Automated Data Augmentation with AutoAugment and RandAugment

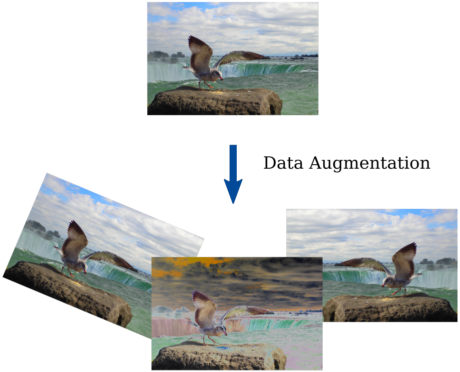
Diane Wagner

Based on: Cubuk *et al.* [2019]

Seminar on Current Works in Computer Vision
Advisor: Sudhanshu Mittal
22/01/2020

Data augmentation

- Deep learning: It's all about data!
- Data augmentation may improve:
  > Accuracy
  > Model robustness
  > Generalization

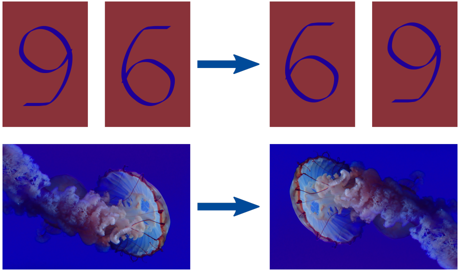


Automated data augmentation

- An optimal augmentation strategy depends on the dataset

- Manual selection:
  > Time-consuming
  > Tedious
  > Sub-optimal
  > Requires expert knowledge

Transformation: 180° rotation

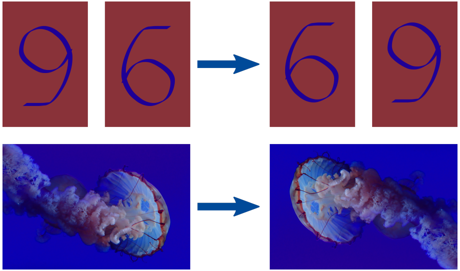


Automated data augmentation

- An optimal augmentation strategy depends on the dataset
- Manual selection:
  - > Time-consuming
  - > Tedious
  - > Sub-optimal
  - > Requires expert knowledge

$\Rightarrow$ High interest in automating this task

Transformation: 180° rotation



Overview

- AutoAugment
- RandAugment
- Discussion

arXiv:1805.09501v3 [cs.CV] 11 Apr 2019

AutoAugment:
Learning Augmentation Strategies from Data

Ekin D. Cubuk *, Barret Zoph*, Dandelion Mané, Vijay Vasudevan, Quoc V. Le
Google Brain

Abstract

Data augmentation is an effective technique for improving the accuracy of modern image classifiers. However, current data augmentation implementations are manually designed. In this paper, we describe a simple procedure called AutoAugment to automatically search for improved data augmentation policies. In our implementation, we have designed 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. We use a search algorithm to find the best policy such that the neural network yields the highest validation accuracy on a target dataset. Our method achieves state-of-the-art accuracy on CIFAR-10, CIFAR-100, SVHN, and ImageNet (without additional data). On ImageNet, we attain a Top-1 accuracy of 83.5% which is 0.4% better than the previous record of 83.1%. On CIFAR-10, we achieve an error rate of 1.5%, which is 0.6% better than the previous state-of-the-art. Augmentation policies we find are transferable between datasets. The policy learned on ImageNet transfers well to achieve significant improvements on other datasets, such as Oxford Flowers, Caltech-101, Oxford-IIT Pets, FGVC Aircraft, and Stanford Cars.

1. Introduction

Deep neural nets are powerful machine learning systems that tend to work well when trained on massive amounts of data. Data augmentation is an effective technique to increase both the amount and diversity of data by randomly "augmenting" it [3, 54, 29]; in the image domain, common augmentations include translating the image by a few pixels, or flipping the image horizontally. Intuitively, data augmentation is used to teach a model about invariances in the data domain: classifying an object is often insensitive to horizontal flips or translation. Network architectures can also be used to hardcode invariances: convolutional networks bake in translation invariance [16, 32, 25, 29]. However, using data augmentation to incorporate potential invariances can be easier than hardcoding invariances into the model architecture directly.

Dataset	GPU hours	Best published results	Our results
CIFAR-10	5000	2.1	1.5
CIFAR-100	0	12.2	10.7
SVHN	1000	1.3	1.0
Stanford Cars	0	5.9	5.2
ImageNet	15000	3.9	3.5

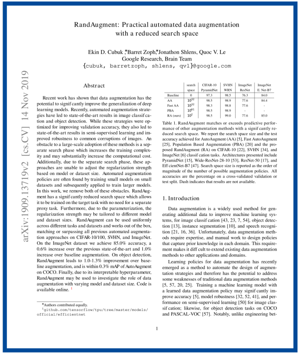
Table 1. Error rates (%) from this paper compared to the best results so far on five datasets (Top-5 for ImageNet, Top-1 for the others). Previous best result on Stanford Cars fine-tuned weights originally trained on a larger dataset [66], whereas we use a randomly initialized network. Previous best results on other datasets only include models that were not trained on additional data, for a single evaluation (without ensembling). See Tables 2,3, and 4 for more detailed comparison. GPU hours are estimated for an NVIDIA Tesla P100.

Yet a large focus of the machine learning and computer vision community has been to engineer better network architectures (e.g., [55, 59, 20, 58, 64, 19, 72, 23, 48]). Less attention has been paid to finding better data augmentation methods that incorporate more invariances. For instance, on ImageNet, the data augmentation approach by [29], introduced in 2012, remains the standard with small changes. Even when augmentation improvements have been found for a particular dataset, they often do not transfer to other datasets as effectively. For example, horizontal flipping of images during training is an effective data augmentation method on CIFAR-10, but not on MNIST, due to the different symmetries present in these datasets. The need for automatically learned data-augmentation has been raised recently as an important unsolved problem [57].

In this paper, we aim to automate the process of finding an effective data augmentation policy for a target dataset. In our implementation (Section 3), each policy expresses several choices and orders of possible augmentation opera-

*Equal contribution.

1



arXiv:1909.13719v2 [cs.CV] 14 Nov 2019

RandAugment: Practical automated data augmentation
with a reduced search space

Ekin D. Cubuk *, Barret Zoph*, Jonathon Shlens, Quoc V. Le
Google Research, Brain Team
{cubuk, barretzoph, shlens, qvl}@google.com

Abstract

Recent work has shown that data augmentation has the potential to significantly improve the generalization of deep learning models. Recently, automated augmentation strategies have led to state-of-the-art results in image classification and object detection. While these strategies were optimized for improving validation accuracy, they also led to state-of-the-art results in semi-supervised learning and improved robustness to common corruptions of images. An obstacle to a large-scale adoption of these methods is a separate search phase which increases the training complexity and may substantially increase the computational cost. Additionally, due to the separate search phase, these approaches are unable to adjust the regularization strength based on model or dataset size. Automated augmentation policies are often found by training small models on small datasets and subsequently applied to train larger models. In this work, we remove both of these obstacles. RandAugment has a significantly reduced search space which allows it to be trained on the target task with no need for a separate proxy task. Furthermore, due to the parameterization, the regularization strength may be tailored to different model and dataset sizes. RandAugment can be used uniformly across different tasks and datasets and works out of the box, matching or surpassing all previous automated augmentation approaches on CIFAR-10/100, SVHN, and ImageNet. On the ImageNet dataset we achieve 85.0% accuracy, a 0.6% increase over the previous state-of-the-art and 1.0% increase over baseline augmentation. On object detection, RandAugment leads to 1.0-1.3% improvement over baseline augmentation, and is within 0.3% mAP of AutoAugment on COCO. Finally, due to its interpretable hyperparameter, RandAugment may be used to investigate the role of data augmentation with varying model and dataset size. Code is available online. [1]

	search space	CIFAR-10 PyramidNet	SVHN WRN	ImageNet ResNet	ImageNet E. Net-B7
Baseline	0	97.3	98.5	76.3	84.0
AA	10^{32}	98.5	98.9	77.6	84.4
Fast AA	10^{32}	98.3	98.8	77.6	-
PBA	10^{61}	98.5	98.9	-	-
RA (ours)	10^{2}	98.5	99.0	77.6	85.0

Table 1. RandAugment matches or exceeds predictive performance of other augmentation methods with a significantly reduced search space. We report the search space size and the test accuracy achieved for AutoAugment (AA) [5], Fast AutoAugment [25], Population Based Augmentation (PBA) [20] and the proposed RandAugment (RA) on CIFAR-10 [22], SVHN [34], and ImageNet [6] classification tasks. Architectures presented include PyramidNet [15], Wide-ResNet-28-10 [53], ResNet-50 [17], and EfficientNet-B7 [47]. Search space size is reported as the order of magnitude of the number of possible augmentation policies. All accuracies are the percentage on a cross-validated validation or test split. Dash indicates that results are not available.

1. Introduction

Data augmentation is a widely used method for generating additional data to improve machine learning systems, for image classification [43, 23, 7, 54], object detection [13], instance segmentation [10], and speech recognition [21, 16, 36]. Unfortunately, data augmentation methods require expertise, and manual work to design policies that capture prior knowledge in each domain. This requirement makes it difficult to extend existing data augmentation methods to other applications and domains.

Learning policies for data augmentation has recently emerged as a method to automate the design of augmentation strategies and therefore has the potential to address some weaknesses of traditional data augmentation methods [5, 57, 20, 25]. Training a machine learning model with a learned data augmentation policy may significantly improve accuracy [5], model robustness [32, 52, 41], and performance on semi-supervised learning [50] for image classification; likewise, for object detection tasks on COCO and PASCAL-VOC [57]. Notably, unlike engineering bet-

*Authors contributed equally.
[1] github.com/tensorflow/tpu/tree/master/models/official/efficientnet

1

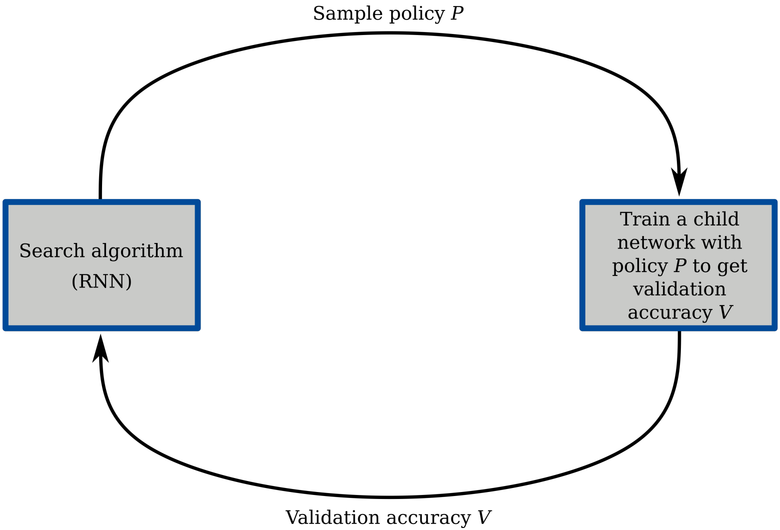
AutoAugment

Problem statement

Goal
Find a good augmentation strategy for a target task

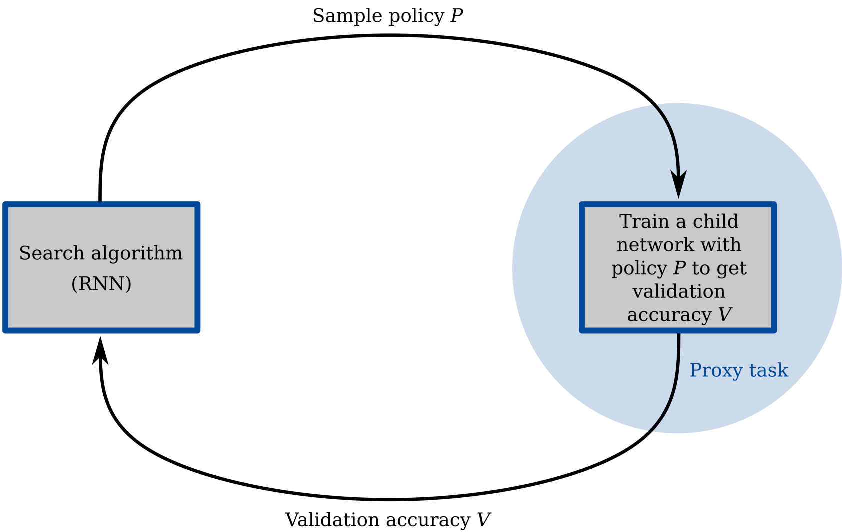
Approach
1. Find an optimized augmentation strategy on a proxy task
2. Apply the strategy on the target task

Find optimal augmentation strategies



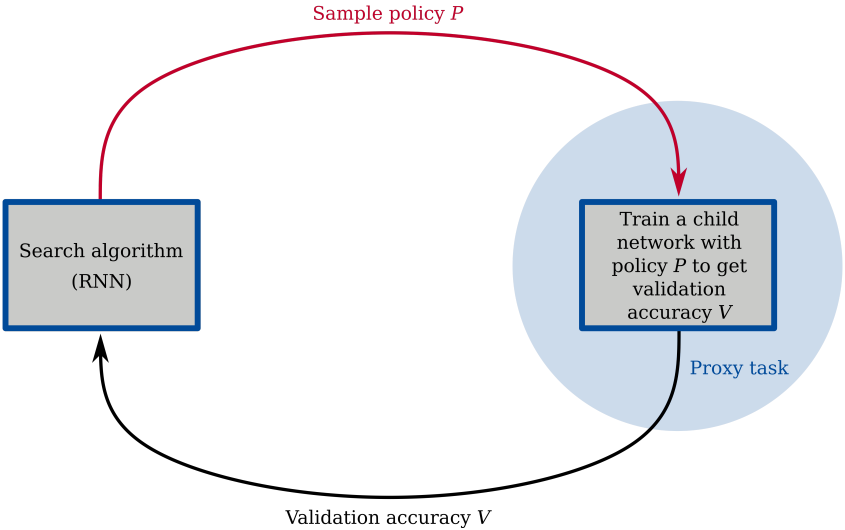
Based on: Cubuk *et al.* [2019]

Find optimal augmentation strategies



Based on: Cubuk *et al.* [2019]

Find optimal augmentation strategies

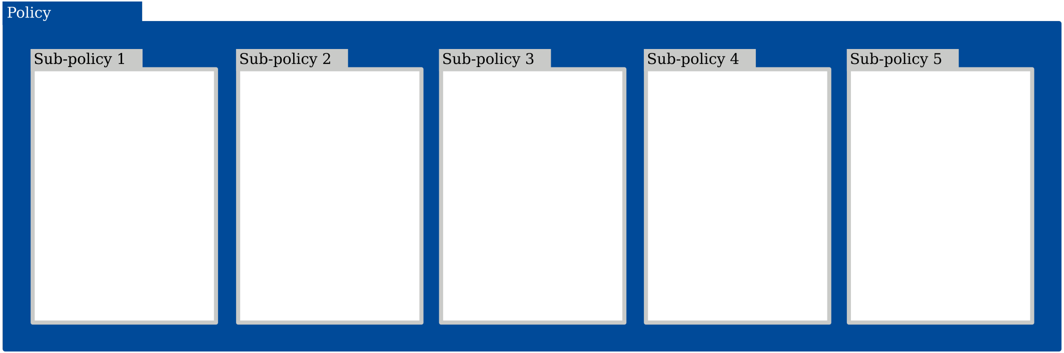


Based on: Cubuk *et al.* [2019]

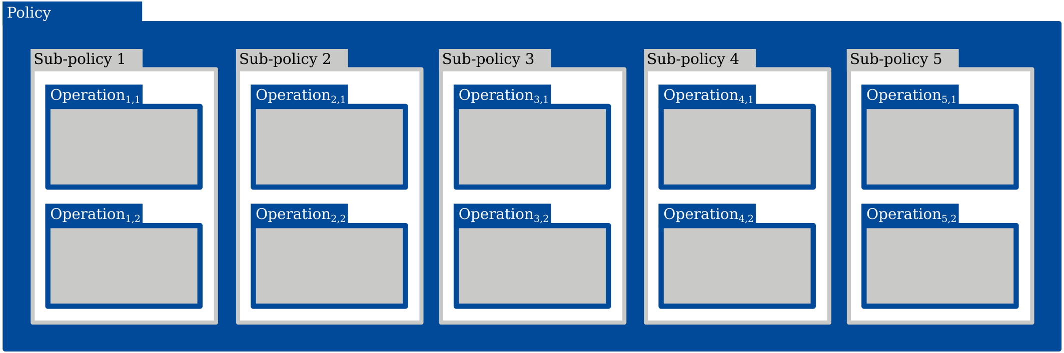
Sample policy P

Policy

Sample policy P



Sample policy P

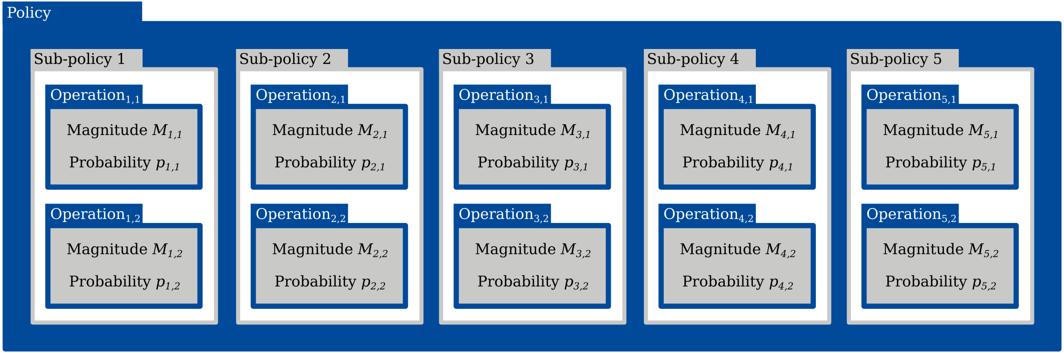


Sample policy P

Policy

- Sub-policy 1
  - Operation$_{1,1}$: Magnitude $M_{1,1}$, Probability $p_{1,1}$
  - Operation$_{1,2}$: Magnitude $M_{1,2}$, Probability $p_{1,2}$
- Sub-policy 2
  - Operation$_{2,1}$: Magnitude $M_{2,1}$, Probability $p_{2,1}$
  - Operation$_{2,2}$: Magnitude $M_{2,2}$, Probability $p_{2,2}$
- Sub-policy 3
  - Operation$_{3,1}$: Magnitude $M_{3,1}$, Probability $p_{3,1}$
  - Operation$_{3,2}$: Magnitude $M_{3,2}$, Probability $p_{3,2}$
- Sub-policy 4
  - Operation$_{4,1}$: Magnitude $M_{4,1}$, Probability $p_{4,1}$
  - Operation$_{4,2}$: Magnitude $M_{4,2}$, Probability $p_{4,2}$
- Sub-policy 5
  - Operation$_{5,1}$: Magnitude $M_{5,1}$, Probability $p_{5,1}$
  - Operation$_{5,2}$: Magnitude $M_{5,2}$, Probability $p_{5,2}$

Sample policy P



$\Rightarrow$ Uniformly sample one sub-policy at random for each image

Sample policy P

Policy

- **Sub-policy 1**
  - Operation$_{1,1}$: Magnitude $M_{1,1}$, Probability $p_{1,1}$
  - Operation$_{1,2}$: Magnitude $M_{1,2}$, Probability $p_{1,2}$
- **Sub-policy 2**
  - Operation$_{2,1}$: Magnitude $M_{2,1}$, Probability $p_{2,1}$
  - Operation$_{2,2}$: Magnitude $M_{2,2}$, Probability $p_{2,2}$
- **Sub-policy 3**
  - Operation$_{3,1}$: Magnitude $M_{3,1}$, Probability $p_{3,1}$
  - Operation$_{3,2}$: Magnitude $M_{3,2}$, Probability $p_{3,2}$
- **Sub-policy 4**
  - Operation$_{4,1}$: Magnitude $M_{4,1}$, Probability $p_{4,1}$
  - Operation$_{4,2}$: Magnitude $M_{4,2}$, Probability $p_{4,2}$
- **Sub-policy 5**
  - Operation$_{5,1}$: Magnitude $M_{5,1}$, Probability $p_{5,1}$
  - Operation$_{5,2}$: Magnitude $M_{5,2}$, Probability $p_{5,2}$

$\Rightarrow$ Uniformly sample one sub-policy at random for each image

$\Rightarrow$ Concatenate the best five policies and train on the full model

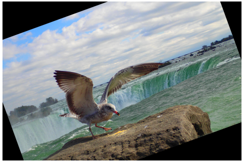
Example of a policy P

Policy P	Sub-Policy	Operation 1	Operation 2
P_1	Sub-policy 1	(Invert, 0.1, 7)	(Contrast, 0.2, 6)
	Sub-policy 2	(Rotate, 0.7, 2)	(TranslateX, 0.3, 9)
	Sub-policy 3	(Sharpness, 0.8, 1)	(Sharpness, 0.9, 3)
	Sub-policy 4	(ShearY, 0.5, 8)	(TranslateY, 0.7, 9)
	Sub-policy 5	(AutoContrast, 0.5, 8)	(Equalize, 0.9, 2)

Reported by: Cubuk *et al*. [2019]

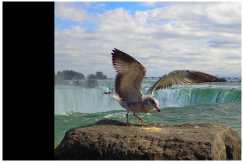
Geometric transformations

Original



Rotate

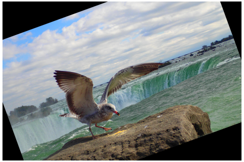
ShearX



TranslateX

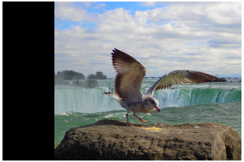
Geometric transformations

Original



Rotate

ShearX



TranslateX

The direction of a geometric transformation is determined randomly

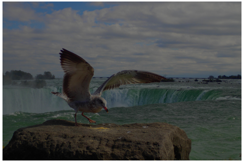
Color transformations

Original

Equalize

Solarize

Posterize



Brightness

Contrast

AutoContrast

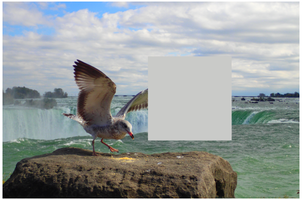
Invert

Color

Sharpness

Other transformations

Original



Cutout

Random cropped and random flipped patches

(Label: Bee)

Label: Gull

Mixing

Label: Gull

Sample Pairing

Results of AutoAugment-direct

Dataset	Architecture	Baseline		AutoAugment	
		Acc.	Search Sp.	Acc.	Search Sp.
Red. CIFAR-10	WR-28-10	83.5	0	**87.7**	10^{32}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}
CIFAR-100	WR-28-10	81.2	0	**82.9**	10^{32}

Reported by: Cubuk *et al.* [2019]

Results of AutoAugment-direct

Dataset	Architecture	Baseline		AutoAugment	
		Acc.	Search Sp.	Acc.	Search Sp.
Red. CIFAR-10	WR-28-10	83.5	0	**87.7**	10^{32}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}
CIFAR-100	WR-28-10	81.2	0	**82.9**	10^{32}

Reported by: Cubuk *et al.* [2019]

$\Rightarrow$ New state-of-the-art accuracies, but high GPU costs

Results of AutoAugment-transfer

- They improved the baselines on five challenging datasets by using the learned policy from ImageNet
- However using the policy found by AutoAugment-direct for a target dataset still yield the best performance

Further results

Relation between #training steps and #sub-policies

A sub-policy needs to be applied for a certain number of training steps before the model benefits from it.

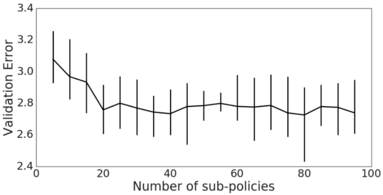
Further results

Relation between #training steps and #sub-policies

A sub-policy needs to be applied for a certain number of training steps before the model benefits from it.

Changing #sub-policies

Increasing the number of sub-policies (up to $\sim$20) improves validation accuracy.



Cubuk *et al.* [2019]

Further results

Randomizing the probabilities and magnitudes

- Improves the baseline from 96.1% to 97.0%
- 0.4% worse than AutoAugment (97.4%)

Further results

Randomizing the probabilities and magnitudes

- Improves the baseline from 96.1% to 97.0%
- 0.4% worse than AutoAugment (97.4%)

Performance of random policies

- Better than the baseline 96.1% to 96.9%
- 0.1% worse than randomizing the probabilities and magnitudes

Problem

A separate search phase on a proxy task:

- Increases training complexity and computational costs
- Only slightly better than random policies

Problem

A separate search phase on a proxy task:

- Increases training complexity and computational costs
- Only slightly better than random policies

Solution

RandAugment

RandAugment

Main contributions of RandAugment

- No proxy task, directly optimize on the target task
- Optimal augmentation strategy depends on the model size and training set size
- Strong reduction of the search space for augmentation strategies

Data augmentation parameter search space

Operation 1 | Operation 2 | $\cdots$ | Operation N

- Sample N transformations uniformly at random (sequentially)
- Use a fixed magnitude M for each augmentation operation

Data augmentation parameter search space

Operation 1 | Operation 2 | $\cdots$ | Operation N

- Sample N transformations uniformly at random (sequentially)
- Use a fixed magnitude M for each augmentation operation

$\Rightarrow$ Optimize the hyperparameters N and M using grid search

Used transformations by RandAugment

Transformations in order to maintain image diversity:

- ShearX/Y
- Equalize
- Brightness
- Color
- ~~Invert~~
- TranslateX/Y
- Solarize
- Contrast
- Sharpness
- ~~Cutout~~
- Rotate
- Posterize
- AutoContrast
- Identity
- ~~Sample Pairing~~

Four strategies for magnitude M

- Random magnitude
- Constant magnitude
- Linearly increasing magnitude
- Random magnitude with increasing upper bound

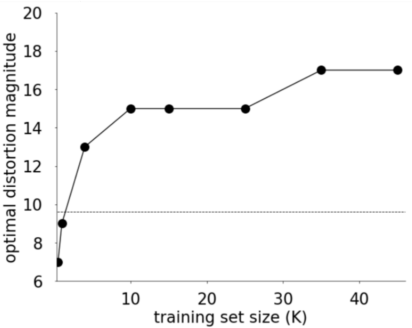
Four strategies for magnitude M

- Random magnitude
- Constant magnitude
- Linearly increasing magnitude
- Random magnitude with increasing upper bound

$\Rightarrow$ Selected constant magnitude through preliminary experiments

Magnitude dependence and results

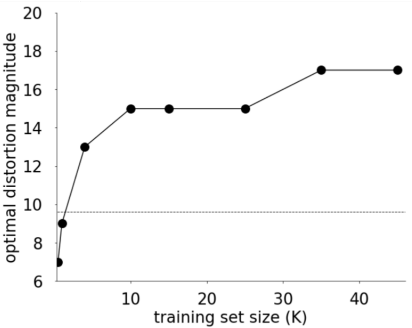
Training set size



Source: Cubuk *et al.* [2019]

Magnitude dependence and results

Training set size



Source: Cubuk *et al.* [2019]

$\Rightarrow$ Larger training set size $\rightarrow$ larger magnitude

Magnitude dependence and results

Training set size

Dataset	Architecture	Baseline		AutoAugment		RandAugment	
		Acc.	Search Sp.	Acc.	Search Sp.	Acc.	Search Sp.
Reduced CIFAR-10	WR-28-10	83.5	0	**87.7**	10^{32}	86.8	10^{2}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}	97.3	10^{2}
SVHN (core set)	WR-28-10	96.9	0	98.1	10^{32}	**98.3**	10^{2}
SVHN	WR-28-10	98.5	0	98.9	10^{32}	**99.0**	10^{2}

Reported by: Cubuk *et al.* [2019]

Magnitude dependence and results

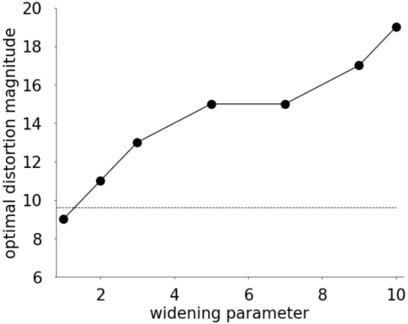
Training set size

Dataset	Architecture	Baseline		AutoAugment		RandAugment	
		Acc.	Search Sp.	Acc.	Search Sp.	Acc.	Search Sp.
Reduced CIFAR-10	WR-28-10	83.5	0	**87.7**	10^{32}	86.8	10^{2}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}	97.3	10^{2}
SVHN (core set)	WR-28-10	96.9	0	98.1	10^{32}	**98.3**	10^{2}
SVHN	WR-28-10	98.5	0	98.9	10^{32}	**99.0**	10^{2}

Reported by: Cubuk *et al.* [2019]

Magnitude dependence and results

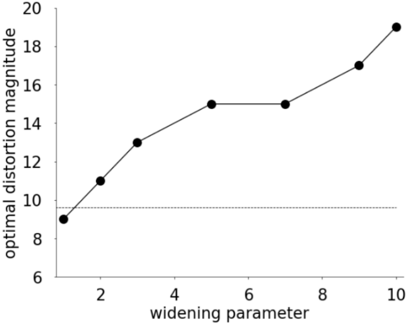
Network size



Source: Cubuk *et al.* [2019]

Magnitude dependence and results

Network size



Source: Cubuk *et al.* [2019]

$\Rightarrow$ Larger network size $\rightarrow$ larger magnitude

Magnitude dependence and results

Network size

Dataset	Architecture	Baseline		AutoAugment		RandAugment	
		Acc.	Search Sp.	Acc.	Search Sp.	Acc.	Search Sp.
CIFAR-10	WR-28-2	94.9	0	**95.9**	10^{32}	95.8	10^{2}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}	97.3	10^{2}
CIFAR-100	WR-28-2	75.4	0	**78.5**	10^{32}	78.3	10^{2}
CIFAR-100	WR-28-10	81.2	0	82.9	10^{32}	**83.3**	10^{2}

Reported by: Cubuk *et al.* [2019]

Magnitude dependence and results

Network size

Dataset	Architecture	Baseline		AutoAugment		RandAugment	
		Acc.	Search Sp.	Acc.	Search Sp.	Acc.	Search Sp.
CIFAR-10	WR-28-2	94.9	0	**95.9**	10^{32}	95.8	10^{2}
CIFAR-10	WR-28-10	96.1	0	**97.4**	10^{32}	97.3	10^{2}
CIFAR-100	WR-28-2	75.4	0	**78.5**	10^{32}	78.3	10^{2}
CIFAR-100	WR-28-10	81.2	0	82.9	10^{32}	**83.3**	10^{2}

Reported by: Cubuk *et al.* [2019]

Dependence on image transformations

Set of transformations	Accuracy
All transformations	85.6 ± 0.3
One transformation removed	85.5 ± 0.3
Only geometric transformations	82.6 ± 0.3

Reported by: Cubuk *et al.* [2019]

Learning probabilities to select transformations

Dataset	RandAugment Acc.	Learned probabilities Acc.
Reduced CIFAR-10	86.8	**87.4**
CIFAR-10	97.3	**97.4**

Reported by: Cubuk *et al.* [2019]

Learning probabilities to select transformations

Dataset	RandAugment Acc.	Learned probabilities Acc.
Reduced CIFAR-10	86.8	**87.4**
CIFAR-10	97.3	**97.4**

Reported by: Cubuk *et al.* [2019]

$\Rightarrow$ Improvement by learning the probabilities

Discussion

Pros and cons

Pros

- AA: Transferability of learned autgmentation policies
- RA: No costs for a proxy task
- Both: Achieved new state-of-the-art accuracies

Pros and cons

Pros

- AA: Transferability of learned autgmentation policies
- RA: No costs for a proxy task
- Both: Achieved new state-of-the-art accuracies

Cons

- AA: Experiments over #sub-policies used a fixed number of epochs
- Errors and contradictions in the papers
- Missing studies

Future works: Ideas from the paper

RandAugment

- Apply RandAugment on other tasks like semantic segmentation, speech recognition, etc.
- More study if and when a separate search phase is required
- Study dependence on image transformations for different datasets

Future works: Further ideas

RandAugment

- Study the magnitude dependence for different $N > 1$ and different datasets

- Transformation importance study
  $\hookrightarrow$ Weight transformations accordingly

Future works: Further ideas

RandAugment

- Study the magnitude dependence for different $N > 1$ and different datasets

- Transformation importance study
  $\hookrightarrow$ Weight transformations accordingly

- Optimize transformation groups separately

- Joint optimization of augmentation strategy and other hyperparameters

Future works: Further ideas

AutoAugment and RandAugment

- Use mixup instead of sample pairing
- Study the number of operation according to the datasets

Conclusion: Automated data augmentation

Presented Works

- Both effectively made use of automated data augmentations
- RandAugment: Successfully solved the problem of AutoAugment

Future Work

Room for further experiments and improvements

Conclusion: Automated data augmentation

Presented Works

- Both effectively made use of automated data augmentations
- RandAugment: Successfully solved the problem of AutoAugment

Future Work

Room for further experiments and improvements

Thank you!

Appendix

AA: More 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	$\mathbf{1.5 \pm 0.1}$
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	$\mathbf{10.0 \pm 0.2}$
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	$\mathbf{10.7 \pm 0.2}$
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**

Source: Cubuk *et al.* [2019]

AA: Results on ImageNet

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.

Source: Cubuk *et al.* [2019]

AA: Transferability of learned policies to other datasets

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

Source: Cubuk *et al.* [2019]

RA: More results

	baseline	PBA	Fast AA	AA	RA
CIFAR-10					
Wide-ResNet-28-2	94.9	-	-	**95.9**	95.8
Wide-ResNet-28-10	96.1	**97.4**	97.3	**97.4**	97.3
Shake-Shake	97.1	**98.0**	**98.0**	**98.0**	**98.0**
PyramidNet	97.3	**98.5**	98.3	**98.5**	**98.5**
CIFAR-100					
Wide-ResNet-28-2	75.4	-	-	**78.5**	78.3
Wide-ResNet-28-10	81.2	**83.3**	82.7	82.9	**83.3**
SVHN (core set)					
Wide-ResNet-28-2	96.7	-	-	98.0	**98.3**
Wide-ResNet-28-10	96.9	-	-	98.1	**98.3**
SVHN					
Wide-ResNet-28-2	98.2	-	-	**98.7**	**98.7**
Wide-ResNet-28-10	98.5	98.9	98.8	98.9	**99.0**

Source: Cubuk *et al.* [2019]

RA: Results on ImageNet

	baseline	Fast AA	AA	RA
ResNet-50	76.3 / 93.1	**77.6** / 93.7	**77.6 / 93.8**	**77.6 / 93.8**
EfficientNet-B5	83.2 / 96.7	-	83.3 / 96.7	**83.9 / 96.8**
EfficientNet-B7	84.0 / 96.9	-	84.4 / 97.1	**85.0 / 97.2**

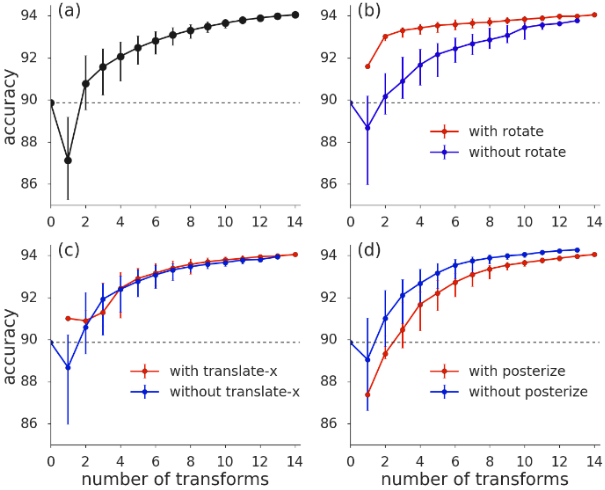
Source: Cubuk *et al.* [2019]

RA: Object detection - COCO dataset

model	augmentation	mAP	search space
ResNet-101	Baseline	38.8	0
	AutoAugment	**40.4**	10^{34}
	RandAugment	40.1	10^{2}
ResNet-200	Baseline	39.9	0
	AutoAugment	**42.1**	10^{34}
	RandAugment	41.9	10^{2}

Source: Cubuk *et al.* [2019]

RA: Transformation importance study 1



Source: Cubuk *et al.* [2019]

RA: Transformation importance study 2

transformation	Δ (%)	transformation	Δ (%)
rotate	**1.3**	shear-x	0.9
shear-y	0.9	translate-y	0.4
translate-x	0.4	autoContrast	0.1
sharpness	0.1	identity	0.1
contrast	0.0	color	0.0
brightness	0.0	equalize	-0.0
solarize	-0.1	posterize	-0.3

Source: Cubuk *et al.* [2019]