Optimizing Convolutional Neural Network Based on Dropout and the Stochastic Gradient Descent Optimizer

Rahil Vora

Research School of Computer Science, ANU
u6415626@anu.edu.au

Abstract. This study proposes a modified convolutional neural network (CNN) algorithm that is based on dropout and the stochastic gradient descent (SGD) optimizer. The SGD adopts the rectified linear unit as the activation function to avoid the gradient problem and accelerate convergence. To address the overfitting problem, the algorithm uses an SGD optimizer, which is implemented by inserting a dropout layer into the all-connected and output layers, to minimize cross entropy. This study used the datasets MNIST a handwritten digit database and use the algorithm for HWR. Handwritten recognition (HWR) is the ability of a computer to receive and interpret intelligible handwritten input from source such as paper documents, photographs, touch- screens and other devices. The algorithm optimizes and decrease complexity by using sensitivity along with dropout layer and SGD on CNN.

Keywords: Handwritten recognition, convolutional neural networks, Deep Learning, sensitivity, MNIST.

1 Introduction

The convolutional neural network (CNN) has attracted considerable attention because of its successful application in target detection, image classification, knowledge acquisition, and image semantic segmentation. Consequently, improving its performance is a research hotspot [1]. When solving the problem of image target detection, the central processing unit (CPU) controls the entire process and the data scheduling of the CNN, and the graphics processing unit (GPU) improves the convolution calculation in the neural network unit and the operation speed of the full-joint layer-merging cell [2]. Although the learning speed of the neural network has been improved, data conversion and scheduling between the CPU and the GPU lead to an increase in time-cost, and a weak GPU platform is prone to process interruption. In [3], an improved CNN based on the immune mechanism was obtained.

Recognition is an area that covers various fields such as, face recognition, image recognition, finger print recognition, character recognition, numerals recognition, etc. [3]. Handwritten Recognition system is an intelligent system able to recognize handwritten digits as human see. Handwritten digit recognition is an important component in many applications; check verification, office automation, business, postal address reading and printed postal codes and data entry applications are few examples [4]. The recognition of handwritten digits is a more difficult task due to the different handwriting styles of the writers.

This study designs an improved activation function to increase the convergence rate by adding a dropout layer between the fully connected and output layers. This convergence rate by adding a dropout layer between the fully connected and output layers. This resolves the overfitting problem caused by the increasing number of iterations and introduces the resolves the overfitting problem caused by the increasing number of iterations and introduces the stochastic gradient descent (SGD) optimizer to the gradient descent strategy to reduce the cumulative stochastic gradient descent (SGD) optimizer to the gradient descent strategy to reduce the complexity and improve the training speed. Thus, a modified CNN algorithm based on dropout and sensitivity and improve the training speed.

1.1 The data:

The MNIST database of handwritten digits, available from this page, has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. It is a good database for people who want to try learning techniques and pattern recognition methods on real-world data while spending minimal efforts on pre-processing and formatting [5].

The original black and white images from NIST were size normalized to fit in a 20x20 pixel box while preserving their aspect ratio. The resulting images contain grey levels as a result of the anti-aliasing technique used by the normalization algorithm. the images were centered in a 28x28 image by computing the center of mass of the pixels, and translating the image so as to position this point at the center of the 28x28 field.

Despite these modern marvels, a pen together with a paper is much more convenient than a keyboard or a mouse. Computers that process handwritings will have to deal with many writing styles and languages, work with arbitrary user-defined alphabets, and understand any handwritten message by any writer. Thus, this research can lead to further handwriting recognition of alphabets, so it can steer the path towards recognising the sentences with little loss and good accuracy.

2 Method

2.1 The Convolution Neural Network

CNN is a multilayer perceptron inspired by the visual neural mechanism. The hidden layer is composed the many neurons and connections between the input and output layers, and usually includes the of the many neurons and connections between the input and output layers, and usually includes convolution, excitation, and pool layers. If the CNN has multiple hidden layers, then multiple the convolution, excitation, and pool layers. If the CNN has multiple hidden layers, then multiple activation functions should be used. The commonly used activation functions include the ReLU and activation functions should be used. The commonly used activation functions include the ReLU and sigmoid functions. The output layer achieves the output results during the transmission of sigmoid functions. The output layer achieves the output results during the transmission of information information after neuron connection, analysis, and balance.

2.2 Stochastic Gradient Descent (SGD) optimizer

Stochastic Gradient Descent (SGD) [7] is a stochastic approximation of the gradient descent optimization and an iterative method for minimizing/maximizing an objective function, which struggles to discover the minima or maxima by iteration. SGD outperforms those bath methods by following the negative gradient of the objective after checking only a single or some training examples. In other words, when the number of training datasets is high, SGD works quickly because it does not use the whole training instances in each integration, which reduces the amount of calculation and improves the computing speed. Furthermore, SGD can dynamically adjust the estimation of the first- and second-order matrices of the gradient of each parameter according to the loss function. Thus, the risk of the model converging to the local optimum can be reduced.

2.3 Dropout layer

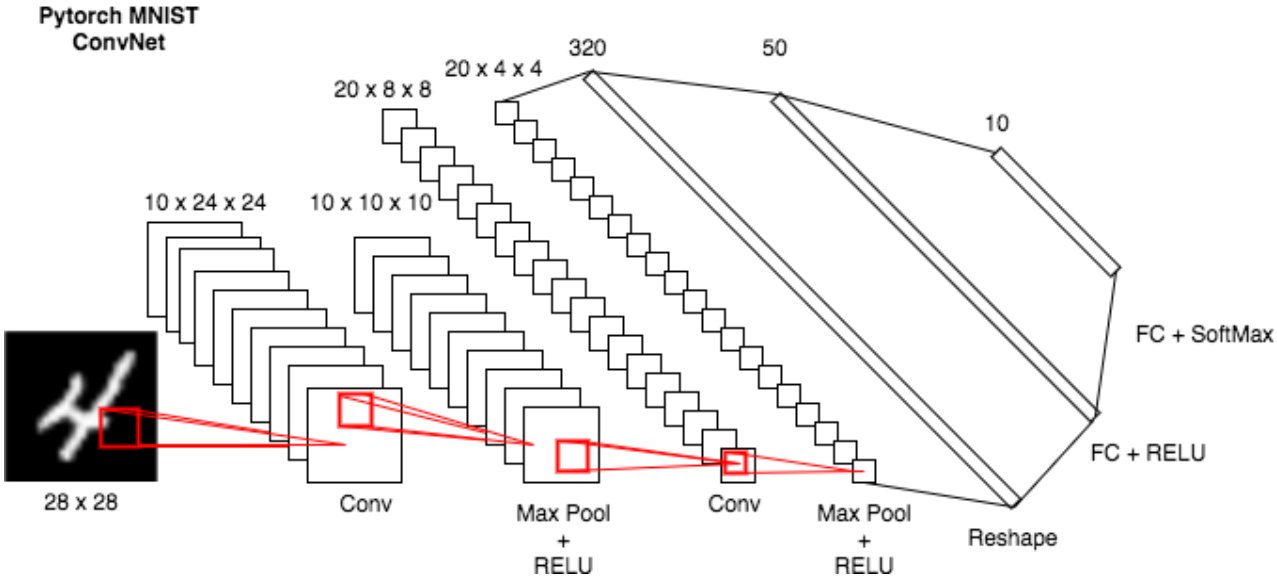
Dropout is a model average. It is the weighted average of the estimated or predicted results from different models. The combinatorial estimate and the combinatorial prediction are the most commonly used methods. The random selection in the dropout may ignore the hidden layer nodes in the training process, consequently, each training network is different, and each training can be regarded as a new model. In particular, the hidden nodes randomly appear according to certain probabilities. Any two hidden nodes are not guaranteed to appear in the model multiple times simultaneously; thus, the updating of weights does not depend on the interaction

of the implicit nodes of the fixed relation and can avoid the possible dependence of several features on another particular feature.

2.4 The model

This study constructs eight-layer neural network. It contains an input layer, four hidden layers consisting of convolution and pooling layers, a connected network layer and one output layer. The activation functions of layers except the output are ReLU (Rectified Linear Unit) functions. The output layer has sigmoid as activation function. Figure 1 shows the visual representation of the model.

Figure1:



2.5 Network Reduction

Sensitivity of the global error function to the removal of a unit can be done by recording the incremental changes to synaptic weights during an epoch of back-propagation.

The information used in approximating the sensitivity uses terms which are often available during training with back-propagation, such as the weight increments, and the partial derivatives of the error surface. The unit with the lowest sensitivity can be removed [8].

There are different ways to implement sensitivity. The Network Reduction technique implemented is sensitivity. It is accomplished by evaluating the output of each hidden neurons to the output layer, and if weight is less than 0.01 for any given hidden neuron in more than one epochs then the weights are assigned to zero. That particular neuron would not contribute to the next layer. Sensitivity helps in reducing the network complexity.

3 Results and Discussion

In this section, we present the experimental results. First, we evaluated the performance of the Convolutional neural network without implementing sensitivity. We trained neural network with 60,000 images. Using the 10,000 images, we tested the accuracy of the model. Our accuracy of model without the network reduction technique is 98.40% but with sensitivity is 98.49%. It seems not much increase but the difference was in CPU time.

3.1 Accuracy

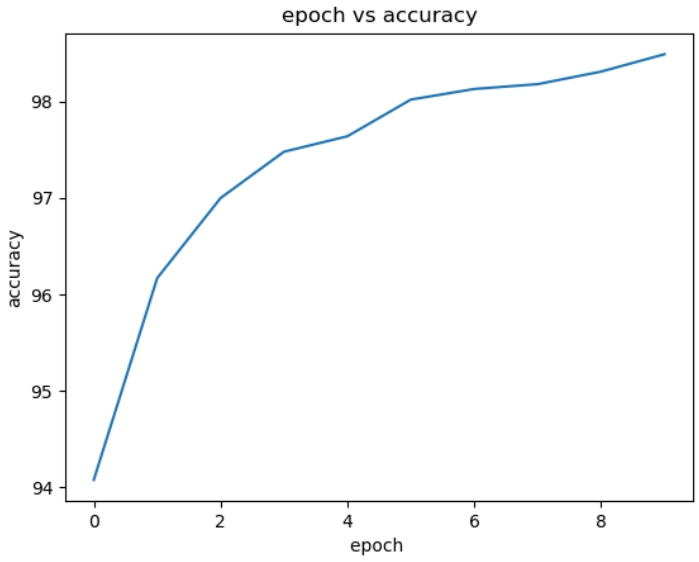
Testing accuracy for epochs are in Table I. The test accuracy of the model stands at 98.49% of the new model compared to model without sensitivity was 98.40%.

Table I. Epoch accuracy

	Test Set Accuracy	
Epochs	**Before Network Reduction**	**After Network Reduction**
1	94.08%	94.08%
2	96.20%	96.17%
3	97.07%	97.00%
4	97.49%	97.48%
5	97.67%)	97.64%
6	98.00%	98.02%
7	98.09%	98.13%
8	98.21%	98.18%
9	98.29%	98.31%
10	98.40%	98.49%

The accuracy graph shows the increase in accuracy on the new proposed mode with sensitivity as network reduction technique. The graph explains the increase in accuracy with increase in number of epoch as the model is improving and it increases finally to 98.49%.

Graph I. Testing Accuracy Graph



3.2 CPU time

Two types of CPU times:

- **User** is the amount of CPU time spent in user-mode code (outside the kernel) *within* the process. This is only actual CPU time used in executing the process. Other processes and time the process spends blocked do not count towards this figure.
- **Sys** is the amount of CPU time spent in the kernel within the process. This means executing CPU time spent in system calls *within the kernel*, as opposed to library code, which is still running in user-space. Like 'user', this is only CPU time used by the process. See below for a brief description of kernel mode (also known as 'supervisor' mode) and the system call mechanism.

With applying the network reduction technique, we can decrease the amount of time consumed by the network on the CPU. The table II shows that the new algorithm decrease the complexity.

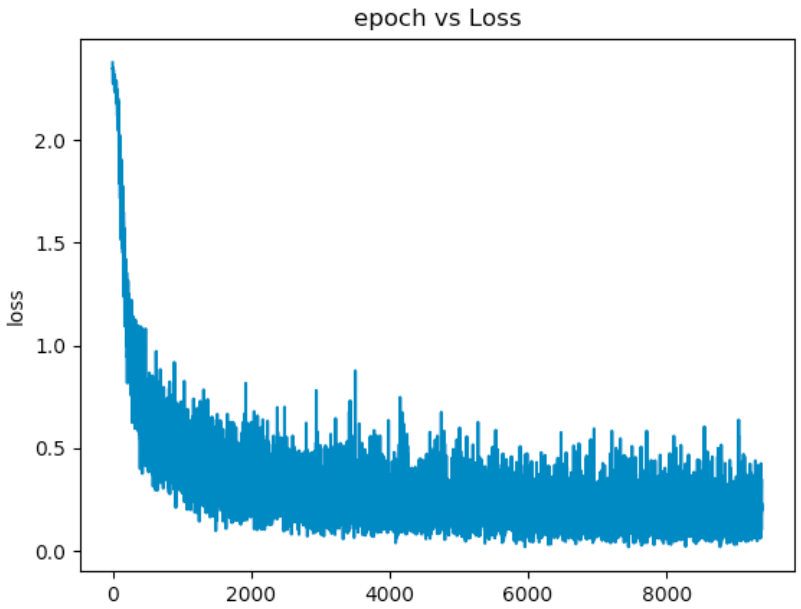
Table II. Time complexity

	Time	
CPU	**Before Network Reduction**	**After Network Reduction**
user	13m13.205s	22m52.385s
sys	0m40.524s	0m53.458s

3.3 Loss

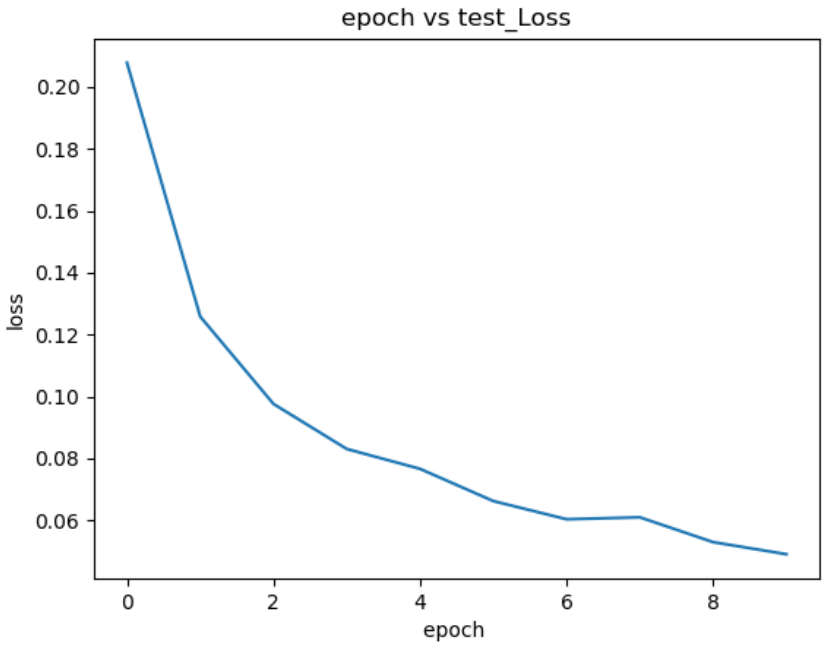
The graph of train loss of new model with network reduction is in Graph II. First the loss starts at 2.37 at first iteration which goes on decreasing, its lowest is 0.9 but keeps on fluctuating from 1.0 to 1.5.

Graph II. Train Loss



The loss at testing of improved algorithm is explained by the Graph III. It displays that the loss is decreasing at each epoch. The loss at first epoch is 0.2 which keeps on decreasing to 0.04 at last epoch.

Graph III. Test Loss Graph



3.1 Comparison:

Ayumi et al [9] have the following results (with same data set) which is better for when it epochs are extended till 100 but sensitivity with Dropout proves a little better that Microcanonical Annealing Algorithm in testing accuracy till 10 epochs:
The result is 96.89% with Ayumi et al's Algorithm at 10 epochs with goes up to 98.75% at 100 epochs. Compared to this algorithm with is 98.49% at 10 epochs.
The time reported by Ayumi et could not be compared with time as it on computer with different configurations.

Epoch	CNN		CNNMA	
	A1(%)	T1(seconds)	A2(%)	T2(seconds)
1	82.39	91.75	86.99 ± 0.53	109.99 ± 10.47
2	89.06	193.39	91.33 ± 0.43	203.04 ± 0.83
3	91.13	297.31	93.14 ± 0.31	302.84 ± 1.74
4	92.33	379.44	94.48 ± 0.15	402.42 ± 0.57
5	93.11	479.04	95.11 ± 0.28	514.52 ± 7.54
6	93.67	576.38	95.72 ± 0.12	612.07 ± 2.33
7	94.25	676.57	95.99 ± 0.22	781.32 ± 23.57
8	94.77	768.24	96.26 ± 0.26	1062.11 ± 6.79
9	95.37	855.85	96.49 ± 0.27	1144.01 ± 79.84
10	95.45	954.54	96.89 ± 0.15	1274.00 ± 47.69
100	98.65	10731	98.75	17090

4 Conclusion and Future Work

To improve the accuracy of handwriting recognition while reducing time-cost, this study investigates the structure of the CNN, the overfitting problem, and combines dropout and network reduction technique and the SGD optimizer with the CNN. The proposed algorithm decrease complexity and improves accuracy. But the Microcanonical Annealing Algorithm on CNN performed better in terms of accuracy when reaching over 100 epochs [9] for handwriting recognition on MNIST dataset.

As the present test dataset is limited, future studies need to use additional test sets to improve the performance of the algorithm. Moreover, the algorithm should be combined with the service robot system [10] established by the research group to conduct application research and to develop and improve model robustness. In addition, reducing the number of neurons in the CNN can lead to the reduction of computations at the working and testing stages; however, the dropout algorithm used at the learning stage increases the time of processing. We focused on improving the recognition accuracy at the beginning of the research.

References:

1. Vieira, S.; Pinaya, W.H.L.; Mechelli, A. Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders: Methods and applications. Neurosci. Biobehav. Rev. 2017, 74, 58–75.

2. Li, S.; Dou, Y.; Niu, X.; Lv, Q.; Wang, Q. A Fast and Memory Saved GPU Acceleration Algorithm of Convolutional Neural Networks for Target Detection. Neurocomputing 2017, 230, 48–59.

3. Gong, T.; Fan, T.; Guo, J.; Cai, Z. GPU-based parallel optimization of immune convolutional neural network and embedded system. Eng. Appl. Artif. Intell. 2016, 36, 226–238.

4. Selvi, P.P., Meyyappan, T.: Recognition of Arabic numerals with grouping and ungrouping using back propagation neural network. In: International Confer- ence on Proceedings of Pattern Recognition, Informatics and Mobile Engineering (PRIME), pp. 322–327 (2013)

5. Mahmoud, S.: Recognition of writer-independent off-line handwritten Arabic (Indian) numerals using hidden Markov models. Sig. Process. 88(4), 844–857 (2008)

6. "MNIST handwritten digit database, Yann LeCun, Corinna Cortes and Chris Burges", *Yann.lecun.com*. [Online]. Available: http://yann.lecun.com/exdb/mnist/. [Accessed: 27- May- 2018].

7. An overview of gradient descent optimization algorithms ». [Online]. Available: http://ruder.io/optimizing-gradient-descent/ [Accessed 16 April 2018].
8. Gedeon, T. D., & Harris, D. (1991). Network reduction techniques. In Proceedings International Conference on Neural Networks Methodologies and Applications (Vol. 1, pp. 119-126).

9. V. Ayumi, L. Rere, M. Fanany and A. Arymurthy, "Optimization of Convolutional Neural Network using Microcanonical Annealing Algorithm", 2018.

10. Yang, G.-C.; Yang, J.; Su, Z.D.; Chen, Z.-J. Improved YOLO feature extraction algorithm and its application to privacy situation detection of social robots. *Acta Autom. Sin.* 2018, 1–12.