International Journal of Intelligent Computing and Information Sciences

IMPROVING FEATURE MAPS IN EARLY LAYERS OF CONVOLUTIONAL NEURAL NETWORKS USING OTSU METHOD

A. T. Al-furas M. AL-dosuky T. Hamza

Faculty of Computer and Information,Mansoura University, Egypt.
amroso783@gmail.com *Dr_dos_ok@yahoo.com* *taher_hamza@yahoo.com*

Abstract: *A novel deep architecture Thresholding Convolution Neural Network (ThCNN) progresses in this paper; Which is a simple and effective method to regularizing features map in the early layers of Convolution Neural Network(CNN). One of the issues identified with deep learning is the features in early layers that robustness and discriminativeness. In this paper, we compute the optimal global threshold to determine the features that are passed to the next layers. We then evaluate ThCNN on an MNIST dataset comparing it CNN by applying multiple trained models. It yield decent accuracy compared to traditional CNN. It gives a* ***99.5%*** *accuracy compared to 99.3% for traditional CNN.*

Keywords: *Deep learning, CNN, ThCNN, Otsu Method.*

1. Introduction

Most tasks of the computer vision system are unraveled utilizing machine learning. With machine learning, models are not straightforwardly developed by specialists, they are rather prepared taking into account information. In a previous couple of years, profound learning calculations have altered machine learning. In numerous classification assignments, deep learning has definitely surpassed past best accuracy in classification [1]. Deep convolutional neural networks (CNNs) [2] have as of late been connected to huge picture datasets, for examples MNIST[3] , SVHN[4] , for image classification [5].In spite of the advances that have been made in the improvement of this innovation, numerous issues identified with deep learning remain, including: (1) training data are limited, which may prompt overfitting[6] ;(2) During training gets the erosion of the gradients and unit activation saturated[7]; (3)and in early layers learned features are robustness and distinguished[8].Motivated by that a classifier trained on the high-discriminatory level features will supply better performance than the classifier that are trained on a low-discriminatory level features. This paper has provided an effective way to take advantage of features map in the early layers, by focusing on the most points in the effectiveness of these layers, depending on the features map in these layers is a set of points that responded to the filter in the previous convolution layer. So keep the strongest points in response to the filter, which gives a more separation. The method used often is the method of Otsu [9], which selects the optimal global threshold by maximizing the contrast between the class. In bi-level threshold, the pixel gray level is less than the threshold will be set to the background, and another to the fore.

2 . Background

CNNs [3] comprise of exchanging convolutional layers and pooling layers. Convolution layers take the inward result of the straight channel and the fundamental open field took after by a nonlinear actuation capacity at each neighborhood part of the input. The subsequent yields are called feature maps[3]. The late accomplishment of deep CNN is to a great extent credited to the simultaneous advances of the two specialized streams. The new profound CNN structural planning with components, for example, Dropout[5], DropConnect[10], Redressed Straight Units-ReLU[11], Maxout Network [12] and also new advancement techniques[13] have engaged profound CNN with more prominent learning limit. Typically, data are disseminated on nonlinear manifolds, linear filters can't separable it. For upgrading model distinguishable, Network In Network(NIN)[14] model uses a sliding small-scale neural system, multilayer perception (MLP), to expand the non-linearity of neighborhood patches keeping in mind the end goal to empower the deliberation of more prominent amounts of data inside of the open fields. So also, Deeply Supervised Nets (DSN)[15] gives comrade objective functions to restriction hidden layers, such that strong components can be learned in the early layers of a deep CNN structure. Dropout[5] and Dropconnect [10] methods are broadly used to regularize deep networks so as to avoid overfitting. The thought of the method is to haphazardly drop units or associations with keep units from co-adjusting, which has been appeared to enhance arrangement precision in various studies[12, 15].ReLU, Maxout Network, Batch-normalized studied some of the solutions to the problem of vanishing[16].

2.1 convolutional neural network

Convolution network consists of reciprocally layers of convolution and pooling. The goal of the first convolutional layer is extracted patterns located inside the local areas of the input image and that are common in all elements of the data set. This is done by convolving a particular filter on each Pixels in the input image and pass this filter on all parts of the image to come out of the next layer a feature map h, each of the filters in the layer. The output here is a measure of the extent Filters matches with every part of the image. Thereafter applied a non-linear function g() to each feature map h : a = g(h). Thereafter passed the resulting activations to the pooling layer. These aggregates information within the range of small local areas, R, resulting in a pooled map feature s (smaller) as output. Indicate the aggregation function as pool (),for every feature map h, s_i represented by:

$$s_i = \text{pool}\left(g(h_j)\right) \qquad \forall j \in Ri \tag{1}$$

where Ri is pooling region i in feature map h and j is the index of each element belong to it. The motive of the pooling process is that the activations in the image of the original map feature h more sensitive to specific locations of structures within the pooled map s .The convolutional layers, which it's input is pooled map s, and then cannot extract features increasingly to the local transformations of the input image. This is significant to carry out the functions of classification because these transformations confusion the identity of object[17]. The method proposed in this paper based on the separation of the object from the background in the features map resulting from the previous layer, and thus getting rid of a great deal of confusion. There is a set of functions that can be used for g(), with tanh() and sigmoid functions being common selects. Average and max are two familiar choices for pool():The first takes the arithmetic mean of the elements in each pool area:

$$s_i = \frac{1}{|R_i|}\sum_{j\in R_i} a_j \tag{2}$$

while the largest element selected in max operation:

$$s_i = \max_{j \in R_i} a_j \quad (3)$$

2.2 Otsu Method

Assuming Lv gray levels [0,1,2,……,Lv-1] represent image,ni is the amount of pixels at level j , N= n0+ n1+……+ nLv-1 is a total number of pixels. The probability of gray level j is represented by:

$$p_j = \frac{n_j}{N} \ , p_j \geq 0 \ , \quad \sum_{0}^{Lv-1} p_j = 1 \quad (4)$$

Divided the pixels of the image into two classes C1 with [0,1,…..,t]gray levels and C2 [t,1,…..,Lv-1]with gray levels. For the two classes the probability distributions of gray level are:

$$m_1 = \Pr(C_1) = \sum_{j=0}^{t} p_j \quad (5)$$

$$m_2 = \Pr(C_2) = \sum_{j=t+1}^{Lv-1} p_j \quad (6)$$

The means of tow class's C1, C2 are :

$$me_1 = \sum_{j=0}^{t} \frac{j \times p_j}{m_1} \quad (7)$$

$$me_2 = \sum_{j=t+1}^{Lv-1} \frac{j \times p_j}{m_2} \quad (8)$$

The total mean are :

$$me_t = me_1 \times m_1 + me_2 \times m_2 \quad (9)$$

The variances of tow class's are :

$$\sigma_1^2 = \sum_{j=0}^{t} \frac{(j - me_1)^2 \times p_j}{m_1} \quad (10)$$

$$\sigma_2^2 = \sum_{j=t+1}^{Lv-1} \frac{(j - me_2)^2 \times p_j}{m_2} \quad (11)$$

The within - class variance is :

$$\sigma_W^2 = m_1 \times \sigma_1^2 + m_2 \times \sigma_2^2 \quad (12)$$

The between-class variance is :

$$\sigma_B^2 = m_1 \times (me_t - me_1)^2 \ + m_2 \times (me_t - me_2)^2 \quad (13)$$

The gray levels total variance is :

$$\sigma_T^2 = \sigma_W^2 + \sigma_B^2 \quad (14)$$

Select the optimal threshold t by minimizing variance of the within - class, which is tantamount to maximizing variance of the between - class, where the total variance (the sum of the between - class variance and the within - class variance) is constant for all partitions.

$$t = \arg\left\{\max_{0\le t<Lv-1}\left(\sigma_B^2(t)\right)\right\} = \arg\left\{\min_{0\le t<Lv-1}\left(\sigma_W^2(t)\right)\right\} \quad (15)$$

3 . Proposed Model

We are motivated by this simple observation: In general, a classifier trained on the high-discriminatory level features will provide better performance than the discriminatory classifier that are trained on a low-discriminatory level features. We expect that we will get the features that are more effective and give more differentiation between the deep layers of if we only rely on features resulting directly from layers. ThCNN architectural is similarity traditional CNN architectural ,the difference is adding a new layer, which regularize the features resulting from the pooling layer. In this layer determined global threshold depending on the method of Otsu technique on all features map, To determine feature that represent the objects from those feature that are not significant and affect the classification process negatively. Indicate the regularization function as threshold (), for every feature map s we have:

$$st_i = \text{threshold}(s_i) \qquad \forall R_i \quad (16)$$

ThCNN architectural can be described as involving the following process:

1. Convolve several small filters on the input image.
2. Subsample this space of filter activations.
3. **regularize the features resulting by determining global threshold .**
4. Repeat steps 1,2 and 3 until your left with sufficiently high level features.
5. Use a standard FFNN to solve a particular task, using the results features as input.

Where N represent the number of maps on feature map s , the following algorithm describes the steps to convert the feature map s:-

```
For i=1 to N
  TH←Get  global threshold(s(i))
   For each point in s(i)
     If (point <TH) THEN
point←0
end if
end for
end for
```

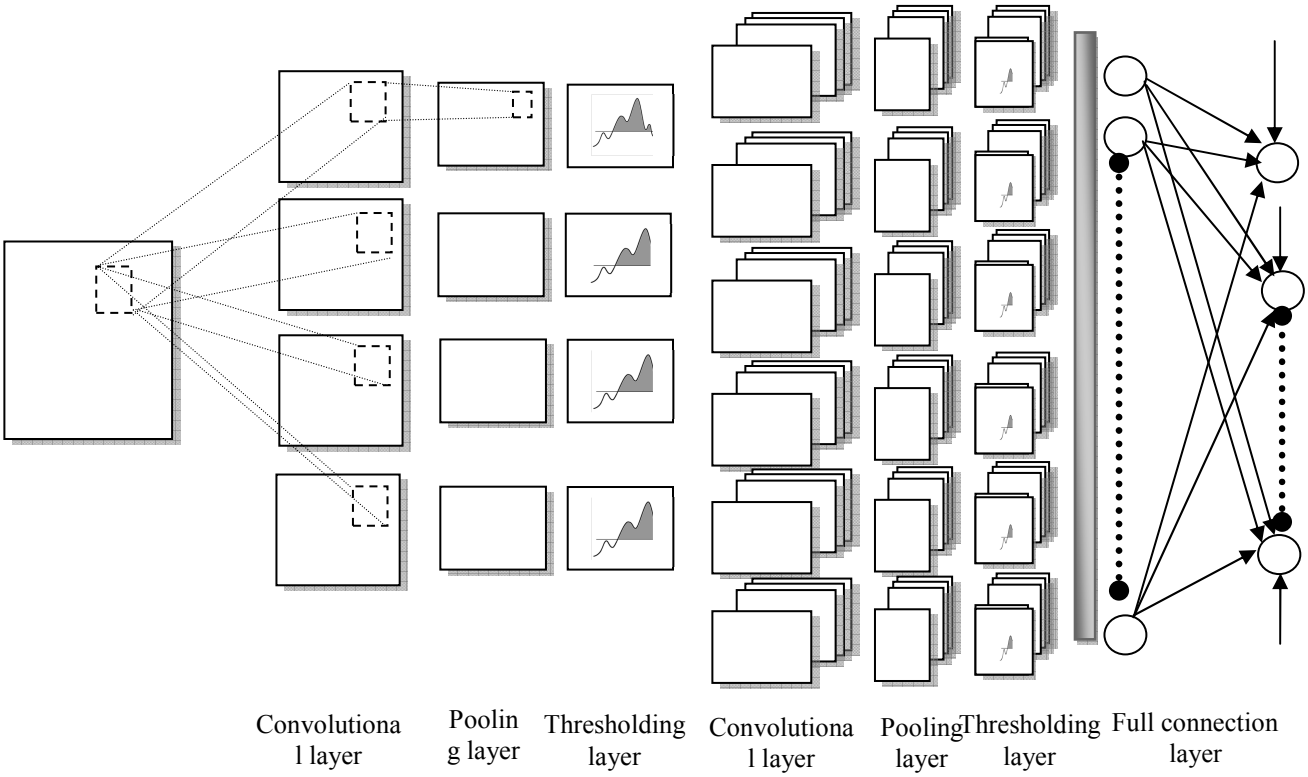

Figure 1: Thresholding Convolutional Neural Network architecture for visual recognition.

4. Experiments

We evaluate our method on MNIST[3]grayscale images datasets, consists of 28× 28 pixels of handwritten digits 0-9, with 60,000 images for training and 10,000 images for testing. We selected 500 samples per class from the training and 100 samples per class from the testing. In CNN architecture, the input layer connected to the first layer , that has 6 feature maps during 6 5x5 kernels. The second layer is a 2x2 pooling layer connected to the third layer which has 12 feature maps during 72 5x5 kernels. The fourth layer is a 2x2 pooling layer, where it's feature maps concatenated into a feature vector feeds into the full connection layer. CNN and ThCNN were trained with SGD, which a batch size of 50 images and learning rate of 1 was used for first 50 epochs and 0.5 for second 50 epochs. The network was evaluated using the DeepLearnToolbox[18].

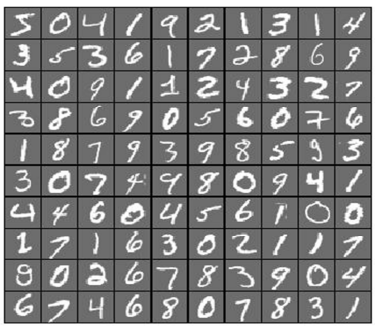

Figure 2:MNIST dataset

First experiment applied a non-linear function sigmoid in all feature map , sigmoid in output layer with the change of the pooling function. Table 1 shows that the ThCNN give the best results, whatever the type of pooling function

.

Table1: classification error for method with various pooling functionand sigmoid in output layer

pooling function	Method	Accuracy(%)
Avg-pooling	CNN	98.2
	ThCNN	**98.3**
Max-pooling	CNN	98.6
	ThCNN	**98.8**

Figure 3 shows that ThCNN began by giving a smaller error at early time better than CNN, figure 4 shows that the mean squared error in ThCNN always lower than in CNN This demonstrates that ThCNN learned better than from CNN.

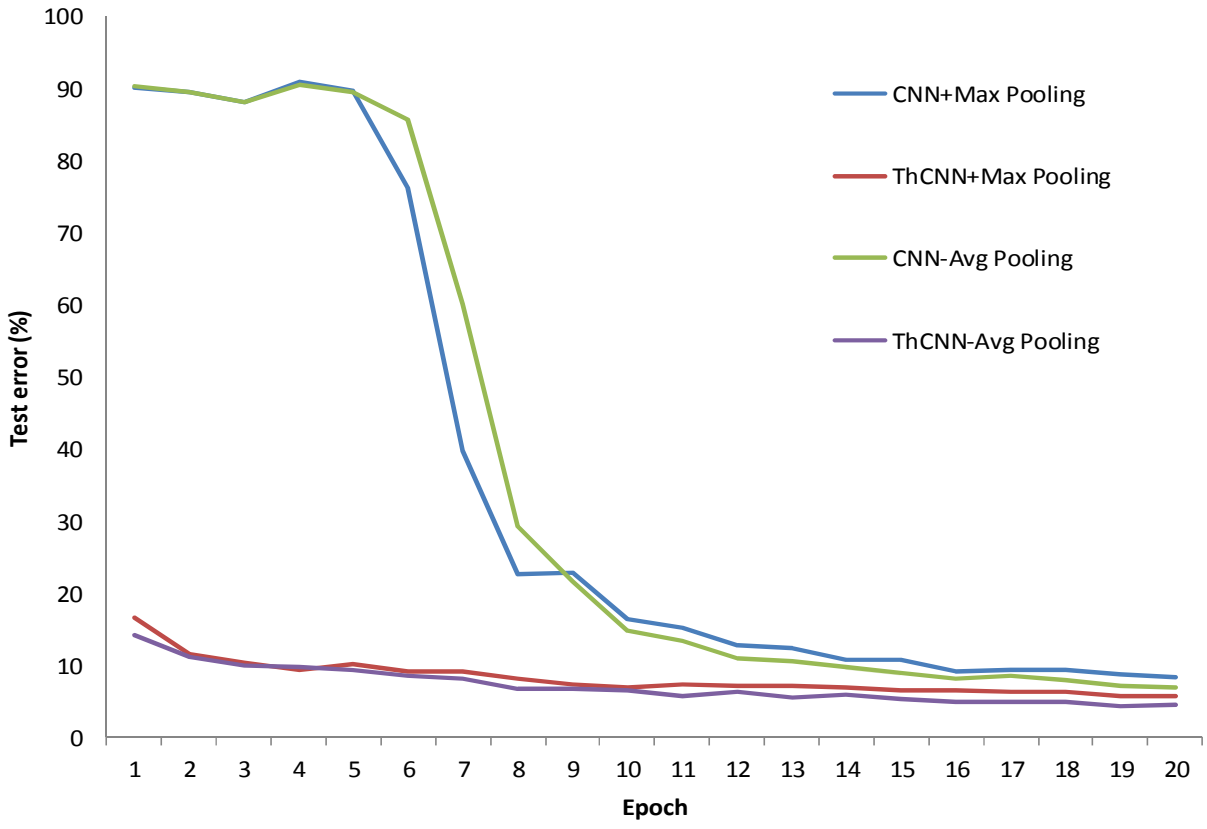

Figure 3 : Test error in first 20 epoch

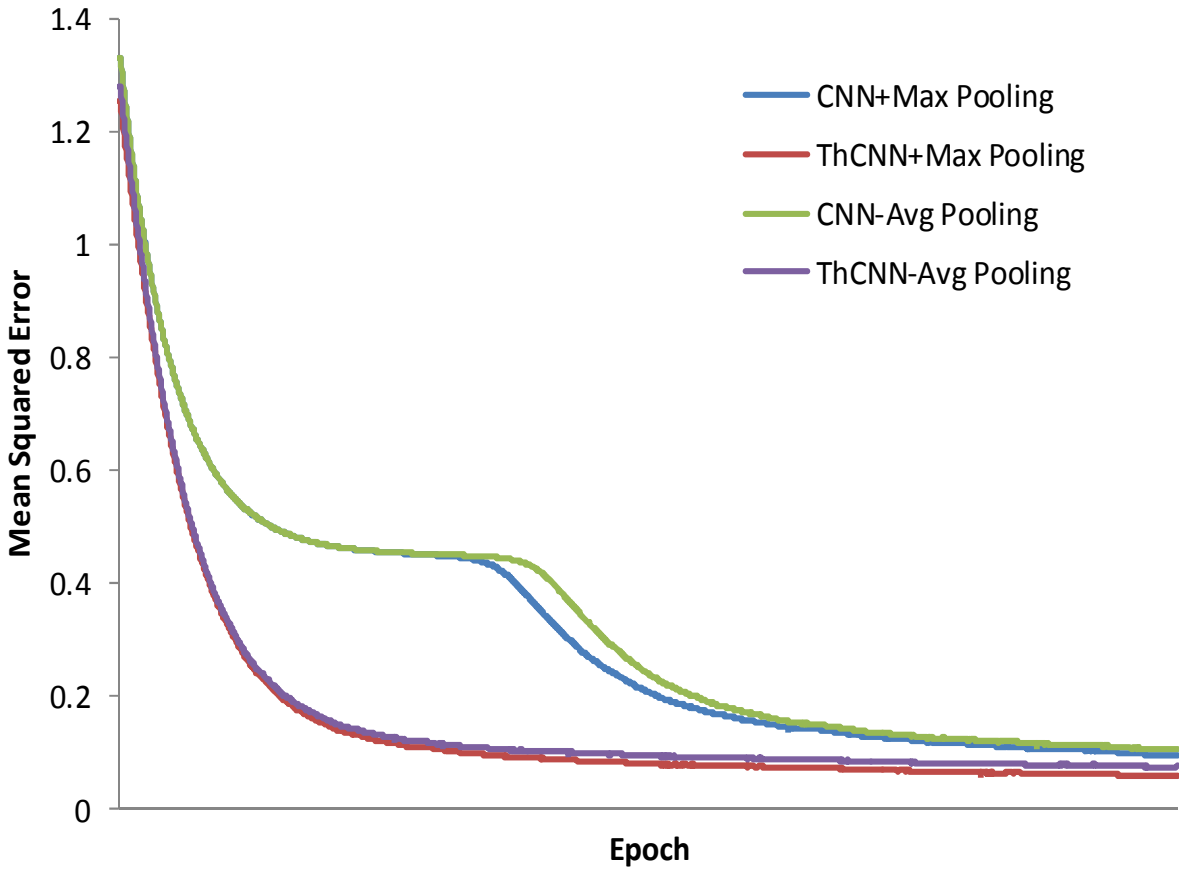

Figure 4 :Mean squared error in first 20 epoch

Second experiment applied a non-linear hyperbolic tangent (tanh) in all feature map , sigmoid in output layer with the max-pooling function.

Table2: classification error a non-linear hyperbolic tangent (tanh) in all feature map

Method	Accuracy(%)
CNN	99.1
ThCNN	**99.2**

Third experiment applied a non-linear hyperbolic tangent (tanh) in all feature map , rectified linear unit (ReLU) in output layer .In this experiment the rate of 0.1 was used for first 50 epochs and 0.01 for second 50 epochs.

Table3: classification accuracy for method with various pooling function and ReLUin output layer

pooling function	Method	Accuracy(%)
Avg-pooling	CNN	98.1
	ThCNN	**98.5**
Max-pooling	CNN	98.4
	ThCNN	98.4

Finale experiment applied a non-linear hyperbolic tangent (tanh) in all feature map , sigmoid in output layer with 50% dropout and the max-pooling function.

Table4: classification error a non-linear hyperbolic tangent (tanh) in all feature map with dropout

Method	Accuracy (%)
CNN	99.3
ThCNN	**99.5**

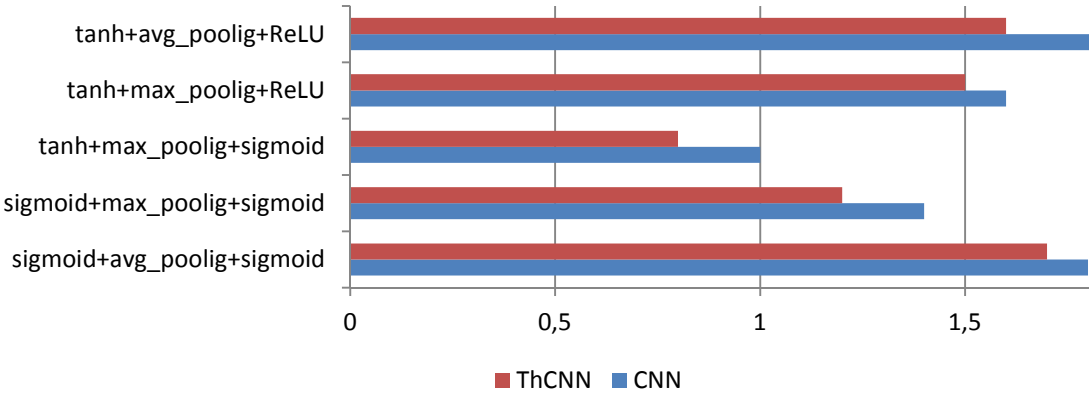

Figure 5 :Test error rates for various pooling function , non-linear function and activation function with each method

5. Conclusions

This paper present a novel deep architecture, ThCNN. AThCNN block consist of a convolutional layer, pooling layer and thresholding layer. We demonstrate that ThCNN is a deep architecture to upgrade over existing deep CNN models. Feature maps extraction showed the effectiveness of categorical representation using a method ofThCNN.In future work, we can apply a thresholding method to perform efficiently, and to run faster than Otsu method. Which needs first to compute a gray-level histogram.

References

1. Y. Bengio, "Learning deep architectures for AI," *Foundations and trends® in Machine Learning,* vol. 2, pp. 1-127, 2009.
2. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.
3. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, pp. 2278-2324, 1998.
4. Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, 2011, p. 5.
5. G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580,* 2012.
6. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research,* vol. 15, pp. 1929-1958, 2014.
7. X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, 2010, pp. 249-256.
8. M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014*, ed: Springer, 2014, pp. 818-833.
9. N. Otsu, "A threshold selection method from gray-level histograms," *Automatica,* vol. 11, pp. 23-27, 1975.
10. L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus, "Regularization of neural networks using dropconnect," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 1058-1066.
11. V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807-814.
12. I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, "Maxout networks," *arXiv preprint arXiv:1302.4389,* 2013.
13. J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao*, et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223-1231.
14. M. C. Lin, Qiang; Yan, Shuicheng, "Network In Network," *arXiv eprint arXiv:1312.4400,* 2014.
15. C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," *arXiv preprint arXiv:1409.5185,* 2014.
16. J.-R. Chang and Y.-S. Chen, "Batch-normalized Maxout Network in Network," *arXiv preprint arXiv:1511.02583,* 2015.
17. M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," *arXiv preprint arXiv:1301.3557,* 2013.
18. R. B. Palm, "Prediction as a candidate for learning deep hierarchical models of data," *Technical University of Denmark,* 2012.