CS231N Project Proposal
Deep Neuroevolution with Shared Parameters: Ensembling a Better Solution

Michael Painter
Stanford University
mp703@stanford.edu

1. Problem Introduction

A lot of work has gone into designing neural network architectures, however, often they are still difficult to design and many architectures are tried and tested to select the best one for a task. Automating this process leads to the notion of a neural architecture search [9, 19, 18]. Naively implemented, this involves training hundreds of models from scratch, typically requiring hundreds of GPUs. This can be improved by using transfer learning (via function preserving transforms [1, 2]) and sharing parameters between models searching over [8]. Moreover, likely from computational brute force, architecture searches tend to lead to state of the art performance [19].

2. Methods And Algorithms

In this project, we wish to explore performing an architecture search, using neuroevolution to search the "architecture space". We will share parameters between models, similar to the efficient architecture search described by Pham et al. [8], allowing a single GPU to be used in training, and new models will be created using function preserving transforms, extended from the works of Chen et al. [2] and Cai et al. [1], so that models need not be trained from scratch.

This project will require new versions of the transforms defined by Chen et al. [2], and will consist of adding modules into Incpetion networks, initialized such that it doesn't alter the overall function the network represents, which is outlined in figure 1.

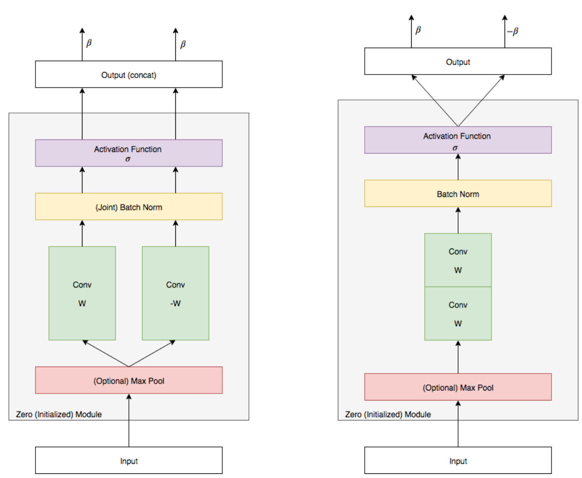

Figure 1. Left: Our initial concept for a zero initialized module. Adding two convolutional filters, one initialized with weights W and the other with weights $-W$. The split arrows represent duplication here. Right: Altered zero initialized module, so that it can be represented as a single filter, and also allows for non-symmetry in the activation function, weights β and $-\beta$ must be used on the output, to provide the "zero output". The split arrows represent splitting according to the two sets of filters here.

At the end, of training, we will incorporate multiple of the learned networks into an ensemble model. If time, we will investigate methods of maintaining a diverse population of networks (i.e. each network is good at classifying some classes that other networks are not), so that when combined in a ensemble model, it gives greater performance.

3. Data

We will be use Imagenet [3] to evaluate the performance of the algorithms and architectures, which is openly available.

4. Evaluation

Our evaluation metric will be classification accuracy on the test set of Imagenet [3]. As we are looking to perform an architecture search efficiently, without requiring hundreds of GPUs, we also will analyze the computational efficiency. Specifically, we should compare the number of floating point operation (FLOPs) (rather than training time or epochs) with respect to training and test accuracies between models. In particular, for this comparison, we would look to compare training a single network, an ensemble of networks and our method.

5. Reading

We have already undertaken a reasonable amount of reading for this project, however, in total, we have identified the following topics as important to survey prior to implementation. Here is a brief list of what will/has be surveyed:

- Architecture searches [9, 19, 18],
- Efficient Architecture searches [1, 8],
- Transfer Learning and Function Preserving Transformations [2, 5],
- Neuroevolution [4, 9, 12, 11, 14],
- State of the art CNN architectures [7, 13, 15],
- State of the art image classification [6, 10, 16, 15, 17, 19].

References

[1] H. Cai, T. Chen, W. Zhang, Y. Yu, and J. Wang. Reinforcement learning for architecture search by network transformation. *arXiv preprint arXiv:1707.04873*, 2017.

[2] T. Chen, I. Goodfellow, and J. Shlens. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*, 2015.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[4] S. E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In *Advances in neural information processing systems*, pages 524–532, 1990.

[5] S. Gutstein, O. Fuentes, and E. Freudenthal. Knowledge transfer in deep convolutional neural nets. *International Journal on Artificial Intelligence Tools*, 17(03):555–567, 2008.

[6] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*, 2017.

[7] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, page 3, 2017.

[8] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean. Efficient neural architecture search via parameter sharing. *arXiv preprint arXiv:1802.03268*, 2018.

[9] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, Q. Le, and A. Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017.

[10] S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3859–3869, 2017.

[11] K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial life*, 15(2):185–212, 2009.

[12] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[13] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.
[14] S. Whiteson. Reinforcement learning state of the art. *Adaptation, learning, and optimization*, 12:325–355, 2012.
[15] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5987–5995. IEEE, 2017.
[16] Y. Yamada, M. Iwamura, and K. Kise. Shakedrop regularization. *arXiv preprint arXiv:1802.02375*, 2018.
[17] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.
[18] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.
[19] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. *arXiv preprint arXiv:1707.07012*, 2017.