KANTIPUR ENGINEERING COLLEGE

(Affiliated to Tribhuvan University)

Dhapakhel, Lalitpur

[Subject Code: CT755]

A MAJOR PROJECT MID-TERM REPORT ON

IMAGE CLASSIFICATION USING CONVOLUTION NEURAL NETWORK

Submitted by:

Aabhushan Pokharel [45/BCT/2071]

Prabin Bhusal [63/BCT/2071]

Rajesh Mahato [67/BCT/2071]

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

February, 2017

IMAGE CLASSIFICATION USING CONVOLUTION NEURAL NETWORK

Submitted by:

Aabhushan Pokharel [45/BCT/2071]

Prabin Bhusal [63/BCT/2071]

Rajesh Mahato [67/BCT/2071]

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

Kantipur Engineering College

Dhapakhel, Lalitpur

February, 2017

ABSTRACT

With increasing number of images uploaded daily in the internet,there is a need of an automated system that can precisely classify the imgage in appropriate category.Big Data ,image processing,machine learning and natural language processing are working togther to provide vison to the machine which are the potentionally revolutionary teachnoligies in computer sciences.

Convolutional Neural Network (CNN, or ConvNet)t has successfully been applied to analyzing visual imagery.The convolutional layer is the core building block of a CNN. The layer's parameters consist of a set of learnable filters (or kernels), which have a small receptive field, but extend through the full depth of the input volume. As a result, the network learns filters that activate when it detects some specific type of feature at some spatial position in the input.CIFAR-10 data sets makes easier for training and testing the model with 80 million images in 10 object class.

Even intellegent and powerful machines are struggling to understand the images.our goal is to design an intellegent system which can detect the objects in the image and classify them in precise category.

Keywords – Image Classification, Convolutional Neural Network, CIFAR-10

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Background

The human visual cortex is capable of recognizing and classifying a vast amount of objects that it encounters. The visual cortex is a collection of neurons connected in a certain way. A particular neuron is activated once a synapse fires depending on whether the threshold of activation was met. Neural Networks are based on this principle with weights being the analogue to the synapses. So it was natural that neural networks would be the preferred choice for image classification through object recognition.

While neural networks provided a methodology for image classification, the accuracy left room for improvement. So a new type of neural network was developed which was better suited to take images as inputs. This neural network was the Convolutional Neural Network or CovNet. This neural network has four stages including convolution, non-linearity or Rectified Non-Linearity (ReLU), pooling and fully connected layer which is a traditional multilayer perceptron. The convolution stage extracts the features from the image, ReLU introduces non-linearity as all real world images are non-linear, and pooling step reduces the dimensionality of the feature map. The fully connected layer takes the output from the above steps and classifies the image according to training dataset.

The output of the CovNet is a list of probabilities assigned to each class. The CovNet classifies the image according to the highest assigned probability. So, if an image contains a dog, a CovNet would assign a high probability of 0.94 to the dog while only 0.06 to a cat in a two class training dataset. The probability is assigned according to the weights which are adjusted by the CovNet once the inputs are initialized.

1.2 Problem Statement

The number of images online is ever growing. The number of pictures uploaded to the web is increasing day by day. The improvement in cellphone camera technology has only added to this proliferation It has reached such an extent that it becomes impossible for anyone to manually classify these images. If an automated system that can detect the object in the image and classify them is built, then this problem could be solved

1.3 Objective

- To implement convolutional neural network to build an image classification system

1.4 Application

- Automated image organization
- In stock photography websites for effective tagging
- In large image databases

1.5 Features

- Detect the object in the image
- Possible to classify different images in upto 10 classes

1.6 Feasibility Analysis

To investigate and analyze the existing procedures and to establish the key system requirements of the system, feasibility studies were conducted before starting the project. Several alternatives were explored to determine the feasibility of the project. The breakdown of the analysis is as follows.

1.6.1 Economic Feasibility

Economic feasibility determines if the project is economically viable. Since there are no hardware components to be purchased for this project, the project is viable. The project can be completed within reasonable resources as money need not be spent to complete it.

1.6.2 Technical Feasibility

The technical implementation of the software being developed is covered by existing technology. Also the system can be upgraded later as it is being built under software engineering principles. Furthermore, the required resources are available as open source as well as the existence of an online community also helps in this regard. So the project is technically feasible.

1.6.3 Operational Feasibility

The algorithm for the system being built has been implemented before by other developers and tested rigorously. The team working on the project will also be putting the system under test with different inputs to determine the operational feasibility.

1.7 System Requirements

1.7.1 Hardware Requirements

- Intel i5 processor (recommended)
- 8GB RAM (recommended)

1.7.2 Software Requirements

- Windows 7/8/10 64 bit or Mac OS X Lion or higher
- Python

- Keras

CHAPTER 2
LITERATURE REVIEW

2.1 AlexNet

In 2012, Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton released AlexNet which became the basis for most works based on Convolutional Neural Network henceforth. It won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) by a large margin in 2012.

The AlexNet used a subset ImageNet dataset which included 15 million labeled high resolution images belonging to roughly 22,000 categories. However, ILSVRC in which the algorithm was implemented used 1.2 million training images, 50,000 validation images and 150,000 testing images with 1000 images in 1000 categories.

The AlexNet architecture contains eight layers with weights. The first five layers are convolutional layers and the remaining three are fully connected. The output of the last layers is fed to a 1000 way softmax which ultimately classifies the images under the 1000 different labels. Since the training dataset included 1.2 million images, a single GPU was not sufficient. Therefore, AlexNet spread the net across two GPUs. The GPU they used was GTX 580 which had 3GB of memory. They exploited the parallelism offered by the GPU by putting half the neurons in each GPU and letting them communicate only in certain layers. They found the training time to be slightly increased with two GPUs instead of one.

They also used local normalization in addition to ReLU to add to the generalization in certain layers. Denoting by $a^i_{x,y}$ the activity of a neuron computed by applying kernel *i* at position *(x,y)* and then applying the ReLU nonlinearity, the response normalized activity $b^i_{x,y}$ is given by the expression

$b^i_{(x,y)} = a^i_{(x,y)}/(k+\alpha \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)} (a^j_{(x,y)})^2)^\beta$ where the sum runs over n adjacent kernel maps at the same spatial position and N is the total number of kernels in the layer. The

ordering of kernel layer is arbitrary and determined before training begins. This sort of response normalization implements a form of lateral inhibition inspired by the type found in real neurons, creating competition for big activities among neuron outputs computed using different kernels. The constants k, n, , and are hyper-parameters whose values are determined using a validation set; they used $k = 2$, $n = 5$, $\alpha = 10^{-4}$ and $\beta = 0.75$.[1]

CHAPTER 3
METHODOLOGY

3.1 Software Development Model

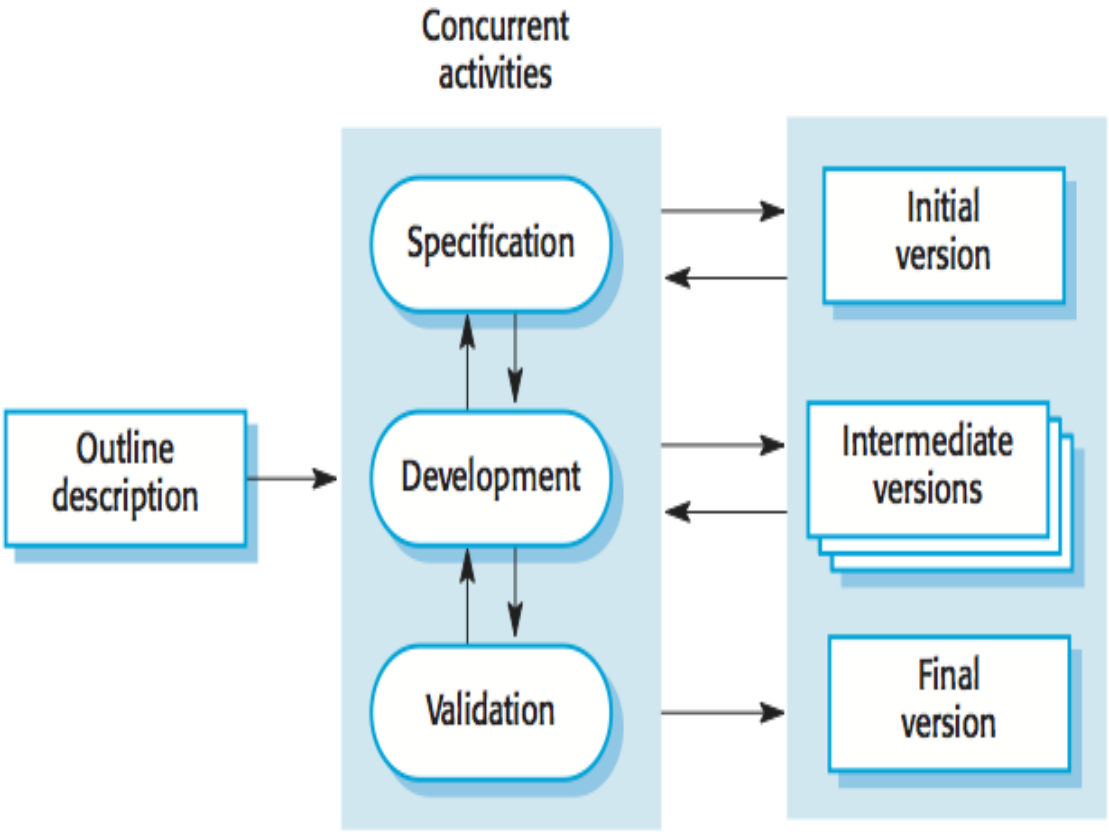

Figure 3.1: Incremental Model of Software Development[2]

The development model to be used is the incremental model of software development. Incremental model is a process of software development where requirements are broken down into multiple standalone modules of software development cycle. Each iteration passes through requirements, design, coding and testing phases. And each subsequent release of the system adds function to the previous release until all the designed functionality has been implemented. So the characteristics of this model include:

1. Development is broken down into many mini developmental works
2. Partial systems are built and functionality subsequently added
3. High priority requirements are tackled first

3.2 Convolutional Neural Network

3.2.1 Algorithm

An image can be considered a matrix of pixel values. Along with pixels an image also includes channel which is a standard term used to refer to a certain component of an image. A standard image consists of three channels red, blue and green. So an image can be considered as a stack of three matrix of pixels. In convolution step a filter is passed through an image. A filter is a smaller matrix of pixels which is passed through an image and the pixels in the image are multiplied with the filter. Once the multiplication is completed the filter is then shifted according to the stride value. If the stride value is 1 then the feature map is shifted by one pixel, if 2 then by two pixels and so on. The multiplication described above is then repeated all over the image. Different filters are used to extract different feature maps from the image. For example, the filter for edge detection and sharpness are different. The number of filters used in convolution step is called depth. If three different filters are used, then it can be visualized as a stack of three matrices and hence the depth is three. The value of the feature map is learned by the network itself although parameters such as number of filters, filter size and the network architecture need to be specified.

The operation of convolution layer in mathematical terms is as follows:

1. Accepts a volume of size $\boldsymbol{W_1 * H_1 * D_1}$
2. Requires four hyper parameters
    - Number of filters **K**
    - their spatial extent **F**
    - their stride **S**
    - the amount of zero padding **P**
3. Produces a volume of size $\boldsymbol{W_2 * H_2 * D_2}$ where:
    - $W_2 = (W_1 - F + 2P)/S + 1$
    - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e width and height are computed equally by symmetry)

- $D_2 = K$

4. With parameter sharing, it introduces $\boldsymbol{F \cdot F \cdot D_1}$ weights per filter. for a total of $\boldsymbol{(F \cdot F \cdot D_1) \cdot K}$ weights and **K** biases
5. In the output volume, the **d**-th depth slice (of size $\boldsymbol{W_2 * H_2}$) is the result of performing a valid convolution of the **d**-th filter over the input volume with a stride of S, and offset by **d**-th bias.

Common setting of the hyperparameters is $\boldsymbol{F = 3}$, $\boldsymbol{S = 1}$ and $\boldsymbol{P = 1}$.[3]

After the convolution step the ReLU operation is used. ReLU works on each element and replaces all the negative pixel values in the feature map by zero. The result of convolution step is linear so the purpose of ReLU is to introduce non-linearity as all the images in the world are non-linear. The output of ReLU is given by: $\boldsymbol{Output = max(zero, input)}$

The next step is the pooling step. This step reduces the dimensionality of each feature map retaining the relevant information. The best result can be obtained by using max pooling. In max pooling a spatial neighborhood is defined and the largest value from that neighborhood in the rectified feature map is taken thus reducing the dimensions of the feature map. Hence, the overall parameters and computations in the network can be reduced.

The fully connected layer is a traditional multilayer perceptron that uses softmax activation. This is the layer in which the classification takes place. The output of this layer is the probabilities for each class with the sum being equal to 1. The Softmax function takes a vector of arbitrary real-valued sources and squashes it to a vector of values between zero and one that sum to one.

3.2.2 Architecture

The most common form of a ConvNet architecture stacks a few CONV-RELU layers, follows them with POOL layers, and repeats this pattern until the image has been merged spatially to a small size. At some point, it is common to transition to fully-

connected layers. The last fully-connected layer holds the output, such as the class scores. In other words, the most common ConvNet architecture follows the pattern:

$$INPUT \rightarrow [[CONV \rightarrow RELU]*N \rightarrow POOL?]*M \rightarrow [FC \rightarrow RELU]*K \rightarrow FC$$

where $*$ represents repetition and POOL$?$ represents an optional pool layer.[3]

3.2.3 Dataset

The dataset to be used will be CIFAR-10 dataset. It consists of 60000 32X32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

CHAPTER 4
EXPECTED OUTPUT

4.1 Work Completed

Since the project is mostly an algorithm implementation, the weeks following the proposal defense has been spent researching how to implement this algorithm. The convolutional neural network involves complex computations which needed to be understood for implementation. The intervening weeks following the last defense were spent studying lectures on convolution neural networks. Furthermore, implementation of this algorithm required learning python programming language for machine learning. So a certain amount of time has been spent familiarizing with this language. Also the dataset for training the network has been collected.

4.2 Work Remaining

The research has moved further, yet is not complete. Also the implementation of the system still remains. The coding phase of the system is expected to begin in the coming weeks. Once that is begun, the subsequent testing of the intermediate modules will also start.

4.3 Problem Faced

The majority of the problems faced had to do with understanding the purpose and results of the convolution layer. Actually visualizing the effects of the convolutional network was particularly tough. Progress was made in this regard by following the lecture series provided by Standford University in this topic.

4.4 Work Schedule

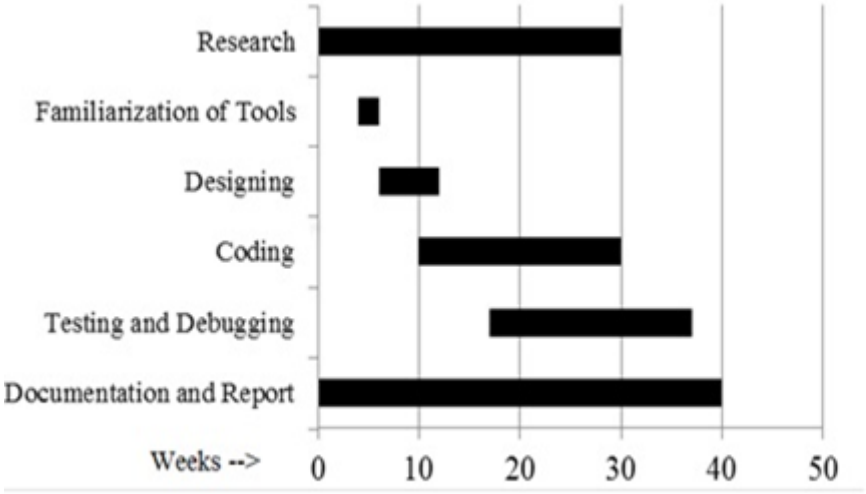

Table 4.1: Gantt Chart

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[2] I. Sommerville, *Software Engineering*, 9th ed. Boston: Addison-Wesley, 2011. [Online]. Available: https://ifs.host.cs.st-andrews.ac.uk/Books/SE9/

[3] "Cs231n convolutional neural networks for visual recognition," Cs231n.github.io, 2018. [Online]. Available: https://cs231n.github.io/convolutional-networks/ [Accessed: 13-Jan-2018].