

# Paper Reading No.13

AutoAugment: Learning Augmentation Strategies from Data

Sheng Lian

September 2019

## 1 Brief Paper Intro

- **Paper ref:** CVPR 2019 , [CVF link](#).
- **Authors:** Ekin D. Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, Quoc V. Le from *Google Brain*
- **Paper summary:** This paper proposed a automatically data augmentation policies. This paper use a RL-based search algorithm to find the best policy such that the neural network yields the highest validation accuracy on target datasets. This work has two use cases: 1)AutoAugment can be applied directly on the dataset of interest to find the best augmentation policy (AutoAugment-direct) and 2) learned policies can be transferred to new datasets (AutoAugment-transfer).
- **Reading motivation:** The choice of data augmentation methods is a issue that worth discussing. When we did research on medical imaging, how to choose the way to augment data is so tricky. We usually

use methods such as flipping, rotation, shearing, etc. However, few researchers did an ablation experiment to find the 'best' combination of augmentation method. Even when augmentation improvements have been found for a particular dataset, they often do not transfer to other datasets as effectively. Let's see how this work do.

## 2 Methods

### 2.1 AutoAugment: Searching Directly

The proposed method consists of two components: a controller RNN for searching, and a search space.

#### 2.1.1 Search space details

In the search space, a policy consists of 5 sub-policies with each sub-policy 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.

As can be seen in Fig 1.

In total, the proposed method have 16 operations in the search space, including ShearX/Y, Rotate, AutoContrast, etc. The range of magnitudes is 10 values, and the probability have 11 values ([0:1]). Both of them is uniform spacing. So for each sub-policy, the searching space is  $(16 * 10 * 11)^2$ . The paper use 5 sub-policies, so the searching space is  $(16 * 10 * 11)^{10}$ . It's a rather large number. So it's time for reinforce learning to search for the best choice!



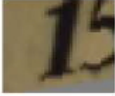



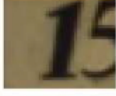
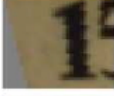
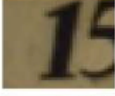
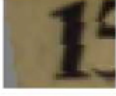
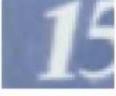

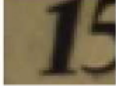
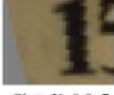
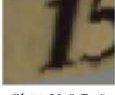

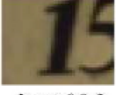
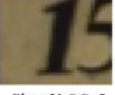
	Original	Sub-policy 1	Sub-policy 2	Sub-policy 3	Sub-policy 4	Sub-policy 5
Batch 1						
Batch 2						
Batch 3						
		ShearX, 0.9, 7 Invert, 0.2, 3	ShearY, 0.7, 6 Solarize, 0.4, 8	ShearX, 0.9, 4 AutoContrast, 0.8, 3	Invert, 0.9, 3 Equalize, 0.6, 3	ShearY, 0.8, 5 AutoContrast, 0.7, 3

Figure 1: One of the policies found on SVHN

### 2.1.2 Search algorithm

This part is not very detailed introduced, because the main idea is similar to Google Brain’s former work *NASNet* [?]. The overview of the proposed framework is shown in Fig 2. 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.

The controller utilizes softmax to output which policy to apply. This decision is then passed as input to the controller’s next step because the controller is an RNN. The controller’s training scheme is similar to NASNet [?]. The controller then decides which magnitude to apply for the operation. The third step is to choose the probability.

The procedure introduced above is described in Fig 3.

In total the controller has 30 softmax predictions in order to predict 5 sub-policies, each with 2 operations, and each operation requiring an operation type, magnitude and probability.

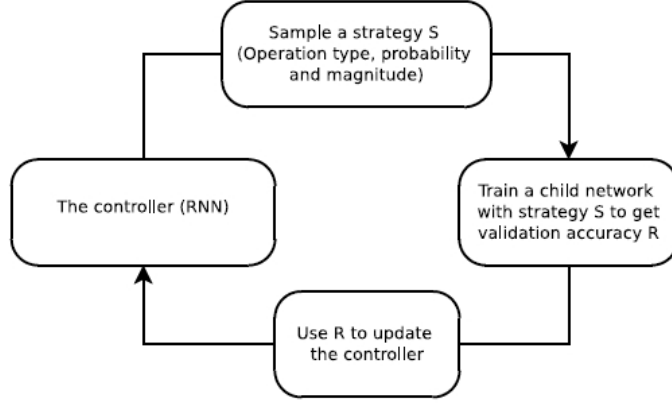


Figure 2: Overview of our framework of using a search method.

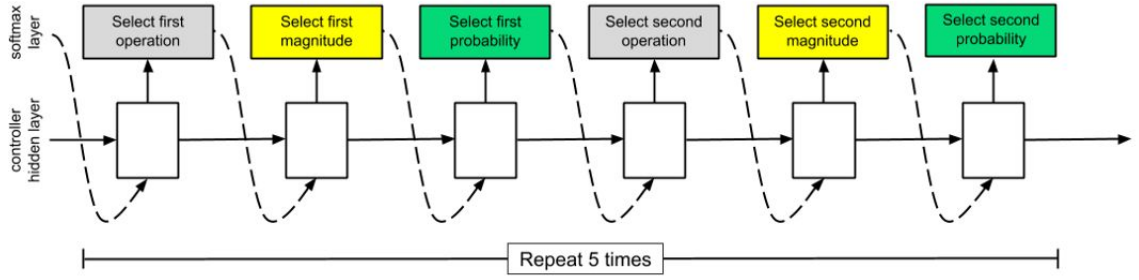


Figure 3: The architecture of controller RNN.

### 3 Experiments

Authors evaluate AutoAugment directly on several datasets, including CIFAR-10, CIFAR-100, SVHN, and ImageNet, etc. The test set error rates on the mentioned datasets are described in

Dataset	Model	Baseline	Cutout [12]	AutoAugment
<b>CIFAR-10</b>	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	<b>1.5 ± 0.1</b>
<b>Reduced CIFAR-10</b>	Wide-ResNet-28-10 [67]	18.8	16.5	14.1±0.3
	Shake-Shake (26 2x96d) [17]	17.1	13.4	<b>10.0 ± 0.2</b>
<b>CIFAR-100</b>	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	<b>10.7 ± 0.2</b>
<b>SVHN</b>	Wide-ResNet-28-10 [67]	1.5	1.3	1.1
	Shake-Shake (26 2x96d) [17]	1.4	1.2	<b>1.0</b>
<b>Reduced SVHN</b>	Wide-ResNet-28-10 [67]	13.2	32.5	8.2
	Shake-Shake (26 2x96d) [17]	12.3	24.2	<b>5.9</b>

Figure 4: Error rate comparison.

### 3.1 The Transferability of Learned Augmentation policies to Other Datasets

The previous text introduced the directly application of AutoAugment to find the augmentation policies on the dataset of interest. However, such application is so time-consuming and resource-consuming. So, this paper try to transfer augmentation policies from one dataset to another.

Here, authors use the same policy learned on ImageNet on other dataset.

The AutoAugment-transfer is effective even on datasets for which fine-tuning weights pre-trained on ImageNet does not help significantly. This means that AutoAugment doesn't overfit to a specific dataset.

Dataset	Train Size	Classes	Baseline	AutoAugment-transfer
Oxford 102 Flowers [43]	2,040	102	6.7	<b>4.6</b>
Caltech-101 [15]	3,060	102	19.4	<b>13.1</b>
Oxford-IIIT Pets [14]	3,680	37	13.5	<b>11.0</b>
FGVC Aircraft [38]	6,667	100	9.1	<b>7.3</b>
Stanford Cars [27]	8,144	196	6.4	<b>5.2</b>

Figure 5: AutoAugment transfer on some datasets.

## 4 Summarize

Authors also compare their proposed AutoAugment with other automated data augmentation methods and find their method superior to others. Also, the number of policies and the randomize of probability and magnitudes are discussed. Overall, the experiments are solid and the performances are convincing. This paper worth the 'oral paper' title.