PAPER • OPEN ACCESS

Understanding the Convolutional Neural Networks with Gradient Descent and Backpropagation

To cite this article: XueFei Zhou 2018 *J. Phys.: Conf. Ser.* **1004** 012028

View the article online for updates and enhancements.

You may also like

- Applying Gradient Descent in Convolutional Neural Networks
  Nan Cui
- Convolutional Neural Networks Combined with Machine Vision for Mechanical Compressor Defect Detection
  Kunshan Li
- Attention-based CNNs for Image Classification: A Survey
  Menghua Zheng, Jiayu Xu, Yinjie Shen et al.

ECS The Electrochemical Society
Advancing solid state & electrochemical science & technology

Learn More & Register

243rd Meeting with SOFC-XVIII

Boston, MA • May 28 – June 2, 2023

Accelerate scientific discovery!

This content was downloaded from IP address 213.177.204.11 on 23/05/2023 at 15:41

Understanding the Convolutional Neural Networks with Gradient Descent and Backpropagation

XueFei Zhou[1, *]

[1]School of Computer Science and Technology, Soochow University, Suzhou 215006, China

*821573861@qq.com

Abstract. With the development of computer technology, the applications of machine learning are more and more extensive. And machine learning is providing endless opportunities to develop new applications. One of those applications is image recognition by using Convolutional Neural Networks (CNNs). CNN is one of the most common algorithms in image recognition. It is significant to understand its theory and structure for every scholar who is interested in this field. CNN is mainly used in computer identification, especially in voice, text recognition and other aspects of the application. It utilizes hierarchical structure with different layers to accelerate computing speed. In addition, the greatest features of CNNs are the weight sharing and dimension reduction. And all of these consolidate the high effectiveness and efficiency of CNNs with idea computing speed and error rate. With the help of other learning altruisms, CNNs could be used in several scenarios for machine learning, especially for deep learning. Based on the general introduction to the background and the core solution CNN, this paper is going to focus on summarizing how Gradient Descent and Backpropagation work, and how they contribute to the high performances of CNNs. Also, some practical applications will be discussed in the following parts. The last section exhibits the conclusion and some perspectives of future work.

1. Introduction

The human has a complex visual recognition system. We distinguish and classify objects independently. The structure of the neural network is imitated from the biological neural network. Each neuron in the neural network is connected with other neurons. When it is "excited", it is necessary to send the next level of neurons to send the chemical substance to change the potential of these neurons. If the potential of a neuron exceeds a threshold, it is activated; otherwise it will not be activated.

One of the most common research fields in computer science is image recognition. And Convolutional Neural Networks (CNNs) are hierarchical neural networks for recognitions, by using significant process, such as Gradient Descent [1] and Backpropagation [2]. The idea of the earliest neural network originated in the 1943. McCulloch and Pitts (MCP) [3] raised an artificial neural model, which was intended to use a computer to simulate the process of human neuron response. It simplifies neurons into two processes: linearization of the input signal, and non-linear activation (threshold method). The first use of MCP for machine learning was the perceptron algorithm invented by Rosenblatt in 1958 [4]. The algorithm uses the MCP model to classify the input multidimensional data and uses the gradient descent method to automatically learn the update weights from the training

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

samples. In 1962, the method proved to be able to converge; the theoretical and practical effects caused the wave of the first neural network [5]. However, the history of the development of disciplines is not always plain sailing. In 1969, the American mathematician and artificial intelligence pioneer Minsky [6] proved that the sensor is essentially a linear model, which can only deal with linear classification problems; even the simplest XOR (or) problems cannot be correctly classified. This is equivalent to a direct sentence of the death of sensor, neural network research also stuck in nearly 20 years of stagnation. The first time to break the nonlinear curse was undoubtedly the research conducted by Rumelhart, Hinton, et al. [7], who invented a multi-layer sensor (MLP) BP algorithm in 1986, and using Sigmoid for non-linear mapping, an effective solution to the nonlinear classification and Learning problems. This method causes the second wave of neural networks. In 1989, Robert Hecht-Nielsen [8] proved MLP's universal approximation theorem, that is, for a continuous function f in any closed interval, it can be approximated by a BP network with a hidden layer. This discovery greatly encouraged the researchers. Also in 1989, LeCun [2] invented the convolution of the neural network - LeNet, and its use for digital identification. In 2012, in order to respond to others' query about the deep learning, Hinton and his students applied deep learning to ImageNet (the largest database in image recognition), and ultimately achieved considerably impressive results [9].

This paper is focusing on the theory of CNNs, including its structure and learning process as well as the understanding of Gradient Descent and Backpropagation. Then some practical applications with CNNs will be displayed.

2. Convolutional Neural Networks

2.1. Layers

In a typical CNN, the beginning layer is convolution layer, and the last layer is output layer. The layers between them are called hidden layers. Then main purpose of convolution layer is to extract image features, then drive them into the hidden layers for computing, and output the results via output layer. Layers among hidden layers usually, such as pooling layers (sub-sampling layers) are partially connected, while the output layers are fully connected. Partial connection could reduce the computing load by using shared weights, namely kernels.

2.2. Process

For example, the input layer consists of 32 × 32 sensing nodes, receiving the original image. The calculation then alternates between convolution and sub-sampling, which stated as follows:

The first hidden layer is convoluted, which consists of eight feature maps, each feature map consists of 28 × 28 neurons, each neuron specifies a 5 × 5 acceptance domain;

The second hidden layer implements sub-sampling and local averaging. It is also composed of eight feature maps, but each feature map consists of 14 × 14 neurons. Each neuron has a 2 × 2 acceptance field, a training coefficient, a training bias, and a sigmoid activation function. The training factor and the bias control the operating point of the neuron.

The third hidden layer is the second convolution, which consists of 20 feature maps, each consisting of 10 × 10 neurons. Each neuron in the hidden layer may have a synaptic connection to several feature maps of the next hidden layer, which operates in a similar manner to the first convolution layer.

The fourth hidden layer performs the second sub-sampling and local average calculation. It consists of 20 feature maps, but each feature map consists of 5 x 5 neurons, which operate in a similar manner to the first sample.

The fifth hidden layer implements the final stage of convolution, which consists of 120 neurons, each of which specifies a 5 × 5 accepted domain.

Finally, a full connection layer, get the output vector.

The successive computational layers alternate between convolutional layers and sampling layers, as spatial resolution decreases in each convolutional layer or sampling layer, the number of feature mappings increases compared to the corresponding previous layer.

3. Gradient Descent

In mathematics, gradient is a vector, referring to the direction with largest changing value at one point. The method Gradient Descent used in machine learning is to define the distinction between two values to train the neural networks.

The gradient descent method is based on the following process:

1) First, assign the value of θ, this value can either be random or an all-zero vector.

2) Second, change the value of θ so that J (θ) decreases in the direction in which the gradient decreases.

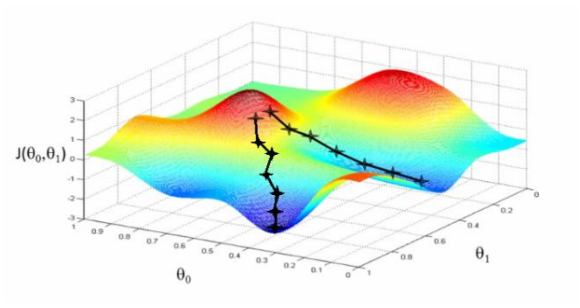

Figure 1. Gradient Descent

Figure 1 is a diagram showing the relationship between the parameter θ and the error function J (θ). The red part is where J (θ) has a relatively high value. We need to be able to make the value of J (θ) as low as possible, where the dark blue part is. Θ0 and θ1 represent two dimensions of the θ vector. The first step of gradient descent is to initiate the value of θ, assuming that the initial value given to the graph is the cross at the graph.

Then we adjust θ in accordance with the direction of the gradient, it will make J (θ) shifts toward a lower direction, as shown in the figure, the algorithm will stop by θ dropping to an extreme point.

Gradient reduction process is stated as follows:

$$\frac{\partial}{\partial\theta}J(\theta)=\frac{\partial}{\partial\theta}\frac{1}{2}\sum_{i=1}^{m}(h_\theta(x)-y)^2=(h_\theta(x)-y)x^{(i)} \tag{1}$$

The following is the process of updating, that is, θi will be reduced according to the direction of the minimum gradient. Θi represents the value before the update, the latter part represents the amount of decrease in the gradient direction, α represents the step size, which is the variation in the direction of the gradient reduction each time.

$$\theta_i=\theta_i-\alpha\frac{\partial}{\partial\theta}J(\theta)=\theta_i-\alpha(h_\theta(x)-y)x^{(i)} \tag{2}$$

A very important thing to be mentioned is that the gradient is directed. For a vector θ, each dimension component θi can find a gradient direction; thus we can find a holistic direction. It is possible to reach a minimum point by following the direction which descends in the quickest way, whether it is local or global.

4. Backpropagation

The Backpropagation error can be seen as the sensitivity of each neuron's base, which defined as follows:

$$\frac{\partial E}{\partial b}=\frac{\partial E}{\partial u}\times\frac{\partial u}{\partial b} \tag{3}$$

$$\text{Since } \frac{\partial u}{\partial b}=1, \frac{\partial E}{\partial b} = \frac{\partial E}{\partial u} = \delta \tag{4}$$

The sensitivity of layer l:

$$\delta^1 = (W^{l+1})^T \times \delta^{l+1} \circ \int{}'(u^l) \tag{5}$$

" ○" means that each element is multiplied. The sensitivity of the neurons in the output layer is different:

$$\delta' = \int{}'(u^l) \circ (y^n - t^n) \tag{6}$$

Finally, the weights are updated for each neuron using the delta rule. Specifically, for a given neuron, CNNs get its input, and then use the delta of the neuron to be scaled. In the form of a vector, for the first layer, the derivative of the error for each weight of the layer (combined into a matrix) is the cross-product of the input of the layer (equal to the output of the previous layer) and the sensitivity of the layer (the delta of the neuron is combined into a vector form). And then get the weight update of this layer of neuron by multiplying partial derivative by a negative learning rate:

$$\frac{\partial E}{\partial W^l} = x^{l-1} \times (\delta^l)^T \tag{7}$$

$$\Delta W^l = -\eta \times \frac{\partial E}{\partial W^l} \tag{8}$$

In fact, there is a specific learning rate η_{l_j} for each weightW_{l_j}.

5. Applications

In 2015, SUN Yan-Feng and her team proposed a deep learning algorithm based on Fisher criterion. In the feed forward spread, the method used convolution neural network to extract automatically image features such as structural information. And it used convolution network of sharing weights and pooling, subsampling methods to reduce the weight number, which considerably reduced the model complexity. When the Back Propagation adjusted the weights, the method adopted the constraints based on Fisher criterion. At the same time, it kept the samples in small distance with-class and large distance between-class; so that the weights could be closer to the optimal value for classification. It improved the recognition rate effectively when the sample size was insufficient or when it had few training iterations.

In 2013, an article entitled "Deep Feature Learning for Knee Cartilage Segmentation Using a Triplanar Convolutional Neural Network" by Adhish Prasoon et al. [10] was published. In the paper, CNN was used to segment the articular cartilage of the knee. The traditional CNN is two-dimensional. The direct expansion to the three-dimensional need more parameters, thus the network is more complex and requires longer training time as well as more training data. And the use of two-dimensional data alone does not utilize the three-dimensional features, may lead to decreased accuracy. To this end, Adhish [10] used a compromise: use xy, yz and xz three 2D CNNs and combine them together.

Google's DeepMind team made a major breakthrough in the computer go by using deep convolution neural network. In Google team's paper, they mentioned "We use 19X19 images to pass the board position" to "train" two different depth neural networks, which is “Policy network" and "value network". Their task is to "pick" out those more promising moves, and to abandon those obviously inferior moves, so that the amount of calculation by computer is confined to a rational range.

6. Conclusion

This paper briefly introduces the process of CNNs. The significance of Gradient Descent and Back Propagation makes CNNs to be trained in a more efficient way.

6.1. Strong non-linear mapping capabilities

BP neural network essentially implements a mapping function from input to output, and mathematical theory proves that the three-layer neural network can approximate any nonlinear continuous function with arbitrary precision. This makes it particularly suitable for solving the complex problem of internal mechanism.

6.2. High degree of self-learning and adaptive ability
When BP neural network is being trained, it can learn to automatically extract "reasonable rules" between input and output data.

6.3. Fault tolerance
BP neural network's overall training results will not be greatly affected even when its local or part of the neurons get damaged, that is to say even if the system is subject to local damage, it can still work properly.

References
[1] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010 (pp. 177-186). Physica-Verlag HD.
[2] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. In Neural Comput., 1(4):541-551.
[3] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The bulletin of mathematical biophysics, 5(4), 115-133.
[4] Rosenblatt, F. (1962). Principles of neurodynamics.
[5] Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of physiology, 160(1), 106-154.
[6] Minsky, M., & Papert, S. (1969). Perceptrons.
[7] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1988). Learning representations by back-propagating errors. Cognitive modeling, 5(3), 1.
[8] Hecht-Nielsen, R. (1988). Theory of the backpropagation neural network. Neural Networks, 1(Supplement-1), 445-448.
[9] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).
[10] Prasoon, A., Petersen, K., Igel, C., Lauze, F., Dam, E., & Nielsen, M. (2013, September). Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network. In International conference on medical image computing and computer-assisted intervention (pp. 246-253). Springer, Berlin, Heidelberg.