Research on Classification and Recognition of Object Image Based on Convolutional Neural Network

Xianlun Wang*, Zuomin Yang*
College of Electromechanical Engineering Qingdao University of Science and Technology Qingdao, China

*Corresponding author e-mail: 1406253777@qq.com, *xlwang@qust.edu.cn

Abstract. Image recognition and classification using a convolution neural network is an important application of image processing. How too reasonably to design the convolutional layer of the convolutional neural network, the number of hidden layers and optimize the parameters of the convolutional neural network to ensure high accuracy and efficiency in image recognition and classification are an extremely important part. The core of this thesis is the model design and parameter optimization of deep convolutional neural networks. This paper mainly designs, implements, optimizes and adjusts the model structure of the convolutional network of Tensor Flow framework platform. We redesigned the convolutional neural network model to a depth of 19 layers and used two data sets for training, testing and parameter optimization. Experimental results show that the convolution neural network models presented in this paper is superior to other neural network models in accuracy and efficiency of image recognition and classification, and has a good guiding role in solving practical engineering problems.

1. Introduction
For convolutional neural networks, it is a kind of neural network with deep structure and containing convolution calculation, which is one of the representative algorithms of deep learning [13, 14]. The study of convolutional neural networks began in the 1980s and 1990s, with TDNN [46] and LeNet-5[28] being the first convolutional neural networks. With the continuous development of the theoretical discipline of deep learning, convolutional neural networks have also developed rapidly. The convolutional neural network structure can be roughly composed of three parts, which are called the input layer, hidden layer, and output layer. Taking LeNet-5 as an example, the order of the three common structures in the hidden layer is usual: input → convolution layer → pooling layer → convolution layer → pooling layer → fully connected layer → output. As shown in Figure 1:
The function of the input layer is to input the reprocessed data onto the neural network model. Specifically, the data is input into the hidden layer, the convolution layer, the pooling layer, and the fully connected layer. The function of the convolution layer is to extract features of the input data. The convolution layer contains multiple convolution kernels, and each element of the convolution kernel corresponds to one weight and one deviation. Like a neural network neuron, the size of the convolution kernel is manually set. The size of the commonly used convolution kernel is 1x1, 3x3, and 5x5. In this paper, multiple convolution kernels are used, and the characteristics of size extraction of different convolution kernels are different. The purpose is to improve the ability to extract features. The activation functions as the convolution layer plays an important role in the ability of the model to enhance learning. Deep learning activation functions include Sigmoid function, Tanh function, pReLU function, ELU function, Maxout function, Leaky ReLu function, ReLu function and so on [15, 16, 20, 30, 33, 4, 26, 21, 5, 9, 17, 36, 37, 8]. Commonly used deep learning activation functions are sigmoid function, Tanh function, and ReLu function. The Sigmoid and Tanh have a very obvious disadvantage: when z is large or small, the derivative is almost zero, which is almost impossible to update in the gradient descent optimization. However, the most popular in machine learning is the correction of the linear unit ReLu. The ReLu function is shown in the Figure 2:
From the figure above, it is not difficult to see that the ReLu function is actually a piecewise linear function, which changes all the values less than zero into zero, while the values greater than zero remain unchanged. This operation is known as unilateral suppression from an academic perspective. This operation can make the neurons in the neural network sparsely active. Especially in the deep neural network model (such as CNN) plays an extremely important role.

In this paper, the entire model is activated using the ReLu function. After the feature map was extracted, the main function of the pooling layer are to perform feature selection and information filtering. The pooling operation in the convolution neural network not only reduces the dimensional of the data, but also improves the distortion tolerance of the model’s small amount of translation, scaling, and rotation in the input sample. At present, the most commonly used pooling methods are general pooling, overlapping pooling, empty pyramid pooling, maximum pooling method, average pooling method, etc.[24, 25]. This article uses the maximum pooling method. In this paper, behind the pooling layer are two fully connected layers, each neuron in the fully connected layer is fully connected with all neurons in the previous layer. The fully connected layer can integrate local information on class discrimination in the convolution layer or the pooled layer. To improve the performance of convolution neural networks, in this paper, the excitation function of each neuron in the two-layer fully connected layer uses the ReLu function. The output value of the last layer of the fully connected layer is passed to an output. This paper uses softmax-linear logistic regression for classification. The final layer of the convolution neural network model is also known as the softmax-linear layer. For a specific classification task, it is very important to choose a suitable loss function. The loss functions as deep learning guides the network parameter learning by the error back propagation generated by the prediction sample and the real sample marker. In deep learning, the commonly used loss functions are hinge loss function, ramp loss function, large interval cross-entropy loss function, central loss function, loss function of regression task, Tukey’s weight loss function, cross-entropy loss function. The cross-entropy loss function was also called the softmax loss function [45, 38, 10, 39, 22, 32, 7, 18]. It is the most commonly used classification target loss function of convolution neural networks, so the cross-entropy loss function was used in this paper. For the design of convolution neural network model, different comprehensive analysis of different application scenarios is needed. This paper mainly classifies different objects, adopts two data sets, one is a five-category flower data set, and the other is a five-category object data set with input sizes set to 64x64 RGB images.
2. Related work

2.1. Structure of the Paper

As shown in Figure 3. In this paper, the training process used roughly the same as that of other neural network training. The focus is on the structural design of the convolution neural network. Different data sets are placed under the same neural network structure model for training and testing, tuning, and optimization. As can be seen from the above flow chart, the first step is to obtain data, and the data is the basis. Without data, other steps cannot be performed. The second step is to process the data to remove unwanted noise and factors that harm the classification target so that the data can be better adapted to the neural network model to achieve better results. The third is that this paper reconstructs a convolution neural network models. The fourth is to put the reprocessed data onto the convolution neural network model for model training. The fifth is that after the training model is completed, the model is evaluated, and the parameters such as the learning rate in the model are adjusted and optimized to achieve the best purpose. The sixth is to save the trained model and test the trained model to realize the function of image classification and recognition.

2.2. Development Environment Construction

In this paper, the development of the convolution neural network models is based on TensorFlow–CPU. TensorFlow [2] is a symbolic mathematics system based on data flow programming. It is widely used in the programming implementation of various machine learning algorithms. TensorFlow is developed and maintained by Google’s artificial intelligence team Google Brain. Its predecessor is Google’s neural network algorithm library Disbelief [1]. During the environment construction process, the computer used in this program is ubuntu16.04, first install anaconda3, python3.7, TensorFlow, PIL, etc.

Figure 3. Overall structure diagram
2.3. Data Sets

2.3.1. Introduction to data sets. This article uses the flower photos data set and the yang data data set. The yang data data set is our dataset. The flower photos data set was created by Google’s research department and can be downloaded from http://download.tensorflow.org/example. Images/flower_photos.tgz. In this paper, the two data sets are processed and input into the designed convolution neural network model for training. The flower photos data set includes a total of 3,672 images of 5 flower types: daisies, dandelions, roses, sunflowers, and tulips. There are approximately 600-900 training sample images for each of these categories. The yang data data set includes five categories of objects: a backpack, a computer monitor, a computer mouse, a Segway, and a self-propelled-lawn-mower, for a total of 494 images. The main distribution of the data set is shown in Figure 4 and Table 1:

![Figure 4. Ten different objects](image)

| Yang-data data set                      | backpack | Computer-monitor | Computer-mouse | Segway | Self-propelled-lawn-mower |
|----------------------------------------|----------|------------------|----------------|--------|---------------------------|
| Number of objects                      | 100      | 100              | 94            | 100    | 100                       |

| Flower-photos data set                 | roses    | tulips           | dandelion     | sunflowers | daisy                     |
|----------------------------------------|----------|------------------|---------------|-------------|---------------------------|
| Number of flowers                      | 641      | 799              | 898           | 699         | 633                       |

2.3.2. Data set processing. The two methods of data collection used in this paper use the same data processing method. The total number of samples were generated: yang data data set generates 500 images with labels, and flower photos data set to generate 4000 images of labels. Since the original data set has different image sizes, the original image needs to be converted to 64x64 size. The detailed steps are as follows:

The first step is to create TFRecords data. Why do you want to make Records data? The TFRecord data file is a binary file that stores images and tags in a unified manner. It can make better use of memory. This article runs under the TensorFlow framework, it can be quickly copied, moved, stored, read, etc. in TensorFlow. TFRecord automatically assigns the same label to each class based on the input file class. In the data set of this paper, there are five categories, 0, 1, 2, 3, and 4. After running the code, a file in the form train.tfrecords is generated.

The second step is to read the tfrecord file. After the TFRecord file is created, the file is read into the data stream. In this paper, we want to check whether the classification is wrong, or in the subsequent network training process can monitor, output pictures, to observe the results of the classification and other operations, so we have read the TFRecord picture from the stream, and then save.

The third step is to preprocess the data. The third step is to process the data. First, we get all the image path names, store them in the corresponding list, and paste the labels into the label list. Second, the
generated image path and the tag list are scrambled. Next, batch processing, passing the generated list of get-batch (), converting the type, generating an input queue, and then reading the image of the queue of tf. Read-file (). Again, the image is decoded, and different types of images can’t be mixed, either with the jpg type or with the png type. After that, data processing is performed to rotate, scale, crop, and normalize the image, so that the calculated model is more robust. Finally, generate a batch. The processed data is input to the designed convolution neural network model structure.

2.4. Structural Design of Convolution Neural Network Model

The convolution neural network model structure designed in this paper consists of 8 layers of convolution layer, 6 layers of maximum pooling layer, 2 layers of Google Net [6] based convolution layer Mixed1 and Mixed2, 2 layers of fully connected layer, and 1 layer of Softmax classification layer. Initially, the preprocessed 64x64 image is fed to the convolution neural network, the 3x3 convolution kernel, and the RGB is 3. The convolution neural network padding='SAME' designed in this paper represents the graph of the padding after padding and the original image size is the same. The activation function ReLu (), stride is all 1, and the detailed convolution neural network model layer structure and parameter information are shown in Table 2.

| type         | Patch size/stride | Input size     | Output size     |
|--------------|-------------------|----------------|-----------------|
| Conv1        | 3x3/1             | 64x64x3        | 64x64x64        |
| Conv11       | 3x3/1             | 64x64x64       | 64x64x64        |
| Pooling1-1lrn| 3x3/2             | 64x64x64       | 32x32x64        |
| Conv2        | 3x3/1             | 32x32x64       | 32x32x64        |
| Pooling2-1lrn| 3x3/2             | 32x32x64       | 16x16x64        |
| Conv2d-1a-3x3| 3x3/1             | 16x16x64       | 16x16x64        |
| Conv2d-2a-3x3| 3x3/1             | 16x16x64       | 16x16x64        |
| Pooling3-1lrn| 3x3/2             | 16x16x64       | 8x8x64          |
| Mixed1       |                   |                |                 |
| Conv2d-1a-3x3| 3x3/1             | 8x8x64         | 8x8x64          |
| Conv2d-2a-5x3| 3x3/1             | 8x8x64         | 8x8x64          |
| Conv2d-2a-3x3| 3x3/1             | 8x8x64         | 8x8x64          |
| Conv2d-2a-3x3| 3x3/1             | 8x8x64         | 8x8x64          |
| Pooling4-1lrn| 3x3/2             | 8x8x64         | 4x4x256         |
| Mixed2       |                   |                |                 |
| Conv2d-1a-3x3| 3x3/1             | 4x4x256        | 4x4x256         |
| Conv2d-2a-3x3| 3x3/1             | 4x4x256        | 4x4x256         |
| Conv2d-1a-3x3| 3x3/1             | 4x4x256        | 4x4x256         |
| Conv2d-2a-3x3| 3x3/1             | 4x4x256        | 4x4x256         |
| Pooling5-1lrn| 3x3/2             | 4x4x256        | 2x2x1024        |
| Conv2d-11a-3x3| 3x3/1             | 2x2x1024       | 2x2x1024        |
| Conv2d-21a-3x3| 3x3/1             | 2x2x1024       | 2x2x1024        |
| Pooling6-1lrn| 3x3/2             | 2x2x1024       | 1x1x1024        |
| FC-loca13     | liner             | 1x1x1024       | 128             |
| FC-loca14     | liner             | 128            | 128             |
| Softmax-linear| classifier        | 128            | n-classes       |

Table 2. Structural Design of Convolutional Neural Network Model

Convolutional neural network model structure is shown in Figure 5. The figure is generated by the visualization tool Tensor Board, which represents the convolutional neural network model structure designed in this paper.
Figure 5. The neural network model structure consists of the following parts: data processing part, convolutional layer, pooling layer, Mixed1(Fig.7) and Mixed2(Fig.8), fully connected layer, classification layer, and finally calculates the loss and accuracy. Among them, Mixed1 and Mixed2, each layer consists of 4 parallel convolutional layers, and all neural network layers are optimized(Fig.6).
3. Training And Testing
I have introduced the overall organization chart, data set configuration and convolution neural network model design and convolution neural network model layer structure and parameter information. Next, I will introduce the training process and testing process of convolution neural networks.
3.1. Training
For the training process, the paper published by Krizevsky et al. [23] at NIPS 2012 details the training of the model in ILSVRC 2012. That is, training is performed by using small-batch gradient descent (based on backpropagation [29]) and momentum-optimized polynomial logistic regression targets. In this paper, in the training process of the convolution neural network model, the data set size of the input dataset is 64x64 for both data sets, the batch size is set to 20, and the learning rate is initially set to 0.1, but the accuracy of the verification set is not ideal. Finally, the learning rate is set to 0.0001, and the maximum number of steps for training is 10,000. The optimization function uses the Adam function. Of course, you can also use BGD gradient descent, SGD random gradient descent, momentum, AdaGrad, RMSprop and other optimization functions [3, 34, 27, 11, 40, 12, 31]. This article only uses the Adam function to optimize the network model. During the running of the code, the training error and accuracy are output every 10 steps. After the training is completed, the training results are evaluated. If not, the parameters of the neural network model are debugged, optimized, and retrained until the expected Effect. Finally save the results.

3.2. Testing
In this paper, two kinds of data sets are used to train the same designed neural network. The main function is to classify the problem and test it from the collected pictures to see if it can achieve the intended purpose. The experimental results show that the convolution neural network structure model designed in this paper can achieve the intended purpose, and in the process of testing, not only can the purpose of classification and identification be achieved, but also the probability of inputting to the object.

4. Training Results
The following part is the structure of the training and verification obtained by the two data sets in this paper. The result is realized by the programmed code. During the training process, the parameters are saved once every round. Parameters include input batch size, maximum number of steps to train, training loss, training accuracy, and more. The results of the following chart are implemented by the visualization tool TensorBoard.

4.1. Results of Training with Yang data Set

![Figure 9. Loss-1](image1)

![Figure 10. Loss](image2)
The above is the result of training the yang data set through data processing into the convolutional neural network model. Fig.9 is a training 10,000 step, outputting an error every 10 steps, and Fig.9 shows the error stability range during a training. Fig.10 showing the downward trend of the training loss value during the training. As can be seen from Fig.10, the training error is declining. When the training reaches about 1000 steps, the training error tends to be the smallest, which is exactly what we expect. Fig.11 reflects the training of 10,000 steps, outputting accuracy every 10 steps, and Fig.11 shows the range of precision stability during a training. As can be seen from Fig.12, when training to the 800th step, the training accuracy rate is the highest, which proves that the training loss value is getting smaller and smaller, the accuracy rate is getting larger and larger, and the expected effect can be achieved.

4.2. Results of Training with Flower photos Data
The above is the result of using the flower photos data set to be trained by data processing into the convolution neural network model. Fig.13 is a training 10,000 steps, outputting an error every 10 steps, the stability range during the whole training process and Fig.14 is the trend of the training error. As can be seen from Fig.14, the training error is decreasing, when the training is reached. At about 9500, the training error tends to be minimal, which is exactly what we expected. Fig.15 is a training of 10,000 steps, outputting an accuracy rate every 10 steps, indicating the range of stability during the entire training process, and Fig.15 is the trend of training accuracy. As can be seen from Fig.16, the training accuracy is constantly improving. When the training reaches about 9,500 steps, the training accuracy tends to be the largest, which is also the result we want.

5. Verification Results
The yang data data set was applied to the convolution neural network model for training after data processing, and the training model was obtained. I collected 50 pictures from the network, named yang-val-data verification set, and verified the network model with 10 pictures of each type. The verification results are shown in Table 3.

| Yang-Val-data | Backpack (accuracy (%)) | Computer-monitor (accuracy (%)) | Computer-mouse (accuracy (%)) | Segway (accuracy (%)) | Self-propelled-lawn-mower (accuracy (%)) |
|---------------|-------------------------|---------------------------------|-------------------------------|----------------------|------------------------------------------|
| 1             | 94.0148                 | 66.9741                         | 99.9733                       | 100.0000             | 99.9999                                   |
| 2             | 100.0000                | 99.997                         | 88.389                        | 100.0000             | 99.9999                                   |
| 3             | 99.9789                 | 99.9992                        | 99.9861                       | 99.5974              | 100.0000                                 |
| 4             | 100.0000                | 99.9992                        | 99.9321                       | 59.1519              | 92.1306                                   |
| 5             | 100.0000                | 99.9996                        | 99.9979                       | 98.7866              | 98.2689                                   |
| 6             | 97.0852                 | 99.991                         | 99.995                        | 75.5642              | 100.0000                                 |
| 7             | 99.9912                 | 99.9954                        | 99.9951                       | 99.9996              | 99.9766                                   |
| 8             | 99.9604                 | 99.7762                        | 88.9416                       | 99.7657              | 100.0000                                 |
| 9             | 97.4838                 | 43.9528                        | 100.0000                      | 99.9999              | 100.0000                                 |
| 10            | 100.0000                | 100.0000                       | 99.9999                       | 99.9999              | 99.9947                                   |
| Average       | 98.85143                | 91.06422                       | 97.721                        | 93.28625             | 99.03706                                  |
| Overall accuracy (%) | 95.991992            |                                 |                               |                     |                                          |

Table 3. The yang-val-data verification set is divided into five types of objects, and there are 10 pictures for each type of objects. The names of these 10 pictures are named with Numbers, and the Numbers of each type of objects are numbered from 1 to 10. Verify the accuracy of all pictures, calculate the average accuracy of each category, and finally calculate the accuracy of all verified pictures as the verification result of this experiment.

The flower photos data set was processed into the convolution neural network model for training, and the training model was obtained. I collected 50 pictures from the network and named them as flower-photos-val-data verification set. 10 pictures of each type were used to verify the network model. The verification results are shown in Table 4.
Table 4. Verification Results

| Flower-photos-Val-data | Roses (accuracy (%)) | Tulips (accuracy (%)) | Dandelion (accuracy (%)) | Sunflowers (accuracy (%)) | Daisy (accuracy (%)) |
|------------------------|----------------------|-----------------------|--------------------------|---------------------------|---------------------|
| 1                      | 99.5858              | 67.7543               | 89.3952                  | 99.2597                   | 95.5429             |
| 2                      | 99.8014              | 99.875                | 99.4264                  | 99.8941                   | 99.3655             |
| 3                      | 98.75                | 99.6056               | 92.1319                  | 99.4332                   | 99.8825             |
| 4                      | 66.7943              | 85.9854               | 97.0716                  | 99.9296                   | 93.9432             |
| 5                      | 99.9524              | 99.3161               | 86.6512                  | 93.6375                   | 88.8055             |
| 6                      | 96.2451              | 97.9132               | 99.2857                  | 99.8844                   | 99.9206             |
| 7                      | 95.5656              | 97.9422               | 84.1936                  | 98.7341                   | 98.3271             |
| 8                      | 99.0375              | 99.3779               | 92.2862                  | 90.4665                   | 99.3953             |
| 9                      | 97.9202              | 92.6182               | 99.2182                  | 98.5249                   | 97.9999             |
| 10                     | 99.1148              | 99.3124               | 99.8441                  | 96.6088                   | 44.1493             |
| Average accuracy (%)   | 95.27671             | 93.97003              | 93.95041                 | 97.63728                  | 91.73318            |
| Overall accuracy (%)   | 96.313522            |                       |                          |                           |                     |

Table 4. Flower-photos-val-data verification set is divided into five types of objects. Each type of object has 10 pictures. Verify the accuracy of all pictures, calculate the average accuracy of each category, and finally calculate the accuracy of all verified pictures as the verification result of this experiment.

6. Comparison of Results

Compare with existing ILSVRC classification methods. My model method is called “YANG”. The results of the existing ILSVRC classification methods in the following list are compared with the results of the model method in this paper, which proves that the method is feasible and achieves a good classification recognition effect. Table 5:

Table 5. The Training Accuracy And Testing Accuracy of Different Models In Object Classification And Recognition

| Method                          | Train accuracy (%) | Test accuracy (%) |
|---------------------------------|--------------------|-------------------|
| Zeiler & Feng [44] (6 nets)     | 85.3               | 85.2              |
| Clarifai [35] (multiple nets)   | -                  | 88.3              |
| MSRA [19] (11 nets)             | -                  | 91.9              |
| CoogLeNet [41] (7 net)          |                    | 93.3              |
| VGG (2 nets, multi-crop & dense eval.) [42] | 93.2            | 93.2              |
| YANG                            |                    |                   |
| Yang-data                       | 99.9778            | 95.991992         |
| Flower-photos                   | 99.9987            | 96.313522         |

Table 5. This table is the training accuracy and test accuracy of different models in object classification and recognition tasks in recent years. The YANG model is designed by myself, and theyang data data set and flower photos are adopted for training and testing, which are compared with the results of other models.

7. Conclusion

In this paper, the designed convolution neural network structure performs maximum pooling every two convolution layers. In Mixed1 and Mixed2, we have widened the neural network. Finally, two fully connected layers and one Softmax layer are added, and experiments are carried out through two different
data sets. Experiments show that the convolution neural network structure designed in this paper can achieve the purpose of classifying images well. A good result has been achieved in classification accuracy. Besides, this article only uses an activation function, a loss function, a classifier, and other functions that have not been tried in this article. In future work, I think it is also a problem worth studying.

Acknowledgments
This article was not funded or received any grants, except the academic support of Qingdao University of Science and Technology.

Conflict of interest: The authors declare that they have no conflict of interest against any company or institution.

References
[1] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).
[2] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., et al. (2016). Tensorflow: a system for large-scale machine learning. In OSDI (Vol. 16, pp. 265-283).
[3] Aylien Ltd., Dublin. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.
[4] Bengio, Yoshua. Estimating or propagating gradients through stochastic neurons. Technical Report arXiv:1305.2982, Universite de Montreal, 2013.
[5] Barrie Gilbert. The multi-tanh principle: a tutorial overview. IEEE Journal of Solid-State Circuits, 1998.
[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1 – 9, 2015.
[7] Cramer, JS (2002). “The origins of logistic regression” (PDF). Tinbergen Institute Working Paper. 119 (4): 167 – 178. doi:10.2139/ssrn.360300.
[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. arXiv: 1409.4842, 2014.
[9] Djork-Errs Clevert, Thomas Unterthiner and Sepp Hochreiter (2016). FAST AND ACCURATE DEEP NETWORK LEARNING BY EXPONENTIAL LINEAR UNITS (ELUS).
[10] De Boer, P-T., Kroese, D.P, Mannor, S. and Rubinstein, R.Y. (2005). A Tutorial on the Cross-Entropy Method. Annals of Operations Research, 134 (1), 19 – 67 [1].
[11] Dozat, T. (2016). Incorporating Nesterov Momentum into Adam. ICLR Workshop, (1), 2013 – 2016.
[12] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research, 12, 2121 - 2159. Retrieved from http://jmlr.org/papers/v12/duchi11a.html.
[13] Goodfellow, I., Bengio, Y.,Courville, A.Deep learning (Vol. 1). Cambridge:MIT press,2016:326-366
[14] Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, L., Wang, G. and Cai, J., 2015. Recent advances in convolutional neural networks. arXiv preprint arXiv:1512.07108.
[15] Goodfellow, Ian J, Warde-Farley, David, Mirza, Mehdi, Courville, Aaron, and Bengio, Yoshua. Maxout networks. arXiv preprint arXiv:1302.4389, 2013.
[16] Glorot, Xavier, Bordes, Antoine, and Bengio, Yoshua. Deep sparse rectifier neural networks. In International Conference on Artificial Intelligence and Statistics, pp. 315 – 323, 2011.
[17] Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A.C. and Bengio, Y. (2013) Maxout Networks. Proceedings of the 30th International Conference on Machine Learning (ICML), 1319 - 1327.
[18] Gelman, Andrew; Hill, Jennifer (2007). Data Analysis Using Regression and Multi-
[19] He, K., Zhang, X., Ren, S., and Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. CoRR, abs/1406.4729v2, 2014.

[20] Hannun, Awni, Case, Carl, Casper, Jared, Catanzaro, Bryan, Diamos, Greg, Elsen, Erich, Prenger, Ryan, Satheesh, Sanjeev, Sengupta, Shubho, Coates, Adam, et al. Deep speech: Scaling up end-to-end speech recognition. arXiv preprint arXiv:1412.5567, 2014.

[21] Han, Jun; Morag, Claudio. The influence of the sigmoid function parameters on the speed of backpropagation learning: From Natural to Artificial Neural Computation., 1995: 195 – 201.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Su, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, LSVRC-2014 contest.

[23] Krizhevsky, I., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In NIPS, pp. 1106 – 1114, 2012.

[24] Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in NIPS, 2012.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Su, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, LSVRC-2014 contest.

[26] LeCun, Y. and Bengio, Y., 1995. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361 (10), 1995.

[27] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1 (4): 541 – 551, 1989.

[28] LeCun, Y., and Bengio, Y., 1995. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, 3361 (10), 1995.

[29] LeCun, Y., Bottou, L., Orr, G. B., and Muller, K. R. (1998). Efficient BackProp. Neural Networks: Tricks of the Trade, 1524, 9–50. http://doi.org/10.1007/3-540-49430-8 - 2.

[30] Murphy, Kevin (2012). Machine Learning: A Probabilistic Perspective. MIT. ISBN 978-0262018029.

[31] Nair, Vinod and Hinton, Geoffrey E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), pp. 807 – 814, 2010.

[32] Qian, N. (1999). On the momentum term in gradient descent learning algorithms. Neural Networks: The Official Journal of the International Neural Network Society, 12 (1), 145 – 151. http://doi.org/10.1016/S0893-6080 (98) 00116 - 6.

[33] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei Fei, L. ImageNet large scale visual recognition challenge. CoRR, abs/1409.0575, 2014.

[34] R Hahnloser, R. Sarpeshkar, M A Mahowald, R. J. Douglas, H. S. Seung (2000). Digital selection and analogue amplification coexist in a cortexinspired silicon circuit. Nature 405: 947 - 951.

[35] R Hahnloser, H. S. Seung (2001). Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks. NIPS 2001.

[36] [Rubinstein, R.Y. and Kroese, D.P. (2004), the Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning, Springer-Verlag, New York ISBN 978-0-387-21240-1.

[37] Rubinstein, R.Y. (1997). Optimization of Computer simulation Models with Rare Events, European Journal of Operational Research, 99, 89 – 112.

[38] Reddi, Sashank J., Kale, Satyen, and Kumar, Sanjiv. On the Convergence of Adam and Beyond. Proceedings of ICLR 2018.
[41] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.

[42] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition [J]. Computer Science, 2014.

[43] Yann LeCun, etGradient-Based Learning Applied to Document Recognition. 1998.

[44] Zeiler, Matthew D. and Rob Fergus Visualizing and Understanding Convolutional Networks. CoRR, abs/1311.2901, 2013. Published in Proc. ECCV, 2014.

[45] Zhang, T., 2004. Statistical behavior and consistency of classification methods based on convex risk minimization. Annals of Statistics, pp.56 - 85.

[46] A. Waibel, T. Hanazawa, G. Hinton, and K. Shikano, “Phoneme recognition using time-delay neural networks,” IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. 37, No. 3, pp. 328 – 339, 1989.