AutoAugment
(2018)

Ekin D. Cubuk et al.
Resume

May 7, 2019

1 Introduction

AutoAugment automatically search for improved data augmentation policies, with a search space where a policy consists of many sub-policies, one of which is randomly chosen for each image in each mini-batch, a sub-policy consists of two operations, each operation being an image processing function such as translation, rotation, or shearing, and the probabilities and magnitudes with which the functions are applied. A search algorithm is used to find the best policy such that the neural network yields the highest validation accuracy on a target dataset.

2 AutoAugment

The problem of finding the best augmentation policy is formulated as a discrete search problem, the method consists of two components: A search algorithm and a search space. At a high level, the search algorithm (implemented as a controller RNN) samples a data augmentation policy S, which has information about what image processing operation to use, the probability of using the operation in each batch, and the magnitude of the operation, the policy S willbe used to train a neural network with a fixed architecture, whose validation accuracy R will be sent back to update the controller. Since R is not differentiable, the controller will be updated by policy gradient methods.

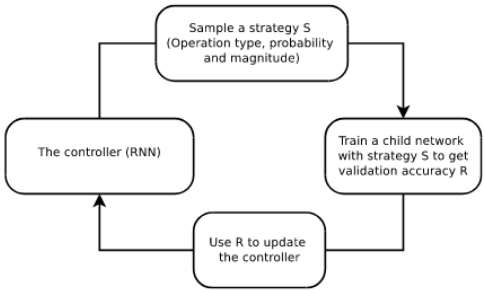

Figure 1. Overview of our framework of using a search method (e.g., Reinforcement Learning) to search for better data augmentation policies. A controller RNN predicts an augmentation policy from the search space. A child network with a fixed architecture is trained to convergence achieving accuracy R. The reward R will be used with the policy gradient method to update the controller so that it can generate better policies over time.

Search space a policy consists of 5 sub-policies with each subpolicy consisting of two image operations to be applied in sequence. Additionally, each operation is also associated with two hyperparameters: 1) the probability of applying the operation, and 2) the magnitude of the operation, the operation used are: ShearX/Y,TranslateX/Y, Rotate, AutoContrast, Invert, Equalize, Solarize, Posterize, Contrast, Color, Brightness, Sharpness, Cutout, Sample Pairing, a total of 16 operations,

the range of magnitudes is discretized into 10 values, and the probability of applying a given policy is discretized into 11 values, so each sub policy is a search problem of $(16 \times 10 \times 11)$ possibilities, the goal is to find 5 sub-policies, so the search space is $(16 \times 10 \times 11)^{10} \approx 2.9 \times 10^{32}$ possibilites.

Search algorithm The search algorithm has two components: a controller, which is a RNN, and the training algorithm, which is the Proximal Policy Optimization algorithm. At each step, the con-troller predicts a decision produced by a softmax; the pre-diction is then fed into the next step as an embedding. In total the controller has 30 softmax predictions in order topredict 5 sub-policies, each with 2 operations, and each operation requiring an operation type, magnitude and probability $(30 = 5 * (2 + 4))$.

Training the RNN The controller is trained with a reward signal, which is how good the policy is in improving the generalization of a child model (a neural network trained as part of the search process), a validation set us used to measure the generalization of a child model. A child model is trained with augmented data generated by applying the 5 sub-policies on the training set. For each example in the mini-batch, one of the 5 sub-policies is chosen randomly to augment the image. The child model is then evaluated on the validation set to measure the accuracy, which is used as the reward signal to train the RNN controller. On each dataset, the controller samples about 15,000 policies.

3 results

Dataset	Model	Baseline	Cutout [12]	AutoAugment
CIFAR-10	Wide-ResNet-28-10 [67]	3.9	3.1	2.6±0.1
	Shake-Shake (26 2x32d) [17]	3.6	3.0	2.5±0.1
	Shake-Shake (26 2x96d) [17]	2.9	2.6	2.0±0.1
	Shake-Shake (26 2x112d) [17]	2.8	2.6	1.9±0.1
	AmoebaNet-B (6,128) [48]	3.0	2.1	1.8±0.1
	PyramidNet+ShakeDrop [65]	2.7	2.3	**1.5 ± 0.1**
Reduced CIFAR-10	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	**10.0 ± 0.2**
CIFAR-100	Wide-ResNet-28-10 [67]	18.8	18.4	17.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	16.0	14.3±0.2
	PyramidNet+ShakeDrop [65]	14.0	12.2	**10.7 ± 0.2**
SVHN	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	**1.0**
Reduced SVHN	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	**5.9**

Table 2. Test set error rates (%) on CIFAR-10, CIFAR-100, and SVHN datasets. Lower is better. All the results of the baseline models, and baseline models with Cutout are replicated in our experiments and match the previously reported results [67, 17, 65, 12]. Two exceptions are Shake-Shake (26 2x112d), which has more filters than the biggest model in [17] – 112 vs 96, and Shake-Shake models trained on SVHN, these results were not previously reported. See text for more details.

Model	Inception Pre-processing [59]	AutoAugment ours
ResNet-50	76.3 / 93.1	77.6 / 93.8
ResNet-200	78.5 / 94.2	80.0 / 95.0
AmoebaNet-B (6,190)	82.2 / 96.0	82.8 / 96.2
AmoebaNet-C (6,228)	83.1 / 96.1	**83.5 / 96.5**

Table 3. Validation set Top-1 / Top-5 accuracy (%) on ImageNet. Higher is better. ResNet-50 with baseline augmentation result is taken from [20]. AmoebaNet-B,C results with Inception-style pre-processing are replicated in our experiments and match the previously reported result by [48]. There exists a better result of 85.4% Top-1 error rate [37] but their method makes use of a large amount of weakly labeled extra data. Ref. [68] reports an improvement of 1.5% for a ResNet-50 model.