# Towards an explorative self-learning builder of deep-learning-networks for object detection

Md Ibrahim Khalil, Md Omor Faruk, Shahriar Real

February 15, 2019

Deep learning has provided an unprecedented improvement in object recognition performance compared to prior state-of-the-art machine learning algorithms. Thanks to myriads of excellent works, some deep learning methods have now exceeded human level performance in some object detection tasks [2] and have opened research directions in hundreds of other object detection applications.

Despite these leaps that recent deep learning works have provided, much of the current research has now narrowed down to using pre-trained weights and architectures of deep learning models that have outperformed others. We argue that this narrowed down general trend of research can be an impediment to an optimal progress in finding optimal network structures. Moreover, we also argue that further exploration is still needed to tackle some failures in deep-learning based object detection tasks - e.g. failure to detect object when only few pixels in an image are changed [4], failure to capture hierarchical shape structures [3], suboptimal object detection accuracy in many applications [1], etc. In [4], they proposed a methodology for generating single-pixel adversarial perturbations. Their basis was differential evolution (DE) for adversarial machine learning. In quest of achieving robustness, they purported to create tools and methodologies for low cost adversarial attacks on neural networks. The dynamic routing between capsules was reported in [3], where, highly overlapped digits were successfully figured out even better than Convolutional Neural Network (CNN). They assumed, an object can be understood by some activity vectors representing a group of neurons also known as capsules. To train MNIST dataset, they utilized several layer capsule system and the result was noteworthy. Quo Vadis in [1] instantiated spatio-temporal feature extraction by tuning imagenet architecture. They worked on kinetics dataset and demonstrated improved action classificaiton.

In our experience, in most deep learning-related works we have found an alarming lack of both explanations and experiments that have sufficiently explored alternative network structures.

Therefore, in this work, we propose an algorithm that dynamically learns to build an optimal network structure and provide summarized experiment results that provide recommendations for best to worst network structure building practices that are specific to a particular dataset. We believe that this toolset will be particularly valuable to researchers in their early exploration of a dataset and in building their network architectures.

Algorithm -

1. Start with input layer, and at the end an average global pooling layer and a fully connected layer.
2. Objective: Find an optimal network structure to be placed in between the input layer and the pooling layers in the end.
    - Decision 1 (for each layer): convolutional layer or fully connected layer; with or without relu, sigmoid, etc.
    - Decision 2: increase width or depth
    - Decision 3 (for each inter-connections between layers): type of connection: weighted or non-weighted addition, multiplication, exponent, etc.; with or without relu, sigmoid, etc.
3. Decisions will be chosen based on random sampling of a probability distribution

4. The probability distribution of each decision path will change based on del(performance) resulting from taking a particular decision path

Fortunately, Pytorch is a python based scientific computing package, where for each forward pass, it is possible to choose any number that has many hidden layers. As a consequence, innermost hidden layers can be computed by using similar weights more than one times. For example, for a forward propagation, if the depth increment of an amount can reduce the loss 3 times and width increment can reduce loss 4 times, it is possible to optimize loss leveraging the dynamic behavior of Pytorch. Which means, considering several situations inside a loop and performing the computation. By checking train and validation accuracy, we can do that.

Initially, we will use the famous MNIST dataset and test all the aforementioned on that. MNIST dataset contains 70,000 images and 784 (28x28) greyscale images (`https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d`). Upon receiving noteworthy outcomes, we will start testing that on CIFAR-10 dataset too. CIFAR-10 is an established computer-vision dataset used for object recognition. It is a subset of the 80 million tiny images dataset and consists of 60,000 (32x32) color images containing one of 10 object classes, with 6000 images per class (according to `http://www.kaggle.com/c/cifar-10`)

# References


[1] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 4724–4733. IEEE, 2017.

[2] Samuel Dodge and Lina Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *Computer Communication and Networks (ICCCN), 2017 26th International Conference on*, pages 1–7. IEEE, 2017.

[3] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3856–3866, 2017.

[4] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.