StochasticNet in StochasticNet

Mohammad Javad Shafiee, University of Waterloo, ON, Canada
Paul Fieguth, University of Waterloo, ON, Canada
Alexander Wong, University of Waterloo, ON, Canada

Abstract

Deep neural networks have been shown to outperform conventional state-of-the-art approaches in several structured prediction applications. While high-performance computing devices such as GPUs has made developing very powerful deep neural networks possible, it is not feasible to run these networks on low-cost, low-power computing devices such as embedded CPUs or even embedded GPUs. As such, there has been a lot of recent interest to produce efficient deep neural network architectures that can be run on small computing devices. Motivated by this, the idea of StochasticNets was introduced, where deep neural networks are formed by leveraging random graph theory. It has been shown that StochasticNet can form new networks with 2X or 3X architectural efficiency while maintaining modeling accuracy. Motivated by these promising results, here we investigate the idea of StochasticNet in StochasticNet (SiS), where highly-efficient deep neural networks with Network in Network (NiN) architectures are formed in a stochastic manner. Such networks have an intertwining structure composed of convolutional layers and micro neural networks to boost the modeling accuracy. The experimental results show that SiS can form deep neural networks with NiN architectures that have ∼4X greater architectural efficiency with only a ∼2% drop in accuracy for the CIFAR10 dataset. The results are even more promising for the SVHN dataset, where SiS formed deep neural networks with NiN architectures that have ∼11.5X greater architectural efficiency with only a ∼1% decrease in modeling accuracy.

1 Introduction

Deep learning approaches [1, 2] have been shown to provide state-of-the-arts performance in a number of different computer vision and machine learning problems. In particular, deep neural networks (DNNs) are deep computational models comprised of several computational layers that represent the input data at different levels of abstraction. The advent of high-performance computing devices designed for highly parallel computations such as GPUs have enabled the development of very powerful, complex deep neural networks. However, such networks are often infeasible for use on low-cost, low-power computing devices such as embedded CPUs or even embedded GPUs. Therefore, it is highly desirable to provide efficient deep neural network architectures that facilitate the use of powerful deep neural networks on small, embedded devices. There are two important factors that must be considered when tackling the problem of designing highly-efficient yet powerful deep neural network architectures: I) storage requirement, and II) run-time efficiency.

Designing highly-efficient deep neural network architectures with strong modeling power has been a topic of great interest in recent years. Lecun *et al.* [5] were the first to tackle the problem of network architecture efficiency, which they achieved by dropping synapses based on their strength in the deep neural network. The idea of vector quantization was utilized by Gong *et al.* [6] to compress deep neural networks, thus reducing the storage requirements of deep neural networks. Han *et al.* [8] utilized pruning, quantization, and Humffman coding to further reduce the storage requirements of deep neural networks. Hashing is another trick which has been utilized by Chen *et al.* [9] to compress the network in a smaller amount of storage space.

While such deterministic data compression strategies were successful in greatly reducing the storage space requirements of deep neural networks, the issues of runtime speed and memory requirements were not well addressed using such approaches. As such, highly efficient deep neural networks with low runtime speed and memory requirements as well as low storage requirements are highly desired, as they would enable the use of powerful deep neural networks on low-cost, lower-power embedded CPUs or GPUs for important applications such as driverless cars, drones, and mobile intelligence.

Shafiee *et al.* [4] tackled the aforementioned problem via the concept of StochasticNet, where the architecture of a deep neural network is modeled as a random graph [10]. Motivated by the random synapse formation behavior in the brain and neural plasticity, deep neural networks with highly sparse network architectures are formed by StochasticNet in a stochastic manner. The proposed StochasticNet framework showed that it is possible to form a network with much fewer synaptic connectivities while maintaining the modeling accuracy.

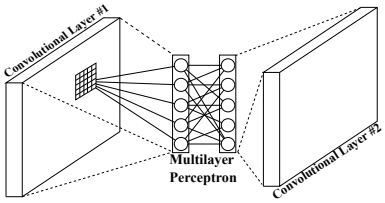

Fig. 1: An example NiN network architecture with two convolutional layers and one micro network in between (in this case, a multilayer perceptron network). The output of linear filters in convolutional layers are transformed to a nonlinear domain via the activation function. This is then fed in to a multilayer perceptron to increase feature discrimination.

Motivated by these promising results, here we investigate the idea of StochasticNet in StochasticNet (SiS), where highly-efficient deep neural networks with Network in Network (NiN) [11, 12] architectures are formed in a stochastic manner. Such networks have an intertwining structure composed of convolutional layers and micro neural networks (such as multilayer perceptrons) to boost the modeling accuracy via additional nested non-linearities.

2 Methodology

The main objective of this study is to investigate and explore the idea of StochasticNet in StochasticNet (SiS), where highly-efficient deep neural networks with NiN architectures are formed in a stochastic manner. First, the idea behind NiN network architectures is explained briefly, followed by a detailed description of the proposed SiS approach.

2.1 Network-in-Network

The Network-in-Network (NiN) [11, 12] architecture is an intertwining structure composed of convolutional layers and micro neural networks, thus providing a more complex nonlinear domain compared to the conventional deep convolutional networks (CNN). More specifically, in a conventional CNN, a convolutional layer is followed by a nonlinear activation function to transform the linear filtering output of to a nonlinear domain. However, in a NiN architecture, the convolutional layer is followed by a micro neural network (with the most commonly used being a multilayer perceptron [11, 12]). The role of this new micro neural network layer is to improve the discriminatory power of the produced features after each convolutional block (i.e., convolutional layer and the activation function). Figure 1 shows an example of the NiN network architecture. Effectively, the micro neural network layer transforms the convolutional features to a more complex nonlinear domain, which are then fed into the next convolutional layer in the network.

2.2 StochasticNet in StochasticNet

Motivated by the increased modeling capabilities gained from NiN architectures, we now introduce the concept of StochasticNet in StochasticNet (SiS), which can be described as follows. Let the network architecture of a deep neural network be formulated as a random graph $G(V,E)$. Each $e_{i,j} \in E$ represents a synapse in the network $G(\cdot)$. The formation of synaptic connectivity $e_{i,j}$ is determined via $P(s_{i,j}=1|\Theta)$, where $s_{i,j}=1$ indicates the formation of $e_{i,j}$ in the network architecture and Θ encodes the synaptic probability distribution. For illustrative purposes, the synaptic probability distribution is formulated here based on a uniform distribution (i.e., Θ is a uniform distribution).

To formulate the underlying random graph of the proposed SiS framework, the synaptic connectivities are grouped into two different types: I) the synaptic connectivity in a convolutional layer, $e^{c}_{i,j}$,

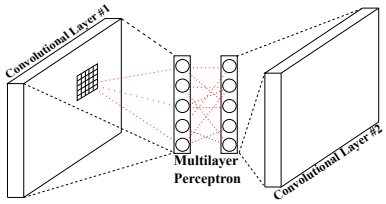

Fig. 2: An example SiS network architecture with two convolutional layers and one micro network in between (in this case, a multilayer perceptron network). The synaptic connectivities in the SiS network are a subset of synaptic connectivities of the NiN network shown in Figure 1.

and II) the synaptic connectivity in a micro neural network layer, $e^{p}_{k,l}$. Since these two types of synaptic connectivities play two distinct roles, the associated probability distributions are defined in different ways. Here, we utilize uniform distribution for both convolutional layers and micro neural network layers, but the parameters of the distributions are selected differently,

$$\begin{aligned} e^{c}_{i,j} \quad &\text{exists if} \quad P(s_{i,j}=1|\Theta)=U(0,t^{c}) \\ e^{c}_{i,j} \quad &\text{exists if} \quad P(s_{i,j}=1|\Theta)=U(0,t^{p}) \end{aligned} \tag{1}$$

where $U(0,t^{c})$ is a uniform distribution in the range of $[0,t^{c}]$. $t^{c} \leq 1$ and $t^{p} \leq 1$ define the uniform distributions. These distributions also implicitly determine the portion of synaptic connectivities that can be formed in the deep neural network in convolutional layers, and in micro neural network layers, respectively.

Figure 2 shows an example of a formed SiS network architecture.

3 Results & Discussion

Two benchmark datasets are used to examine the performance of the proposed SiS framework for forming highly efficient yet powerful deep neural networks. The CIFAR10 image dataset [14] consists of 50,000 training images categorized into 10 different classes of natural scenes. Each image is an RGB image that is 32×32 in size. The SVHN image dataset [13] comprises of 604,388 training images and 26,032 test images of digits in natural scenes. The images of this dataset are also RGB with 32×32 in size.

The network architecture introduced in [11] is utilized as a full network structure (NiN) and the proposed SiS is formed based on that architecture. The network is comprised of, C1: 192@5×5, MP1: 160@1×1 and 96@1×1, C2: 192@5×5, MP2: 192@1×1 and 192@1×1, C3: 192@3×3 and MP3: 160@1×1 and 10@1×1.

Tables 1 and 2 show the architecture efficiency versus the modeling accuracy of three different sparse deep neural networks formed using the proposed SiS framework with different levels of sparsity, compared to a reference deep neural network with a NiN architecture with all available synaptic connectivities. As observed in Table 1, the SiS framework was able to generate highly-efficient deep neural networks with NiN architectures that possess up to ∼4X greater network efficiency while the modeling accuracy drops by only 2% for the CIFAR10 dataset. Table 2 demonstrates the modeling accuracy of three different sparse deep neural networks formed using the proposed SiS framework with different levels of sparsity for the SVHN dataset. As evident by the results in Table 2, SiS was able to form highly efficient deep neural networks with NiN architectures with up to 11.5X greater efficiency with only 1% in modeling accuracy. It is worth noting that the network formation does not utilize any prior knowledge about the synaptic strengths.

Table 1: Architectural efficiency vs. test accuracy for different formed networks for CIFAR dataset. " #Synap. in CL" shows the number of synaptic connectivities in convolutional layers while "#Synap. in MP" demonstrates the number of synaptic connectivities in multilayer perceptron blocks. The column "Arch. Eff." shows the network efficiency compared to the NiN architecture.

Network	#Synap. in CL	#Synap. in MP	Total Synp.	Arch. Eff.	Accuracy
NiN	806976	152128	959104	1	0.8894
SiS#1	201522	152128	353650	2.71	0.8872
SiS#2	203475	63008	266483	3.60	0.8745
SiS#3	212544	28240	240784	3.98	0.8646

Table 2: Architectural efficiency vs. test accuracy for different formed networks for SVHN dataset.

Network	#Synap. in CL	#Synap. in MP	Total Synp.	Arch. Eff.	Accuracy
NiN	806976	152128	959104	1	0.9499
SiS#1	203697	152128	355825	2.69	0.9459
SiS#2	207618	63008	270626	3.54	0.9435
SiS#3	54048	28240	82288	11.65	0.9342

4 Conclusion

We demonstrate the efficacy of the proposed StochasticNet in StochasticNet framework in forming highly efficient deep neural networks with complex NiN architectures that possess significantly fewer synaptic connectivities. Results showed that the proposed StochasticNet in StochasticNet approach can form very sparse networks with NiN architecture that have significantly greater network efficiencies while retaining modeling capabilities.

Acknowledgments

This work was supported by the Natural Sciences and Engineering Research Council of Canada, Canada Research Chairs Program, and the Ontario Ministry of Research and Innovation. The authors also thank Nvidia for the GPU hardware used in this study through the Nvidia Hardware Grant Program.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[2] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE international conference on acoustics, speech and signal processing*, 2013, pp. 6645–6649.

[3] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *CoRR, abs/1412.6115*, 2014.

[4] M. J. Shafiee, P. Siva, and A. Wong, "Stochasticnet: Forming deep neural networks via stochastic connectivity," *IEEE Access*, vol. 4, pp. 1915–1924, 2016.

[5] Y. LeCun, J. S. Denker, S. A. Solla, R. E. Howard, and L. D. Jackel, "Optimal brain damage." in *Advances in Neural Information Processing Systems (NIPS)*, 1989.

[6] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *CoRR, abs/1412.6115*, 2014.

[7] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *CoRR, abs/1510.00149*, 2015.

[8] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[9] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," *CoRR, abs/1504.04788*, 2015.

[10] P. Erdos and A. Renyi, "On random graphs i," *Publ. Math. Debrecen*, vol. 6, pp. 290–297, 1959.

[11] Z. Liao, and G. Carneiro, "On the importance of normalisation layers in deep learning with piecewise linear activation units," *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016.

[12] M. Lin, Q. Chen, and S. Yan, "Network in network", *International Conference on Learning Representations (ICLR)*, 2013.

[13] 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*, volume 2011, page 5. Granada, Spain, 2011.

[14] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009.