Convolutional Neural Network (CNN) for Image Classification

Yibo Cheng
Department of Computer Science
(The Jonsson School)
The University of Texas at Dallas
Richardson, TX
yxc190039@utdallas.edu

Mo Han
Department of Computer Science
(The Jonsson School)
The University of Texas at Dallas
Richardson, TX
mxh190001@utdallas.edu

Minchao Zhu
Department of Computer Science
(The Jonsson School)
The University of Texas at Dallas
Richardson, TX
mxz190002@utdallas.edu

Abstract— **Science and technology are developing in the fast pace. Algorithm and theoretical foundation, technology starts to lead in people's daily life and transforms the way it is. One of the common applications is using convolutional neural network to classify images or objects based upon their features. Convolutional Neural Network (CNN) has excellent ability to develop an internal representation of a two-dimensional image, making it the preferred network for image classification. In this paper, a classical deep learning algorithm, convolutional neutral network , was implemented to realize image classification. The team built up a convolutional neutral network from the scratch. Stages include preprocessing images, convolutional layer, activation layer, pooling layer, fully connected layer and forward and backward propagation. Meanwhile, effects brought by parameters, like convolutional matrix, batch size, were studied and compared in terms of network efficiency, accuracy and generality. The testing accuracy of CNN is around 30%~60%, which is acceptable as there are few issues the team does not intend to tackle down at this time. Open issues include overfitting and underfitting, activation function, image resolution loss, etc. Potential improvement on classification accuracy could be realized if these problems were addressed and will be left for the future tasks.**

Keywords—CNN, Image Classification, Tuning, Convolutional Layer

I. INTRODUCTION & BACKGROUND

Science and technology are developing in the fast pace. Especially equipped with advanced computing hardware, algorithm and theoretical foundation, technology starts to lead in people's daily life and transforms the way it is. One of the common applications is using convolutional neural network to classify images or objects based upon their features[1].

Since 1943 when Warren McCulloch, a neurophysiologist, and a young mathematician, Walter Pitts, wrote a paper on how neutron might work, artificial neutral network has been studied all over the world[2]. By now, there are thousands of types of specific neural networks proposed by researchers as modifications or tweaks to existing models and even more are still emerging. These models can be categorized into three classes in general, Multilayer Perceptron (MLP), Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) [3]. Each class has its own design purpose and focus. CNN's ability to develop an internal representation of a two-dimensional image makes it the preferred network for image classification[4].

The convolutional neural network (CNN) was first proposed in 1960s[5]. Hubel and Wiesel proposed the concept of "receptive field" which observed that neurons were sensitive to moving edge on visual cortex cells of cats. Later in the 1980s, based on concept of "receptive field", Fukushima and Miyake proposed "neocognitron" which is regarded as the first implementations of CNNs[6][7]. However, due to lack of proper learning algorithm, CNN was not the main focus in the network. After that, researchers started to use multilayer perceptron to learn features and incorporated backpropagation (BP) algorithm[8]. However, since traditional BP neutral network would have series of problems requiring detailed study, the research on deep neutral network model was stopped. Until Hinton et al found that the artificial neural network with multiple hidden layers addresses those old issues and has great performance in feature learning, deep learning starts to re-gain attention and more and more sophisticated and accurate models were developed and used in daily practice, especially in the fields of OCR, autonomous drive, image recognition and analysis, social media, etc[9][10].

In this paper, a classical deep learning algorithm, convolutional neutral network (CNN), was implemented to realize image classification. The team built up a convolutional neutral network from the scratch. Stages include preprocessing images, convolutional layer, activation layer, pooling layer, fully connected layer and forward and backward propagation. Meanwhile, effects brought by parameters, like convolutional matrix, batch size, were studied and compared in terms of network efficiency, accuracy and generality.

II. TECHNIQUE & ALGORITHM

A. Convolutional Neutral Network (CNN)

Compared to other image classification algorithms, CNNs do not require extra work on preprocessing images and this means that they can learn the filters that have to be hand-made in other algorithms. Also, CNNs have advantage in dimensional reduction without losing learning features[11]. This could tremendously save computing time. Layers including convolution, pooling, activation and fully connection constitute the network and the sequence of each operation could be shuffled according to the need[12].

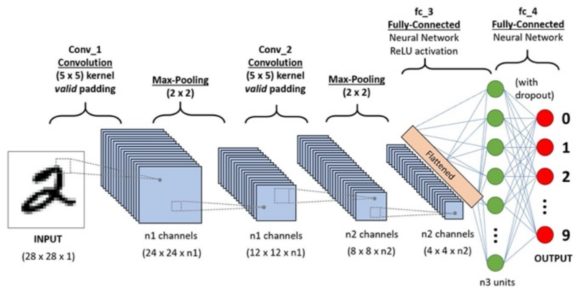


Fig. 1: A CNN sequence to classify handwritten digits.[13]

B. Convolutional Layer

First layer of CNN normally starts with a convolutional layer. A convolutional layer contains a set of filters whose parameters including height, weight, number of filters are determined through inputs[14]. In general, the height and weight of the filters are smaller than those of the input volume (in research, input is a 2D matrix image with 1 channel, gray image). Each filter is convolved with the input volume to compute an activation map. In specific, the filter is slid across the width and height of input with appointed stride and the dot products between the input matrix and filter are computed at every spatial position. After number of filter's iterations, the output of convolutional layer is obtained by stacking the activation maps of all filters along the depth dimension. For example, in the research, given the situation that input volume is a uniform a*a matrix, convolutional filter is chosen to be b*b matrix and each stride is assigned as c, the final output's shape after convolutional layer will be (a + c – b) * (a + c – b).

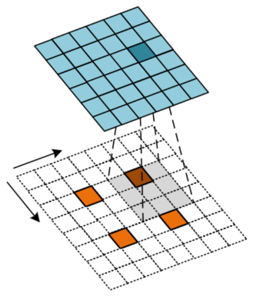


Fig. 2: Convolution process of transposed convolution layer.[15]

C. Pooling Layer

The purpose of pooling layer in the network is to progressively reduce the spatial size of representation to reduce the number of parameters and computation in the network. Normally, pooling layer will exist in-between convolutional layers and it takes a series input parameter including input volume, number of filters, height, width, stride and padding. Two main techniques in pooling are popular and widely used, average pooling and maximum pooling. In the research, maximum pooling is implemented for down sampling input volume. Max Pooling ensures a lower resolution version of an input signal is created and still contains large portion of key features, without fine detail that might not be as useful to the task.

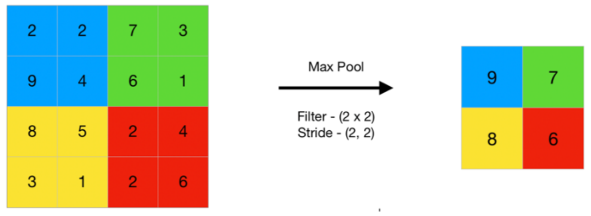


Fig. 3: Max Pooling Layer.[16]

D. Activation Layer

The function of activation layer in the network is to transform linear output from either convolutional layer or pooling layer to non-linear output so that the network is able to learn complex patterns in the data. Many activation functions are available and, in the research, rectified linear unit (ReLU) is implemented, which simply computes the function: f(x) = max (0, x).

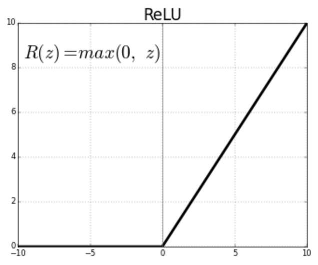


Fig. 4: ReLU Activation Function.[17]

E. Fully Connected Layer

The objective of fully connected layer in the network is used to classify the image into a target label. At this layer, it will receive output from either convolutional layer or pooling layer. The input will then be further flattened into a single vector of values, each representing a probability that a certain feature belongs to a label. [18][19]

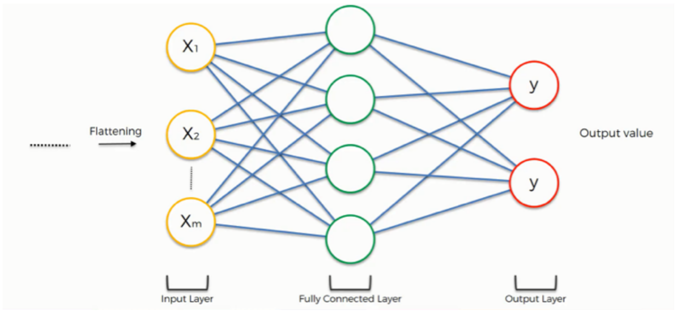


Fig. 5: Fully Connected Layer.[20]

Identify applicable funding agency here. If none, delete this text box.

III. Experiment Detials

The main purpose of experiment is to implement Convolutional Neural Network from the scratch and to get some practice with training it. In this paper, we only use Python, Numpy, and OpenCV to build up our own Convolution Neural Network without using any open source data flow engine.

A. *The Architecture of Convolutional Neural Networks*

Our Convolutional Neural Networks mainly contains Convolution Layer, ReLu Layer, Max Pooling Layer, Flatten Layer, FullyConnected Layer, and Softmax Layer. Among them, Convolution Layer, ReLu Layer, MaxPooling Layer can be combined as a small layer group. And this group can be duplicated many times in the architecture of CNN.

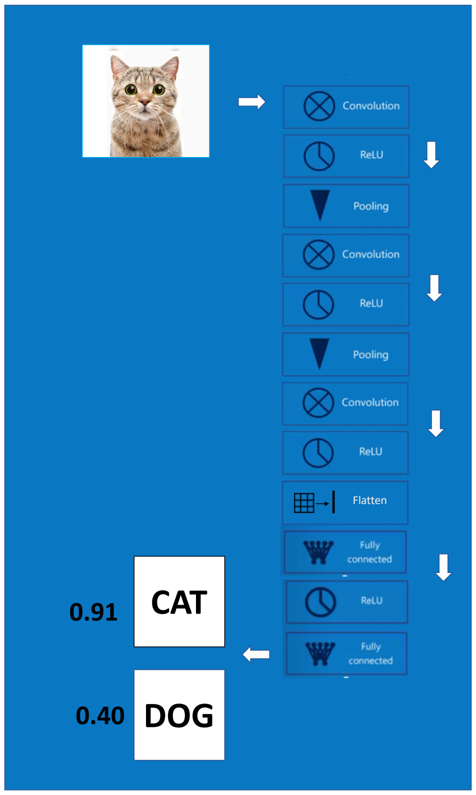


Fig. 6: The architecture of our Convolutional Neural Networks

Figure 6 shows a representative order of layers in the Convolutional Neural Networks we built up. The first Convolution layer has 6 filters with filter length 5. All Max Pooling Layers have a length of 2. The second Convolution layer has 16 filters with filter length 5. And the third Convolution layer has 120 filters with filter length 5. The input and output of each layer will change with the original size of images. Table1 shows the parameters of layers when the input size of image is 28*28.

B. *Purpose of experiment.*

The most important purpose of this project is to implement Convolutional Neural Networks successfully. By writing the codes of different layers and forward and backward functions in all the layers, we can get comprehensive understanding of how backpropagation algorithm works and how different layers influence the result. Then the team gets much practice with training them by using different input data, batch size, or filter number. In this process, we can gain a better understanding of how to make CNN work better.

Table 1: Parameters of layers

	Input	Num of filters	Matrix length	Padding	Learning Rate	Output
Conv	28*28	6	5	2	0.01	28*28*6
ReLu	28*28*6	/	/	/	/	28*28*6
MaxPool	28*28*6	/	2	/	/	14*14*6
Conv	14*14*6	16	5	0	0.01	10*10*16
ReLu	10*10*16	/	/	/	/	10*10*16
MaxPool	10*10*16	/	2	/	/	5*5*16
Conv	5*5*16	120	5	0	0.01	1*1*120
ReLu	1*1*120	/	/	/	/	1*1*120
Flatten	1*1*120	/	/	/	/	120
FullyConnect	120	/	/	/	0.01	60
ReLu	60	/	/	/	/	60
FullyConnect	60	/	/	/	0.01	2

In this paper, a simple classifier of cat and dog images was accomplished by the Convolutional Neural Networks. Totally, 4000 images of cat and 4000 images of dog have been used as training data. 1000 images of cat and 1000 images of dog have been used as testing data. Cat was classified as (0,1) and dog was classified as (1,0). All the images were changed to gray before training or testing.

IV. Result and Analysis

A. *Influence of distribution of training data*

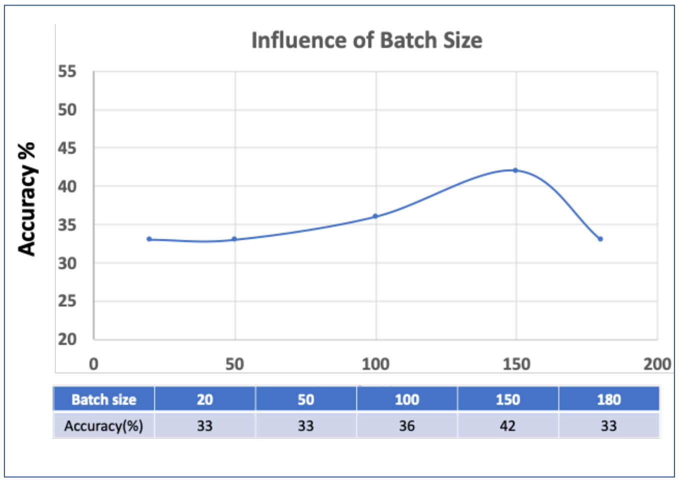
The distribution of training data is found to have obvious influence on the result. If CNN was trained using all the images of cat firstly, the testing results may be always CAT. If we reversed the order of training, that means training CNN using all the dog images firstly, all the results changed to DOG. Therefore, the order of images being used in the training process is very important.

The distribution of different classification should be as uniform as possible in the training dataset, which has been paid attention to in the following experiments.

B. *Influence of batch size*

Batch size means the number of training examples in one forward/backward pass. Since the forward/backward process is the kernel of neural network, batch size can obviously influence the training process and the result of testing. Generally, the batch size needs to be smaller than the number of all samples. If higher batch size is used in the training process, the experiment will need more memory space but the training process may cost much less time. If batch size is changed to be smaller, training surely requires less memory number. In the meanwhile, the estimate of the gradient may become less accurate. Therefore, choosing a suitable batch size also plays an important role in our training process.

In our experiment, we tried different batch sizes to train CNN and got the relation between batch size and testing accuracy. We chose five different values (20, 50,100, 150, and 180) as batch size and tried to find the best batch size roughly. According to the results in Figure 7, we can find that the accuracy becomes better with the increasing of batch size at first. The accuracy reaches the largest (42%) when the batch size is 150. Then accuracy begins to flow down after batch size becomes larger than 150. Hence, we choose 150 as our best batch size and use it in the following experiments.



Batch size	20	50	100	150	180
Accuracy(%)	33	33	36	42	33

Fig. 7: The change of accuracy with the increasing of batch size.

C. *Potential Influence of input size*

Besides batch size, we also pay attention to the influence of input size upon accuracy. To save the computation, we firstly chose a small input size of image, this is 28*28. Figure 8 shows the different conditions of input images. Before training, we preprocessed all the images to gray and with a size of 28*28. According to the images in Figure 8 , we can find that the image of gray and 28*28 has still maintained most of characteristic features of original image. By using this kind of input images, we trained CNN and the accuracy of testing was around 30%, which is surely not an ideal result. Then we tried to increase the size of our input image, which is more similar to full image. However, changing input size means that most of the parameters of convolutional layers need to be optimized again. By now, we have only realized the training of CNN using images of 28*28 or 56*56. The accuracy of testing is largely improved from 34% to 67% when we change the input size from 28*28 to 56*56. Therefore, large input size may benefit the training process. But our result still needs to be repeated and verified. If this conclusion is true, we plan to further increase the input size in the following experiment.

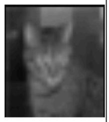
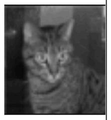
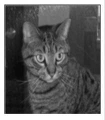
	original	gray	Gray 28*28	Gray56*56	Gray128*128
image					
Result (Accuracy %)	/	/	33.5	67	Continue...

Fig. 8: The relation between accuracy and image size.

V. Conclusion

Our results show that a complete and comprehensive convolutional neutral network (CNN) is able to be reproduced without using any official libraries in either TensorFlow or Python machine learning. The team follows few steps to accomplish the image classification tasks. First, the team preprocessed image datasets including 4000 images on cat and dog each. Preprocessing includes converting RGB images into gray images and utilizing Gaussian filtered edge detection technique prior to the CNN. Then, the team implemented convolutional layer, max-pooling layer, activation layer and fully connected layer algorithms along with forward and backward propagation. Experiments on tuning parameters in layers and sequencing layers were conducted to explore potential opportunity in improving result accuracy. The image classification accuracy the team obtained is around 30%. After optimizing the batch size of training process, the accuracy increases to 42%. We further increase the accuracy to 67% by increasing the input size and optimized the convolutional layer. Although the result still needs to be verified, the accuracy is promising to be increased. The current result is acceptable as there are few issues the team does not intend to tackle down at this time. Open issues include overfitting and underfitting, activation function, image resolution loss, etc. Potential improvement on classification accuracy could be realized if these problems were addressed and will be left for the future tasks.

References

[1] Xu L, Krzyzak A, Suen C Y. Methods of combining multiple classifiers and their applications to handwriting recognition[J]. IEEE transactions on systems, man, and cybernetics, 1992, 22(3): 418-435.

[2] Graves A, Liwicki M, Bunke H, et al. Unconstrained on-line handwriting recognition with recurrent neural networks[C]//Advances in neural information processing systems. 2008: 577-584.

[3] Lawrence S, Giles C L, Tsoi A C, et al. Face recognition: A convolutional neural-network approach[J]. IEEE transactions on neural networks, 1997, 8(1): 98-113.

[4] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.

[5] Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences[J]. arXiv preprint arXiv:1404.2188, 2014.

[6] Zhang X, Zhou X, Lin M, et al. Shufflenet: An extremely efficient convolutional neural network for mobile devices[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 6848-6856.

[7] Dong C, Loy C C, Tang X. Accelerating the super-resolution convolutional neural network[C]//European conference on computer vision. Springer, Cham, 2016: 391-407.

[8] Hu B, Lu Z, Li H, et al. Convolutional neural network architectures for matching natural language sentences[C]//Advances in neural information processing systems. 2014: 2042-2050.

[9] Li H, Lin Z, Shen X, et al. A convolutional neural network cascade for face detection[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 5325-5334.

[10] Jin K H, McCann M T, Froustey E, et al. Deep convolutional neural network for inverse problems in imaging[J]. IEEE Transactions on Image Processing, 2017, 26(9): 4509-4522.

[11] Lin X, Zhao C, Pan W. Towards accurate binary convolutional neural network[C]//Advances in Neural Information Processing Systems. 2017: 345-353.

[12] Cai Z, Fan Q, Feris R S, et al. A unified multi-scale deep convolutional neural network for fast object detection[C]//European conference on computer vision. Springer, Cham, 2016: 354-370.

[13] Karpathy A, Toderici G, Shetty S, et al. Large-scale video classification with convolutional neural networks[C]//Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2014: 1725-1732.

[14] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[15] https://www.researchgate.net/figure/Convolution-process-of-transposed-convolution-layer_fig4_340020703

[16] https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

[17] https://medium.com/@kanchansarkar/relu-not-a-differentiable-function-why-used-in-gradient-based-optimization-7fef3a4cecec

[18] Bearman A, Russakovsky O, Ferrari V, et al. What's the point: Semantic segmentation with point supervision[C]//European conference on computer vision. Springer, Cham, 2016: 549-565.

[19] Shafiee A, Nag A, Muralimanohar N, et al. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars[J]. ACM SIGARCH Computer Architecture News, 2016, 44(3): 14-26.

[20] https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection