Softmax & Cross-Entropy

So, how do convolution neural networks actually operate?

The first thing you need to know here is the elements that are included in the operation:

- Input Image
- Convolution Neural Network
- Output Label(Image Class)

These elements interact in the following manner:

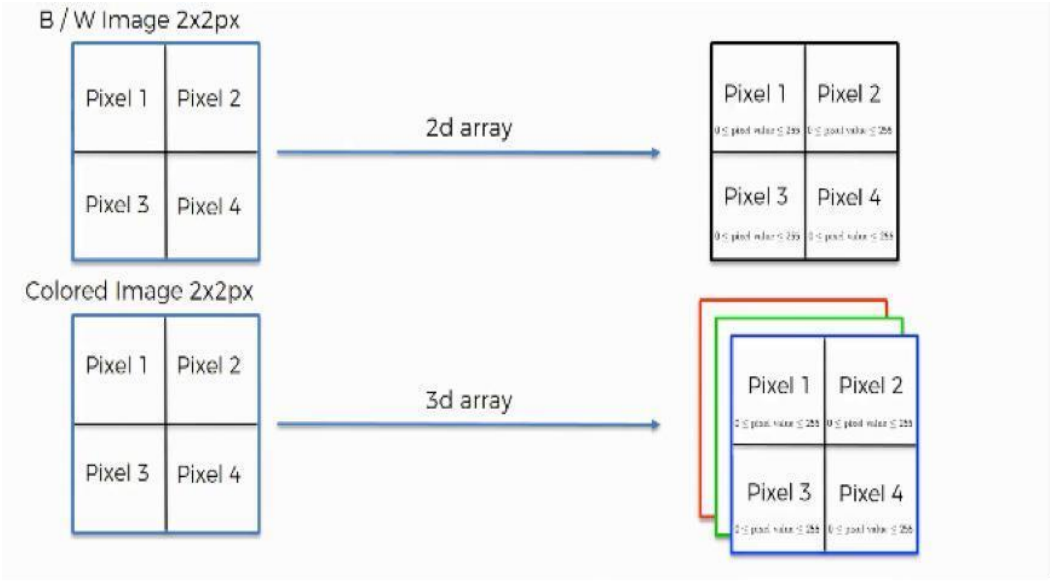


For example, convolution neural networks can be used in detected human emotions in an image. You provide them with someone's photo, and they produce a classification to the effect of what that person seems to be feeling.

Of course, this requires a somewhat more advanced level of training since being able to infer someone's emotions from their facial expressions is often quite a puzzling task to humans themselves.

Naturally, only a small portion of people can do so with a fairly high degree of success.

How Convolution Neural Networks Scan Images



Scanning of CNN

Both types of images are similar in the following respects:

- Each pixel contains 8 bits (1 byte) of information.
- Colors are represented on a scale from 0 to 255. The reason for this is that bits are binary units, and since we have 8 of these per

byte, a byte can have any of 256 (2^8) possible values. Since we count 0 as the first possible value, we can go up to 255.

- In this model, 0 is pitch black and 255 is pure white, and in between are the various (definitely more than 50!) shades of gray.
- The network does not actually learn colors. Since computers understand nothing but 1's and 0's, the colors' numerical values are represented to the network in binary terms.

Now let's delve into the major difference that we mentioned above.

Black & white images are two-dimensional, whereas colored images are three-dimensional. The difference this makes is in the value assigned to each pixel when presented to the neural network. In the case of two-dimensional black & white images, each pixel is assigned one number between 0 and 255 to represent its shade.

On the other hand, each pixel inside a colored image is represented on three levels. Since any color is a combination of red, green, and blue at different levels of concentration, a single pixel in a colored image is assigned a separate value for each of these layers.

That means that the red layer is represented with a number between 0 and 255, and so are the blue and the green layers. They are then

presented in an RGB format. For example, a "hot pink" pixel would be presented to the neural network as (255, 105, 180).

The steps that go into this process are broken down as follows:

- Step 1: Convolution
- Step 1b: ReLU Layer
- Step 2: Pooling
- Step 3: Flattening
- Step 4: Full Connection

Convolution

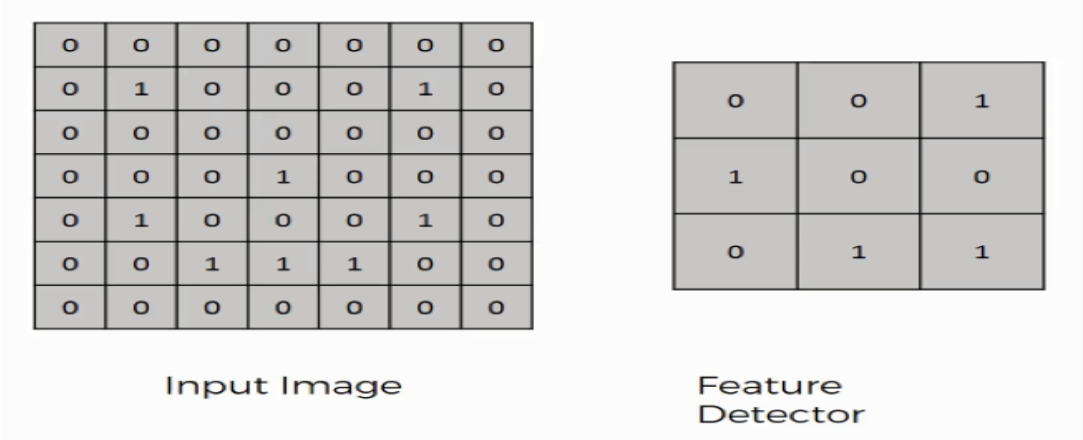
Step 1- Convolution Operation

In purely mathematical terms, convolution is a function derived from two given functions by integration which expresses how the shape of one is modified by the other. That can sound baffling as it is, but to make matters worse, we can take a look at the convolution formula:

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

Convolution formula

The Convolution Operation



Here are the three elements that enter into the convolution operation:

- Input Image
- Feature detector
- Feature Map

As you can see, the input image is the same smiley face image that we had in the previous tutorial. Again, if you look into the pattern of the 1's and 0's, you will be able to make out the smiley face in there.

Sometimes a 5x5 or a 7x7 matrix is used as a feature detector, but the more conventional one, and that is the one that we will be working with, is a 3x3 matrix. The feature detector is often referred to as a "kernel" or a "filter," which you might come across as you dig into other material on the topic.

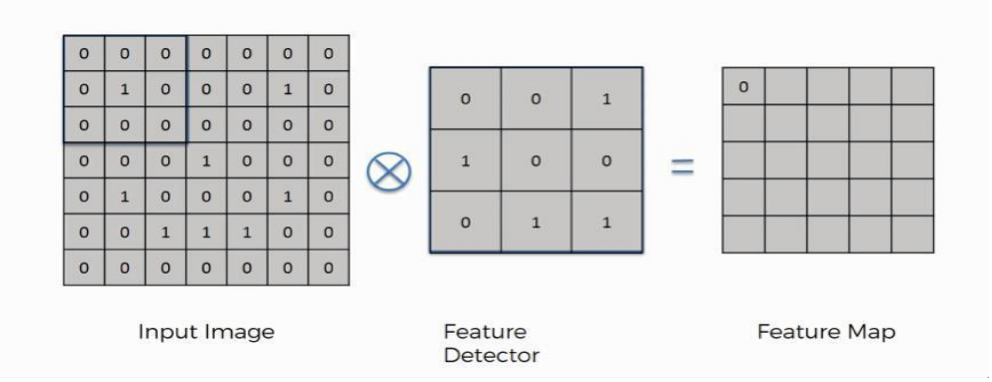
It is better to remember both terms to spare yourself the confusion. They all refer to the same thing and are used interchangeably, including in this course.

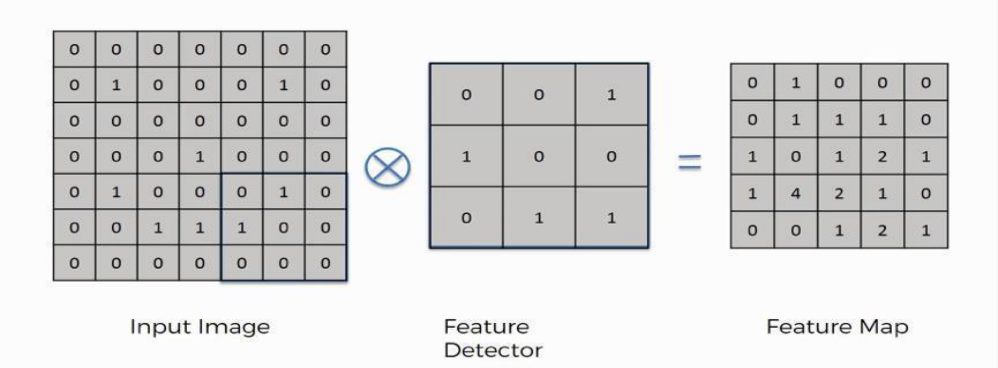
How exactly does the convolution operation work?

You can think of the feature detector as a window consisting of 9 (3x3) cells. Here is what you do with it:

- You place it over the input image beginning from the top-left corner within the borders you see demarcated above, and then you count the number of cells in which the feature detector matches the input image.
- The number of matching cells is then inserted in the top-left cell of the feature map.

- You then move the feature detector one cell to the right and do the same thing. This movement is called a and since we are moving the feature detector one cell at time, that would be called a stride of one pixel.
- What you will find in this example is that the feature detector's middle-left cell with the number 1 inside it matches the cell that it is standing over inside the input image. That's the only matching cell, and so you write "1" in the next cell in the feature map, and so on and so forth.
- After you have gone through the whole first row, you can then move it over to the next row and go through the same process.





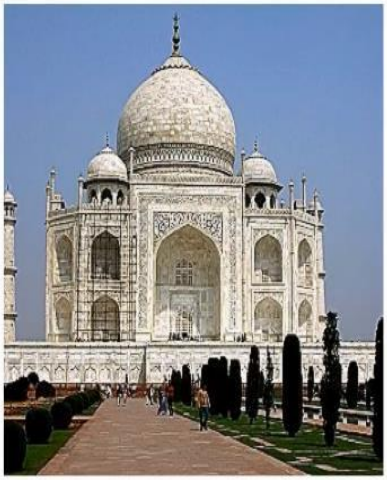
What are other uses of convolution matrices?

There's another use for convolution matrix, which is actually part of the reason why they are called "filters." The word here is used in the same sense we use it when talking about Instagram filters.

You can actually use a convolution matrix to adjust an image. Here are a few examples of filters being applied to images using these matrices.

Sharpen:

0	0	0	0	0
0	0	-1	0	0
0	-1	5	-1	0
0	0	-1	0	0
0	0	0	0	0



Blur:

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

Edge Detect:

	0	1	0	
	1	-4	1	
	0	1	0	

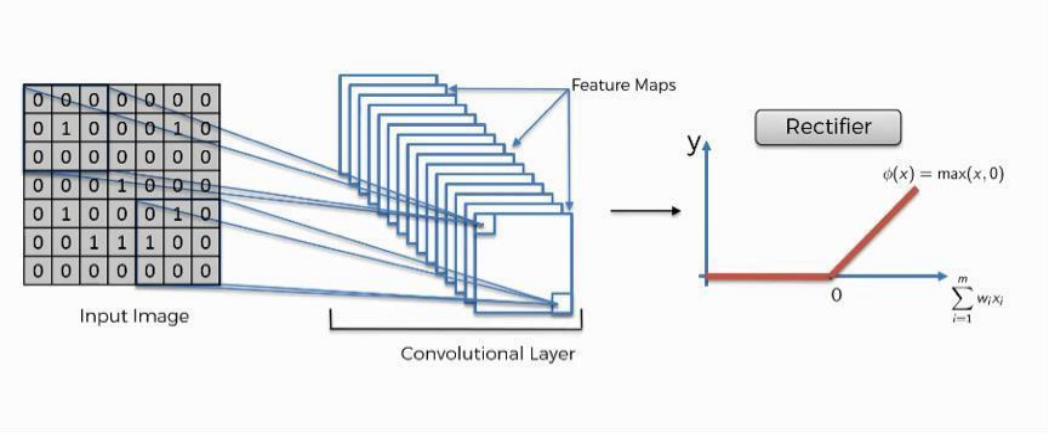
There is really little technical analysis to be made of these filters and it would be of no importance to our tutorial. These are just intuitively formulated matrices. The point is to see how applying them to an image can alter its features in the same manner that they are used to detect these features.

Step 1(b): The Rectified Linear Unit (ReLU)

The Rectified Linear Unit, or ReLU, is not a separate component of the convolution neural networks' process.

It's a supplementary step to the convolution operation that we covered in the previous tutorial. There are some instructors and authors who discuss both steps separately, but in our case, we're going to consider both of them to be components of the first step in our process.

If you're done with the previous section on artificial neural networks, then you should be familiar with the rectifier function that you see in the image below.



Rectifier Function for Convoluted Image

Pooling

Step 2 - Max Pooling

Types of Pooling

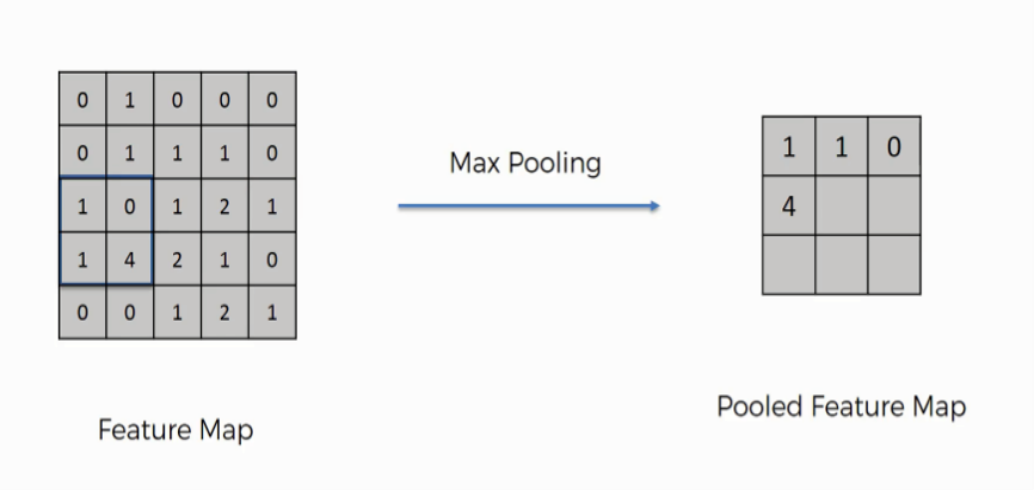
Before getting into the details, you should know that there are several types of pooling. These include among others the following:

- Mean pooling
- Max pooling
- Sum pooling

Pooled Feature Map

The process of filling in a pooled feature map differs from the one we used to come up with the regular feature map. This time you'll place a 2x2 box at the top-left corner, and move along the row.

For every 4 cells your box stands on, you'll find the maximum numerical value and insert it into the pooled feature map. In the figure below, for instance, the box currently contains a group of cells where the maximum value is 4.

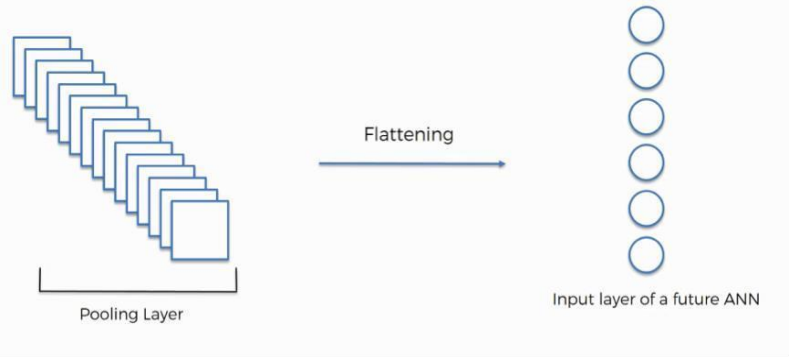


Flattening Step 3:

Flattening

After finishing the previous two steps, we're supposed to have a pooled feature map by now. As the name of this step implies, we are literally going to flatten our pooled feature map into a column like in the image below.

The reason we do this is that we're going to need to insert this data into an artificial neural network later on.



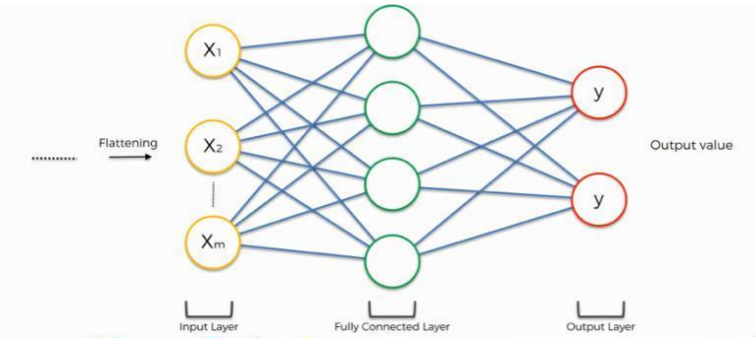
As you see in the image above, we have multiple pooled feature maps from the previous step. What happens after the flattening step is that you end up with a long vector of input data that you then pass through the artificial neural network to have it processed further.

Full Connection Step 4:

Full Connection

Here's where artificial neural networks and convolution neural networks collide as we add the former to our latter.

It's here that the process of creating a convolution neural network begins to take a more complex and sophisticated turn.



SoftMax

The softmax function goes like this:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}}$$

Softmax Function

4. Technologies Used

Python

Python is a robust high-level object-oriented program writing language. An excellent language for beginners due to its readability and various other structural elements made to make it simple to understand, Python isn't limited by basic usage. Actually, it powers a few of the world's most complicated applications and website. First created in the late 80s by Guido van Rossum, Python happens to be in its third edition, Python is normally in its third version currently, released in 2008, although the second version originally released in 2000 is still in common usage.

Python API 's and Libraries

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

TensorFlow

It is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

It is a symbolic math library based on dataflow and differentiable programming.

It is used to create deep learning models directly by using wrapper libraries and simple the process for large numerical computations.

Keras

It is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3 Keras supported multiple backends, including TensorFlow.

Numpy

NumPy is the fundamental package for scientific computing with Python. It contains several other things:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities.

Open CV

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human.

Python IDE

Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code,

equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Paint

Microsoft Paint is a simple raster graphics editor that has been included with all versions of Microsoft Windows. The program opens and saves files in Windows bitmap, JPEG, GIF, PNG, and single-page TIFF formats. The program can be in color mode or two-color black-and-white, but there is no grayscale mode**.**

.

5. Implementation

Import Libraries

Initially we need to import all the modules which are required for building the model.The syntax for importing libraries are as follows:

Code:

```
import matplotlib.pyplot as plt

import tensorflow as tf

import numpy as np

import cv2

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense,Dropout,Activation,Flatten,Conv2D,MaxPooling2D
```

Loading MNIST DataSet

```
mnist=tf.keras.datasets.mnist
```

Divide into Train and Test Datasets

```
(x_train,y_train),(x_test,y_test)=mnist.load_data()

x_train.shape

import matplotlib.pyplot as plt

plt.imshow(x_train[0])
```

```
plt.show()

plt.imshow(x_train[0],cmap=plt.cm.binary)
```

Preprocessing

Normalization

```
x_train=tf.keras.utils.normalize(x_train,axis=1)

x_test=tf.keras.utils.normalize(x_test,axis=1)

plt.imshow(x_train[0],cmap=plt.cm.binary)
```

Creating a Deep Neural Network

```
model=Sequential()

model.add(Conv2D(64,(3,3),input_shape=x_trainr.shape[1:]))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64,(3,3)))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64,(3,3)))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool_size=(2,2))

model.add(Flatten())
```

```
model.add(Dense(64))

model.add(Activation("relu"))

model.add(Dense(32))

model.add(Activation("relu"))

model.add(Dense(10))

model.add(Activation('softmax'))

model.summary()
```

Training the Model

The Accuracy obtained on 60,000 Training samples by performing 5 Epoch is 99%.

Code:

```
model.compile(loss="sparse_categorical_crossentropy",optimizer="
adam",metrics=['accuracy'])
model.fit(x_trainr,y_train,epochs=5,validation_split=0.3)
```

Evaluating Test dataset

The Accuracy obtained on 10,000 test samples is 98%.

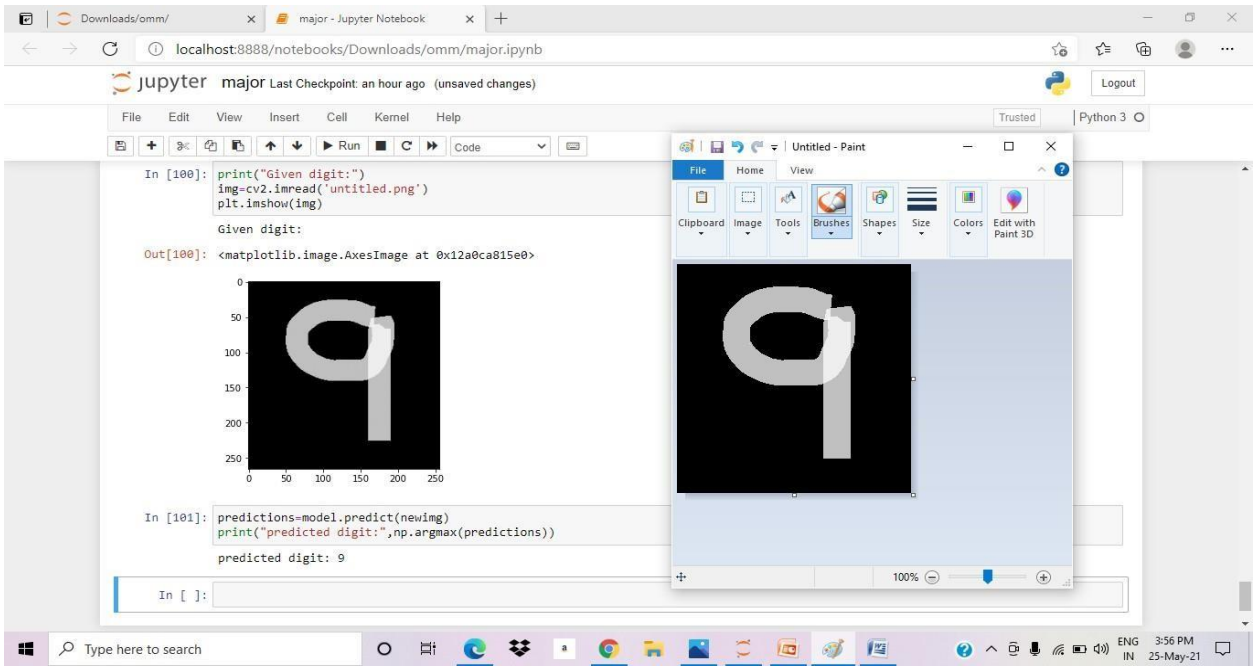
Code:

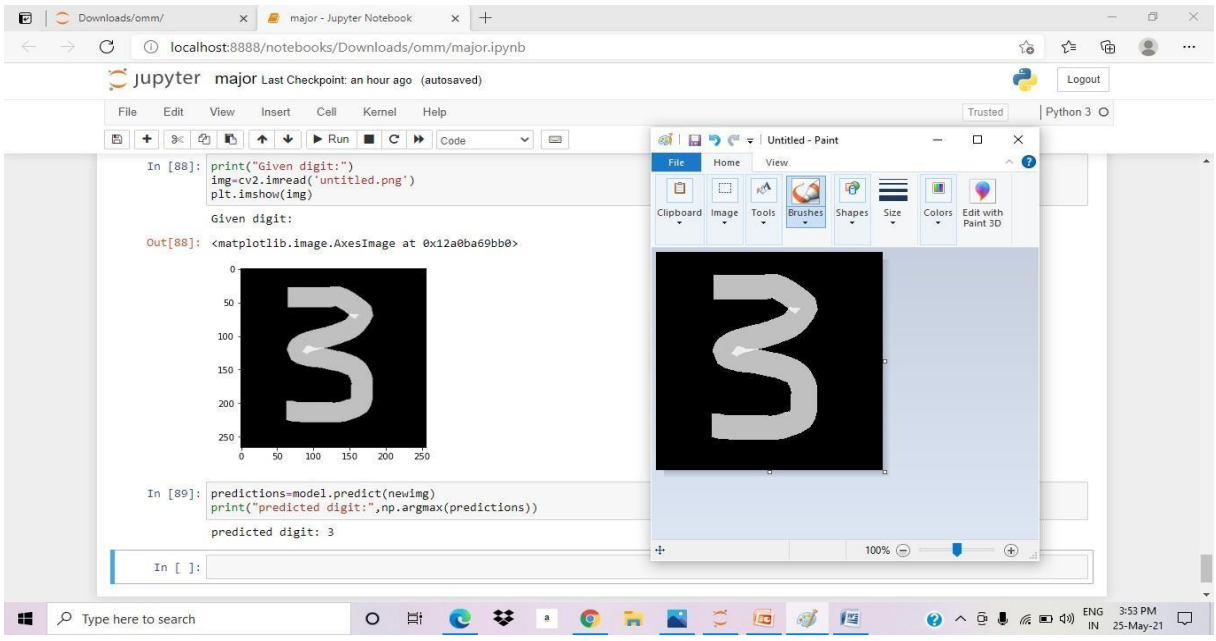
```
test_loss,test_acc=model.evaluate(x_testr,y_test)
print("test loss on 10,000 test samples", test_loss)
print("validation Accuracy on 10,000 test samples",test_acc)
predictions=model.predict([x_testr])
print(predictions)
print(np.argmax(predictions[0]))
plt.imshow(x_test[0])
print(np.argmax(predictions[130]))
plt.imshow(x_test[130])
print(y_test[130])
```

Predicting the new Image

```
img=cv2.imread('nine.png')
plt.imshow(img)
img.shape
gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
gray.shape
resized=cv2.resize(gray,(28,28),interpolation=cv2.INTER_AREA)
resized.shape
newimg=tf.keras.utils.normalize(resized,axis=1)
newimg=np.array(newimg).reshape(-1,IMG_SIZE,IMG_SIZE,1)
newimg.shape
predictions=model.predict(newimg)
print(np.argmax(predictions))
```

6. Result


Downloads/omm/
major - Jupyter Notebook
localhost:8888/notebooks/Downloads/omm/major.ipynb
jupyter major Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
In [100]:
print("Given digit:")
img=cv2.imread('untitled.png')
plt.imshow(img)
Given digit:
Out[100]:
<matplotlib.image.AxesImage at 0x12a0ca815e0>
In [101]:
predictions=model.predict(newimg)
print("predicted digit:",np.argmax(predictions))
predicted digit: 9
In [ ]:
Untitled - Paint
File Home View
Clipboard Image Tools Brushes Shapes Size Colors Edit with Paint 3D
100%
Type here to search
ENG IN
3:56 PM
25-May-21


Downloads/omm/
major - Jupyter Notebook
localhost:8888/notebooks/Downloads/omm/major.ipynb
jupyter major Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
In [88]:
print("Given digit:")
img=cv2.imread('untitled.png')
plt.imshow(img)
Given digit:
Out[88]:
<matplotlib.image.AxesImage at 0x12a0ba69bb0>
In [89]:
predictions=model.predict(newimg)
print("predicted digit:",np.argmax(predictions))
predicted digit: 3
In [ ]:
Untitled - Paint
File Home View
Clipboard Image Tools Brushes Shapes Size Colors Edit with Paint 3D
100%
Type here to search
ENG IN
3:53 PM
25-May-21

7. Conclusion

The handwritten digit recognition using CNN has provided to be of a fairly good efficiency.It works better than any other algorithm, including artificial neural networks.We can conclude that we reached the computer to the human's brain by the importance, use of isolated digits recognition for different applications.It is widely used in automatic processing of bank cheques, postal addresses , in mobile phones.

Here we demonstrate a model which can recognize handwritten digit. Later it can be extended for character recognition and real-time person's handwriting. Handwritten digit recognition is the first step to the vast field of Artificial Intelligence and Computer Vision. As seen from the results of the experiment, CNN proves to be far better than other classifiers. The results can be made more accurate with more convolution layers and more number of hidden neurons. It can completely abolish the need for typing. Digit recognition is an excellent prototype problem for learning about neural networks and it gives a great way to develop more advanced techniques of deep learning. In future, we are planning to develop a real-time handwritten digit recognition system.

8. References

- https://dev.to/frosnerd/handwritten-digit-recognition-using-convolutional-neural-networks-11g0
- https://towardsdatascience.com/understanding-convolutions-and-pooling-in-neural-networks-a-simple-explanation-885a2d78f211
- https://arxiv.org/ftp/arxiv/papers/1909/1909.08490.pdf
- https://www.hindawi.com/journals/isrn/2012/834127/
- https://youtu.be/u3FLVbNn9Os