SMASH: One-Shot Model Architecture Search Through HyperNetworks

Authors: Andrew Brock, Theodore Lim, J.M. Ritchie, and Nick Weston

April 14, 2018

Presentation by Kamal Rai

When training neural networks, we:

- Fix the network architecture
- Specify a loss function $\mathcal{L}$
- Find optimal weights W using backprop to minimize $\frac{d\mathcal{L}}{dW}$

Iterate over design decisions until we obtain a good model

Model hyperparameters: Depth, width, connectivity

The Motivation

Finding optimal architectures requires extensive experimentation

Current automated architecture selection methods are expensive

Evolutionary techiniques and reinforcement learning

Given randomly sampled hyperparameters c, we can iteratively:

1. Optimize the weights of an auxilary network using $\frac{\partial \mathcal{L}(w_c)}{\partial W_c} \frac{\partial W_c}{\partial c}$
2. Optimize the weights of the main network

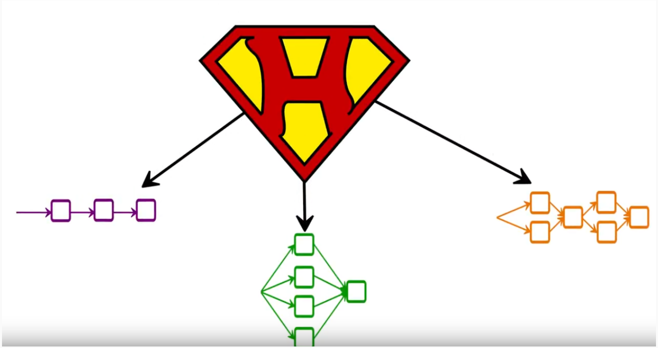

Figure 1: Generate weights using an auxilary network

The Training Algorithm

Algorithm 1: SMASH

Input: Space of all candidate architectures $\mathbb{R}_c$

Initialize HyperNet weights H

loop

Sample input minibatch x_i, random architecture c, and architecture weights $W(c)$

Get training error $E_t = f_c(W, x_i) = f_c(H(c), x_i)$, backprop $\frac{dE}{dW}$ through the HyperNet and then update H

end loop

loop

Sample a random architecture c and evaluate error on validation set $E_v = f_c(H(c), x_v)$

end loop

Fix architecture and train normally with freely-varying weights W

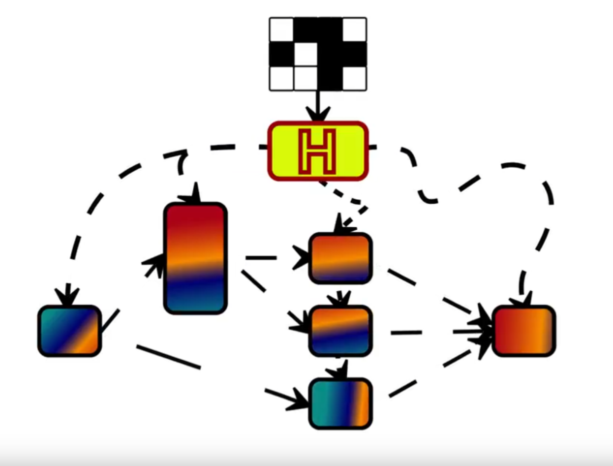

Figure 2: Sampling from a hypernetwork

Ranking Candidate Models

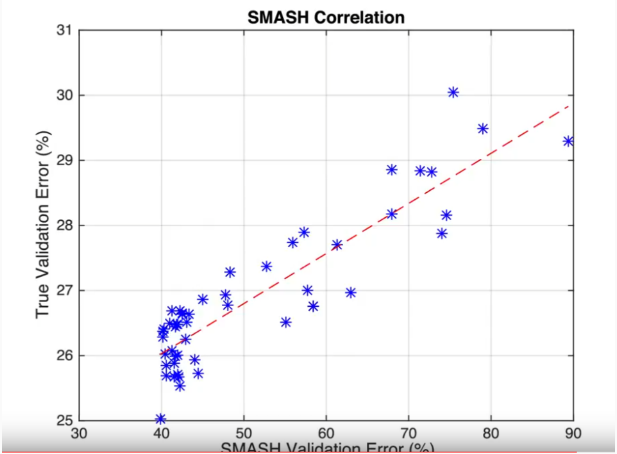

Figure 3: Exploring performance on CIFAR-100

The strength of correlation depends on

- The capacity of the hypernet
- The ratio of hypernet generated weights to freely learned weights

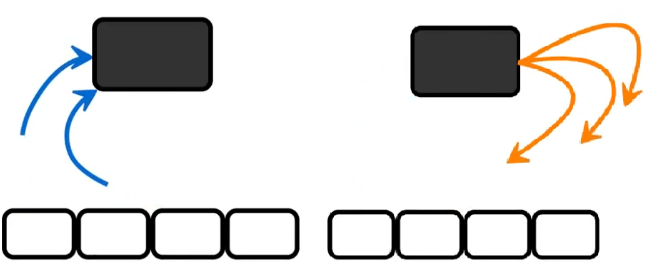

Figure 4: Layers are ops that read and write to memory

Table 1: Error rates (%) on CIFAR-10 and CIFAR-100 with standard data augmentation (+).

Method	Depth	Params	C10+	C100+
FractalNet [20]	21	38.6M	5.22	23.30
with Dropout/Drop-path	21	38.6M	4.60	23.73
Wide ResNet [43]	16	11.0M	4.81	22.07
	28	36.5M	4.17	20.50
DenseNet-BC ($k = 24$) [15]	250	15.3M	3.62	17.60
DenseNet-BC ($k = 40$)	190	25.6M	3.46	17.18
Shake-Shake [11]	26	26.2M	**2.86**	**15.85**
Neural Architecture Search w/ RL[44]	39	32.0M	3.84	-
MetaQNN [3]	9	11.18M	6.92	27.14
Large-Scale Evolution [26]	-	5.4M	5.40	-
	-	40.4 M	-	23.7
CGP-CNN [38]	-	1.68M	5.98	-
SMASHv1	116	4.6M	5.53	22.07
SMASHv2	211	16M	4.03	20.60

Figure 5: Benchmark results

Limitations

- The space of candidate architectures must be pre-specified
- Does not address regularization or learning rate
- Not jointly training the hypernet and the main network
- Not using gradients to optimize the choice of main network

Conclusion

Can efficiently explore architectures using Hypernet weights

Two Related Works

- Hyperparameter Optimization with Hypernets. J. Lorraine and D. Duvenaud
- Hyper-bandit: Bandit-based Configuration Evaluation for Hyperparameter Optimization. L. Li, K. Jamieson, and G. DeSalvo