International Journal of Engineering & Technology, 7 (2.24) (2018) 541-544

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Optical Compute Engine Using Deep CNN

Zainab Zaveri[1], Dhruv Gosain[2], Arul Prakash M[3]

[1,2]Student, [3]Assistant Professor OG
[1,2,3]Department of Computer Science, SRMIST, Chennai, India
[*]Corresponding Author Email: zainab_aziz@srmuniv.edu.in

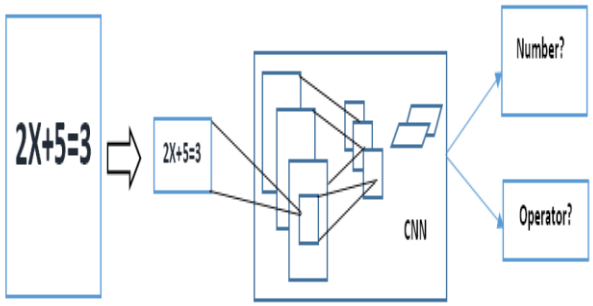
Abstract

We present an optical compute engine with implementation of Deep CNNs. CNNs are designed in an organized and hierarchical manner and their convolutional layers, subsampling layers alternate with each other, thus the intricacy of the data per layer escalates as we traverse in the layered structure, which gives us more efficient results when dealing with complex data sets and computations. CNNs are realised in a distinctive way and vary from other neural networks in how their convolutional and subsampling layers are organised. DCNNs bring us very proficient results when it comes to image classification tasks. Recently, we have understood that generalization is more important when compared to the neural network's depth for more optimised image classification. Our feature extractors are learned in an unsupervised way, hence the results get more precise after every backpropagation and error correction.

Index Terms: *Deep Convolutional Neural Network, feature extractor, R-CNN, SVNH*

1. Introduction

In the past, plenty of equation solving programs have been developed but they have always been short of in some way or the other, be it in terms of accuracy or computational time thus either of them had to be compromised .However now with the aid of Deep Convolutional Neural Network, we have been able to create an optical compute engine that can identify numbers and solve the equation with pinpoint accuracy. The task of identifying the numbers is a laidback job for humans but for an automated system it's much more different and complicated as it needs to be trained on how to catalogue various items and eliminate the unwanted information. Our program can understand diverse fonts and handwritings of digits as we have tested and trained our model on the vast Street View House Numbers (SVNH) data set. D-CNN's application is potent because it uses manifold layers of nonlinear image processing for feature mining, transformation and pattern recognition. The way our paper is systematized is as follows: Our algorithm comprises of two models, one for recognition and the next one for computation. The recognition model uses DCNNs to detect the various numbers and symbols used in equations. The computational model uses the concept of linear regressions and forward propagation to analyze and compute the final results to give a precise answer to the problem.



Input Equation Compare CNN features Classify

Fig.1: Basic architecture of a Convolution Neural Network

2. Recognition

2.1 Deep CNN

The foundation of CNN was laid by the research of D.H Hubel and T.N Wiesel. According to their study the mammalian cortex visions all the images akin where a group of neurons get activated. Thus came into picture the programmed form of the visual cortex "CNN". It is based upon three important traits of the cortex namely local connection, layering and spatial invariance.
CNN follows a hierarchal layered structure whose granularities increase from the lowest to the uppermost one. This is free of the size, alignment and the contrast of the input.

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

2.2 Architecture

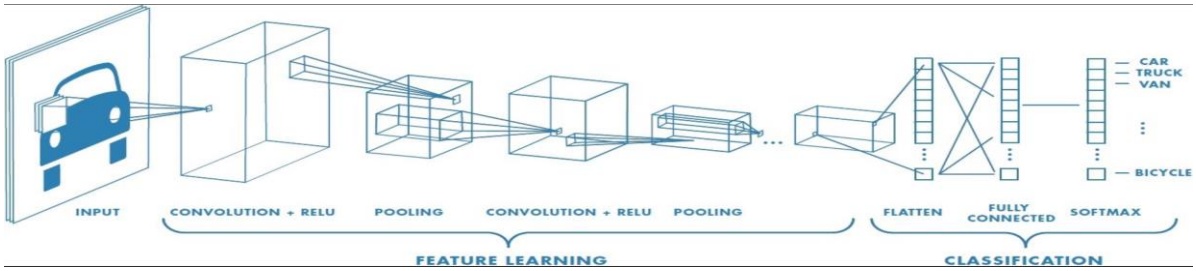


Fig.2: Detailed Architecture of the CNN [9]

The architecture can be roughly divided into Learning and Classification. The first part is the Feature Learning where feature are obtained from the input image. The second part is the classification where similar items are grouped. Usually there are at least three blocks of feature learning wherein each block comprises of Convolution, ReLu (Activation function) and the pooling. After this comes the classification phase where we flatten the content, the combination of these components gives us the Fully Connected layer and then a Softmax function is used to determine the probability of the weight and thus classify the item accordingly giving us the exact probability of the existence of the digits/symbols. The architecture is a feed-forward NN as the output of the previous layer becomes the input of the subsequent one.

A. Convolution: The main function of the convolution operator is to obtain the required features from the image. Receptive field is a part of the image where we apply the convolution operation (a feature window with a determined stride) and slide it all over the image. Thus we apply dot product for the weights in the layers and every part of the matrix in iterations. Each image is a 3-D matrix of pixels whose values vary from 0-255. Image has length, width and depth. Depth contain RGB channel but in case of gray scale image it has only 2 channels.

B. *ReLu:* This operation works on each pixel individually and converts the negative values to zero and thus normalizes the matrix. The purpose of Rectified Linear unit is to make our program nonlinear since convolution makes it linear.

Output=Max (zero, Input) (1)

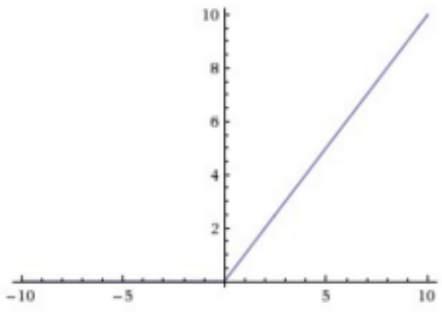


Fig.3: Graphical representation for ReLu Activation [7]

C. *Pooling:* Pooling is also known as subsampling or down-sampling. We can retain the vital information by minimizing the dimension of feature maps. In Max Pooling we take the highest value of the matrix rather than the average value. Since we take the highest value from each stride the overall dimension of the matrix gets reduced but the information that is significant is not lost.

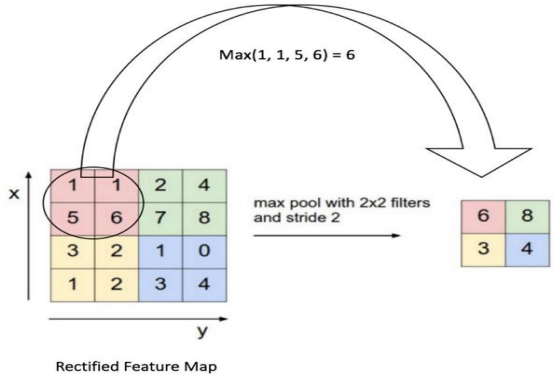


Fig.4: Figure depicting implementation of filters and stride in max pooling [10]

D. *Fully Connected Layer:* It executes operations of the higher order in the NN. The convolution layer produces complex features in data and the subsampling layer normalizes the weights, implementing a fully connected layer which is an easy and effective way of understanding those features.

E. Weights*:* Weights are the strength of network existing among the units. If the weight from unit A to unit B has higher value it means that A has a more significant value than B .Weights can thus be defined as a set of filter coefficients, defining an image feature. The units in higher layer get their input from lower layer rather than the pixels. So, the incoming weights are more like 'preferred input patterns'.

F. Softmax Function: It's a way of forcing the outputs of neural netwrok to sum to one so it can represent probability distribution over descrete mutually exclusive distributed alternatives.If we trying to assign labels to mutually exclusive class labels then the output should always sum to 1, but the network cannot derive this result.Hence we force the outputs to represent an probability distribution across discrete alternatives.Each neruron receives the total input accumulated from the layer below.The sum of all the Y's must be 1,ie.,the value of Y can be either 1 or 0 .Thus we force the Ys to represent the probability distrribution over mutually exculsive alternatives by using the softmax equation.

$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\top \mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\top \mathbf{w}_k}}$$

G. Categorical Cross Entropy Function: This loss function is used to measure the average figure of bits that are required to derive an event made from a set between two distributions t and y. The use of Cross Entropy function has been proven to increase efficiency along with use of the Softmax function. Our aim thus is to reduce the loss to increase proficiency.

The Cross Entropy function is as follows:

$$Total\ Loss = -\sum_{i=1}^{l} t_i log\,|y_i\,|$$ (1)

H. Dataset: We train our model on the SVNH data set and the handwritten data sets containing math symbols. We first need to downgrade the images in MATLAB and then write a python script to slice the images as each image in the data set consists of 3 or more numbers and symbols. After cleaning the data sets, we train our model by linking it and learning from this data that we have gathered. We use an array to convert our 3D integer pixels array into a float array and perform required calculations to attain a relative value between 0 and 1.Hence training our model on similar parameters and values.

3. Computation

After the recognition of the numbers and the operators, the computational phase starts.
Backward propagation is a mathematical expression for solving the derivative $\partial C/\partial w$ and shows us the dependency of the cost function on the weights or bias.
*III.A. Forward Pass***:** One of the oldest Neural Network architecture used till date are the Feed Forward Neural Network and they are used for task of regression and classification. The Neural Network that is used is made up of multiple layers. Every neuron from one layer is connected with the output of each and every neuron from other layer. This is main feature of a fully connected NN and every neuron has its personal set of weights and biases. To be able to differentiate the vectored neurons AAA and weights WWW, we add the layer they belong to as superscript. For getting this result we add a new row of weights for each neuron in a layer. The calculation for forward pass is as follows:

Forward Pass

All Layers (Except the final layer)

Input=>x

Weights=>w

Bias=>b

Output=>y

$$y_in = wx + b \tag{1}$$

$$y = \begin{cases} y_in, & y_in > 0 \\ 0, & otherwise \end{cases} \tag{2}$$

Final Layer

For every neuron with index i

$$(y_in_i) = w_i x + b_i \tag{3}$$

Activation (Softmax)

$$y_i = (e)^{y-in_i} / \sum_{i=1}^{I} (e)^{y-in_i} \tag{4}$$

A. Calculating Total Error:

Using the Cross Entropy loss function we get the average of bits from an event formed by the probability distribution p and q

Loss Evaluation

Final Output=>y
Target =>t

$$Total\ Loss = -\sum_{i=1}^{I} t_i \log |y_i| \tag{5}$$

B. The Backward Pass:

The idea here is to update each weight so that its value is as close as possible to the target value. The closer the value the lower the error that is generated. In backpropagation, we back propagate the error. It is the cost that we estimate by comparing the calculated output and the correct target output, which we then use to change the hyper parameters. The efficiency of matrix to vector multiplication is much higher than the matrix to matrix multiplication. Thus we change the value of the hyper parameters to with experience to attain maximum efficiency.

Backward Pass

For Every weight w

$$\Delta w = \frac{\partial Total\ Loss}{\partial w} \tag{1}$$

$$w_{new} = w_{old} - (Learning\ Rate * \Delta w) \tag{2}$$

4. Results

Our results depict and simulate that, the deep convolutional neural network can give world-class results on our complex data set using unsupervised learning. The network's performance will fall even if one layer is deleted from the stack of the layers we have used in our model. If we remove any of the layers in between the network it will result in a minor loss which will eventually affect the accuracy of our result. So the depth of the network really is important for achieving relevant results. The more we run our network model, the better and deeper it trains itself to give us more accurate and efficient results. Our proposed system has many applications in various industries which can speed up the calculations performed manually in real time, be it the banking sector, finance, or even the education.

5. Conclusion

We presented optical compute engine using D-CNNs. Since SVNH dataset has been used for the recognition of number the accuracy is top notch and can not only recognize printed numbers but also hand written numbers of different handwritings. Principal advantages include generalization capabilities, great flexibility and speed. Using Backpropagation and Feed Forward Linear Regression we were able to compute various mathematical equations with accuracy in real time. Even though there are a few programs available in the market but some constraints limit them, be it performance or efficiency .Currently, the algorithms used for the recognition and computation using deep CNN are the best there is in the market. The various hyper parameters such as the image filters, image size, number of pooling layers, number of maps per layer, stride sizes, are adaptable to any particular application. The results produced are increasing the bars for the benchmarks for recognition and computation speed and accuracy.

6. Acknowledgement

This work was possible by Guidance and help of our Project Guide, Mr. Arul Prakash and the dedicated support of our friends and families. We are obliged by the department for giving us steadfast motivation and support.

References

[1] Dan C.Ciresan, Ueli Meirer,Jonathan Masci,Luca M.Gambardella,Jorgen Schmidhuber, “Flexible,High Performance Convolutional Neural Networks for Image Classification,” IDSIA,USI andSUPSI.

[2] Alex Krizhevsky,Ilya Sutskever,Geoffrey E.Hinton,”ImageNet Classification with Deep Convolutiinal Neural Networks”, University of Toronto.

[3] Andeas Kolsch, Muhammad Zeshan Afzal ,Markus Ebbecke,Marcus Liwicki,“Real-Time Document Image Classification using Deep CNN and Extreme Learning Machines,” in 2017 14th IAPR International Conference on Doocument Analysis and Recognition.

[4] URL:”https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/.”

[5] Ross Girshick, Jeff Donahue,Trevor Darrell, Jitendra Malik,”Rich feature heirarchies for accurate object detection and semantic segmentation”

[6] URL: “ http://cs231n.github.io/neural-networks-1/”

[7] Max Jaderberg,Karen Simonyan,Andrew Zisserman,Koray Kavukcuoglu,”Spatial Transformer Networks”.

[8] Christian Szegedy,Wei Liu,Yangqing Jia,Pierre Sermanet,Scott Reed,Dragomir Anguelov,Dumitru Erhan,Vincent Vanhoucke,Andrew Rabinovich,”GoogleNet:Going Deeper with Convolution”,ILSVRC 2014

[9] URL:”https://hackernoon.com/deep-learning-cnns-in-tensorflow-with-gpus-cba6efe0acc2”

[10] Ian J.Goodfellow, Jean Pouget-Abadie,Mehdi Mirza, Bing Xu,David Warde-Farley,Sherjil Ozair,Aron Courville,Yoshua Bengio,”Generative Adversial Nets”,University of Montreal.

[11] URL: “https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks”

[12] Andej Karpathy,Li Fei-Fei,”Deep Visual –Semantic Allignments for Generating Image Descriptions”,Stanford University