21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Cascaded Heterogeneous Convolutional Neural Networks for Handwritten Digit Recognition

Chunpeng Wu, Wei Fan, Yuan He, Jun Sun, Satoshi Naoi
Fujitsu Research & Development Center Co. Ltd., Beijing, 100025, China
{*wuchunpeng, fanwei, heyuan, sunjun, naoi*}@*cn.fujitsu.com*

Abstract

This paper presents a handwritten digit recognition method based on cascaded heterogeneous convolutional neural networks (CNNs). The reliability and complementation of heterogeneous CNNs are investigated in our method. Each CNN recognizes a proportion of input samples with high-confidence, and feeds the rejected samples into the next CNN. The samples rejected by the last CNN are recognized by a voting committee of all CNNs. Experiments on MNIST dataset show that our method achieves an error rate 0.23% using only 5 CNNs, on par with human vision system. Using heterogeneous networks can reduce the number of CNNs needed to reach certain performance compared with networks built from the same type. Further improvements include fine-tuning the rejection threshold of each CNN and adding CNNs of more types.

1. Introduction

Considerable efforts have been devoted to handwritten digit recognition for many years. The proposed methods are based on different classifiers, such as K-Nearest neighbors [1], boosted stumps [2] and neural networks [3].

Recently, CNN-based methods yield state-of-the-art performance [4, 5]. The CNN automatically learns translation-invariant features without using handcrafted feature extractors. The CNN captures topological properties of the input by the operations of convolution and spatial pooling. Spatial pooling is important to obtain translation-invariant features. Two spatial pooling techniques are popularly used: Sub-sampling [6] and bio-inspired max-pooling [7].

Previous research mainly contributes to the improvement of a single CNN [8, 9]. Handwriting recognition based on multiple CNNs of the same architecture is studied in [10]. However, we focus on the reliability and complementation of heterogeneous CNNs. In our method, each CNN in the cascade will adopt a strict rejection threshold. On the other hand, CNNs of different types are supposed to be complementary in our method. Analysis in Subsection 2.1 and experimental results in Subsection 3.2 will show the advantage of using heterogeneous CNNs.

The rest of this paper is organized as follows: The framework is described in Section 2. Then Section 3 presents the experimental results on MNIST, and the further analysis. Section 4 draws the conclusions.

2. Proposed method

The framework of our proposed method is shown in Fig. 1. Our method is composed of S stages. The first $S-1$ stages respectively correspond to a CNN_i, and the last stage is a voting $Committee$ constructed by combining the above all CNN_i ($i = 1, ..., S-1$). All CNN_i are heterogeneous, and each CNN_i is trained separately with randomly distorted training samples in our method. The testing procedure is as follows: The testing samples are fed into CNN_1. Then each CNN_i recognizes a proportion of input samples and feeds the rejected samples into the next stage, i.e., CNN_{i+1} or the $Committee$. The $Committee$ recognizes all input samples from CNN_{S-1}. The rejection rate of each CNN_i is controlled by a threshold T_i.

2.1. Heterogeneous CNNs

The main difference between various CNNs is the operations of spatial pooling. Empirical results show that max-pooling outperforms sub-sampling, and converge faster [11]. However, recent theoretical analysis indicates that the optimal pooling type for a given classification problem may be neither sub- nor max- pooling, but something in between [12]. Therefore, the CNNs based on the above two pooling types will both be

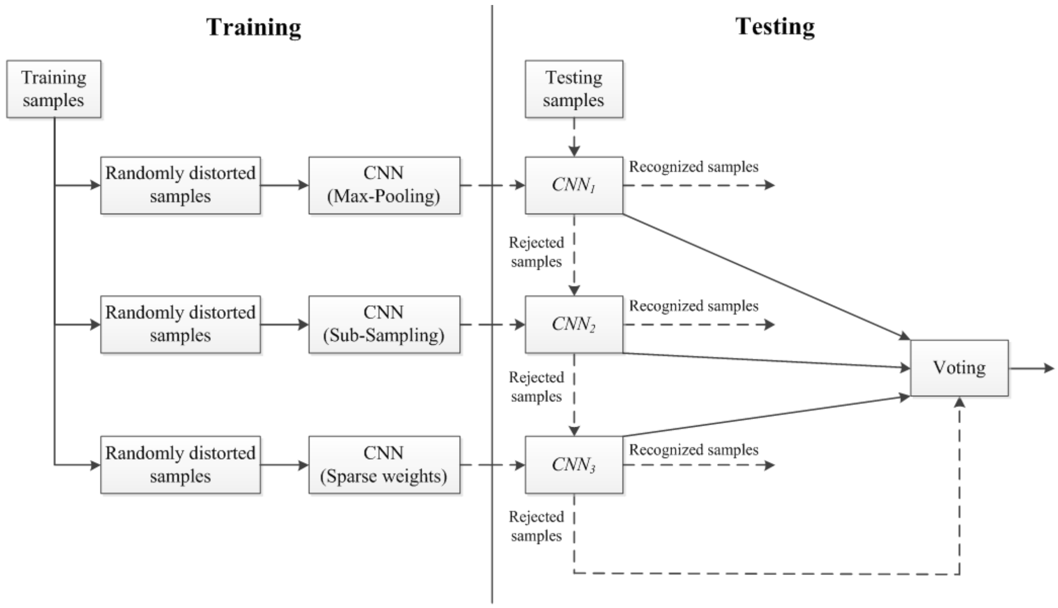

Figure 1. Framework of proposed method.

used in our method. Besides, we also introduce the sparseness measures which are often used for traditional neural networks to reduce the number of free parameters and avoid over-fitting. The CNNs with sparse weights are constructed by randomly reducing some connections between spatial pooling and convolutional layers before training. The above different types of CNNs use the same classical MSE cost function and the same squashing function.

2.2. Rejection rule

The top two most reliable output of CNN_i at output layer are denoted as g_1 and g_2. Given the nth testing sample, the rejection rule is then defined as

$$(g_{1n} - g_{2n}) < T_i. \tag{1}$$

The threshold T_i should be strict, ensuring that the most of suspicious samples will be rejected by a CNN based on a strict rejection rule, and the remaining samples will be recognized with high confidence. The T_i is estimated on the training sets. Given M training samples, the threshold T_i is defined as

$$T_i = \alpha_i \cdot \max_m (g_{1m} - g_{2m}) \tag{2}$$

where $\alpha_i \in [0, 1]$ and $m = 1, ..., M$.

3. Experimental results

Our method is applied to MNIST dataset of handwritten digits to evaluate its effectiveness. The performance at each stage, the misclassified samples and the rejection threshold are further analyzed.

3.1. Settings of parameters and training

The number of stages S is fix to 6. Parameter settings of these six stages are shown in Table 1. "I", "C", "M", "S", "F" and "O" represent input, convolutional, max-pooling, sub-sampling, full-connected and output layer. The number of feature maps and the kernel size of a layer are also specified in Table 1, e.g., "6C5" indicates that this convolutional layer has 6 feature maps and the kernel size is 5. The models of $CNN_1 \sim CNN_4$ are the same except that some neurons in CNN_4 are randomly disconnected before training as stated in Subsection 2.1. The only difference between the first four CNNs and CNN_5 is the type of spatial pooling operators, i.e., max-pooling or sub-sampling. The size of input image is 29x29. The squashing function of a neuron we use is defined as

$$y = 1.1 \cdot tanh(x) \tag{3}$$

where y and x are the output and input of a neuron, and $tanh(\cdot)$ is hyperbolic tangent.

$CNN_1 \sim CNN_5$ are all trained using on-line gradient descent, and the maximum number of training epochs is fix to 800. Actually, if the averaged error rate of the latest five epochs is lower than 0.1%, the training will be stopped. To achieve better generalization, the training set is expanded by random distortion techniques including elastic deformations [4], scaling and rotating transforms.

Table 1. Parameters settings at each stage

Stage	Model
CNN_1	I-6C5-6M3-50C3-50M2-100F5-100F1-O
CNN_2	I-6C5-6M3-50C3-50M2-100F5-100F1-O
CNN_3	I-6C5-6M3-50C3-50M2-100F5-100F1-O
CNN_4	I-6C5-6M3-50C3-50M2-100F5-100F1-O
CNN_5	I-6C5-6S3-50C3-50S2-100F5-100F1-O
$Committee$	

Table 2. Performance on MNIST dataset

Reference	Method	Error Rate
Lecun et al. [6]	Boosted Letnet-4	0.70%
Mizukami et al [13]	KNN	0.57%
Lauer et al. [14]	TFE-SVM	0.54%
Keysers et al. [1]	KNN	0.52%
Simard et al. [4]	CNNs	0.40%
Ranzato et al. [5]	CNNs	0.39%
Ciresan et al. [10]	CNNs	0.27%
Our Method	**CNNs**	**0.23%**

Better performance can be achieved by using different threshold T_i for each CNN, however, the combination of thresholds may be too sensitive to training samples. Therefore, we apply the same threshold $T = \max_i T_i$ to all CNNs, instead of respective T_i. The T is the most strict rejection threshold among T_i ($i = 1, ..., S-1$). Although the CNNs are heterogeneous, they use the same MSE cost function and the same squashing function as stated in Subsection 2.1. Therefore, the output ranges of neurons at output layers across all CNNs are similar. Adopting the same threshold is thus reasonable. According to the squashing function in Eq. 3, the upper bound of $g_{1m} - g_{2m}$ in Eq. 2 is 2.2. Therefore, we fix the threshold T to 2 ensuring a high confidence.

3.2. Performance on MNIST dataset

The MNIST dataset of handwritten digits is composed of 60000 samples for training and 10000 samples for testing [6]. We follow the standard usage of MNIST dataset as [1, 4, 5, 6, 10, 13, 14], and the respective error rates of $CNN_1 \sim CNN_5$ are 0.37%, 0.38%, 0.34%, 0.61% and 0.41%. Our method finally achieves the lowest error rate 0.23% as shown in Table 2. Besides, the recognition error rate of human is estimated as 0.2% [15], our result is thus comparable to human vision.

Table 3. Performance at each stage

No.	Stage	Recognized	Misclassified
1	CNN_1	13.83%	0
2	CNN_2	8.20%	0
3	CNN_3	10.36%	1
4	CNN_4	16.54%	0
5	CNN_5	25.27%	7
6	$Committee$	25.80%	15
		100%	23

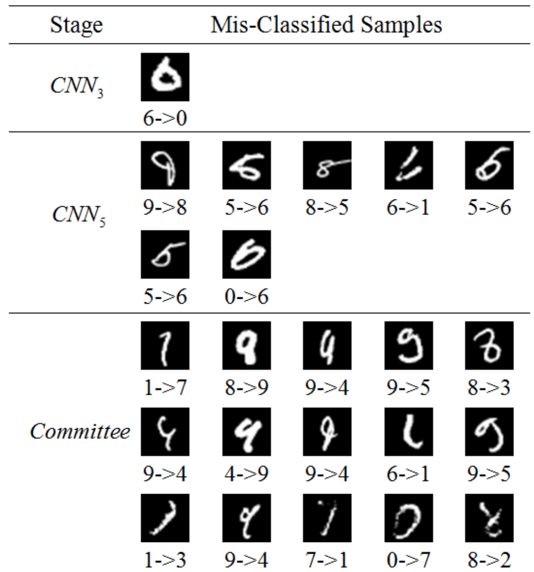

Figure 2. Misclassified samples at each stage. The lower label is "ground truth → prediction".

Each stage in our method contributes to the performance as shown in Table 3. The column "recognized" is the ratio of the number of recognized samples at the stage to the number of all testing samples. Totally only 25.8% samples are rejected by the first five stages ($CNN_1 \sim CNN_5$), and further recognized by stage 6 ($Committee$). The samples misclassified by the CNN_3, CNN_5 and the $Committee$ are shown in Figure. 2. The pairs "4-9" and "5-6" are confused due to cursive writing while other misclassified samples are due to missing strokes, stroke touching and etc. Most of these samples are difficult for a machine to make a correct prediction.

Changing the sequence of cascaded CNNs does not have impacts on the performance. However, the performance is decreased by 13%∼39% without using any

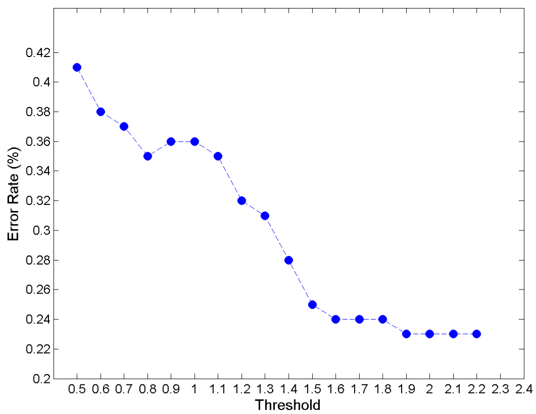

Figure 3. Error rates using different thresholds.

Table 4. Number of needed CNNs

No.	Needed CNNs	Error rate
Max-Pooling	7	0.23%
Sub-Sampling	> 10	0.23%
Sparse Weights	>10	0.23%

one of the five CNNs.

Strict rejection threshold ensures lower error rate as shown in Fig. 3. The error rate rises sharply when the threshold is smaller than 1.5, because the current single CNN is more unreliable for these samples.

Using heterogeneous networks can reduce the number of CNNs needed to reach certain performance compared with networks built from the same type as shown in Table 4. The first column in Table 4 corresponds to the types of CNNs stated in Subsection 2.1. To achieve the error rate 0.23%, at least 7 CNNs of type "Max-Pooling" are needed, while only 5 heterogeneous CNNs are used to achieve the same performance as shown in Table 1.

We use a system with Xeon X5690 (3.47GHz) and 24GB RAM. OpenMP is enabled for parallel computing. The testing speed is about 2.3ms per sample.

4. Conclusions

A handwritten digit recognition method based on cascaded heterogeneous CNNs is presented. Experiments on MNIST dataset show the effectiveness of our method. Some misclassified samples are due to missing strokes, stroke touching which patterns are not contained in the training sets. Therefore, adding training samples corresponding to such patterns can further improve the performance.

References

[1] D. Keysers, T. Deselaers, C. Gollan and H. Ney. Deformation models for image recognition. *PAMI*, 29(8): 1422-1435, 2007.

[2] B. Kegl and R. B. Fekete. Boosting products of base classifiers. *ICML*, 2009.

[3] R. Salakhutdinov and G. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. *Journal of Machine Learning Research*, 2:412-419, 2007.

[4] P. Y. Simard, D. Steinkraus and J. C. Platt. Best practice for convolutional neural networks applied to visual document analysis. *ICDAR*, 2003.

[5] M. Ranzato, C. Poultney, S. Chopra and Y. Lecun. Efficient learning of sparse representations with an energy-based model. *NIPS*, 2006.

[6] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner. Gradient based learning applied to document recognition. *Proc. of the IEEE*, 86(11): 2278-2324, 1998.

[7] M. Ranzato, F. J. Huang, Y. L. Boureau and Y. Lecun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. *CVPR*, 2007.

[8] W. M. Pan, T. D. Bui and C. Y. Suen. Isolated handwritten Farsi numerals recognition using sparse and overcomplete representations. *ICDAR*, 2009.

[9] Q. V. Le, J. Ngiam, Z. Chen, D. Chia, P. W. Koh and A. Y. Ng. Tiled convolutional neural network. *NIPS*, 2010.

[10] D. C. Ciresan, U. Meier, L. M. Gambardella and J. Schmidhuber. Convolutional neural network committees for handwritten character classification. *ICDAR*, 2011.

[11] D. Scherer, A. Muller and S. Behnke. Evaluating of pooling operations in convolutional architectures for object recognition. *ICANN*, 2010.

[12] Y-L. Boureau, J. Ponce and Y. Lecun. A theoretical analysis of feature pooling in visual recognition. *ICML*, 2010.

[13] Y. Mizukami, K. Yamaguchi, J. Warrell, P. Li and S. Prince. CUDA implementation of deformable pattern recognition and its application to MNIST handwritten digit database. *ICPR*, 2010.

[14] F. Lauer, C. Y. Suen, G. Bloch. A trainable feature extractor for handwritten digit recognition. *Pattern Recognition*, 40(6): 1816-1824, 2007.

[15] P. Simard, Y. Lecun and J. Denker. Efficient pattern recognition using a new transformation distance. *NIPS*, 1993.