This CVPR 2020 workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Randaugment: Practical automated data augmentation with a reduced search space

Ekin D. Cubuk [*], Barret Zoph[*], Jonathon Shlens, Quoc V. Le
Google Research, Brain Team

Abstract

Recent work on automated augmentation strategies has led to state-of-the-art results in image classification and object detection. An obstacle to a large-scale adoption of these methods is that they require a separate and expensive search phase. A common way to overcome the expense of the search phase was to use a smaller proxy task. However, it was not clear if the optimized hyperparameters found on the proxy task are also optimal for the actual task. In this work, we rethink the process of designing automated augmentation strategies. We find that while previous work required a search for both magnitude and probability of each operation independently, it is sufficient to only search for a single distortion magnitude that jointly controls all operations. We hence propose a simplified search space that vastly reduces the computational expense of automated augmentation, and permits the removal of a separate proxy task.

Despite the simplifications, our method achieves equal or better performance over previous automated augmentation strategies on on CIFAR-10/100, SVHN, ImageNet and COCO datasets. EfficientNet-B7, we achieve 85.0% accuracy, a 1.0% increase over baseline augmentation, a 0.6% improvement over AutoAugment on the ImageNet dataset. With EfficientNet-B8, we achieve 85.4% accuracy on ImageNet, which matches a previous result that used 3.5B extra images. On object detection, the same method as classification leads to 1.0-1.3% improvement over baseline augmentation. Code will be made available online.

1. Introduction

Although data augmentation is a widely used method to inject additional knowledge to train vision models [36, 17, 6, 48], the fact that it is manually designed makes it difficult to scale to new applications. Learning data augmentation strategies from data has recently emerged as a new paradigm to automate the design of augmentation and has the potential to address some weaknesses of traditional data augmentation methods [4, 51, 15, 19]. Training a machine learning model with a learned data augmentation policy may significantly improve image classification [4, 19, 15], object detection [51], model robustness [25, 46, 34], and semi-supervised learning image classification [44]. Unlike architecture search [53], all of these improvements in predictive performance incur no additional computational cost at inference time.

		CIFAR-10		**SVHN**		**ImageNet**	
	search	PyramidNet		Wide-ResNet		EfficientNet-B7	
	space	cost	acc.	cost	acc.	cost	acc.
Baseline	0	0	97.3	0	98.5	0	84.0
AA	10^{32}	5K	98.5	1K	98.9	15K	84.4
Fast AA	10^{32}	3.5	98.3	1.5	98.8	-	-
PBA	10^{61}	5.0	98.5	1.0	98.9	-	-
RA (ours)	$\mathbf{10^2}$	**0**	**98.5**	**0**	**99.0**	**0**	**85.0**

Table 1. **RandAugment matches or exceeds predictive performance of other augmentation methods with a significantly reduced search space.** We report the computational cost, the search space size, and the test accuracy achieved for AutoAugment (AA) [4], Fast AutoAugment [19], Population Based Augmentation (PBA) [15] and the proposed RandAugment (RA) on CIFAR-10 [16], SVHN [28], and ImageNet [5] classification tasks. Computational cost is reported as the number of GPU hours expended for identifying the augmentation policy in a separate search phase on a proxy task. Search space size is reported as the order of magnitude of the number of possible augmentation policies. Dash indicates that results are not available.

An obstacle to a large-scale adoption of these methods is that they require a separate and expensive search phase. A common way to overcome the expense of the search phase was to use a smaller proxy task. Although the proxy task helps speeding up the search process, it also adds extra complexity to the methods and causes further issues. For example, it was not clear if the optimal hyperparameters found on the proxy task are also optimal for the actual task. In fact, we will provide experimental evidence in this paper to challenge this core assumption. In particular, we demonstrate that this strategy is sub-optimal as the strength of the augmentation depends strongly on model and dataset size. These results suggest that an improved data augmentation may be possible if one could remove the separate search

[*] Equal contribution.

phase on a proxy task.

In this work, we aim to make AutoAugment and related methods [4, 15, 19] better, and more practical. While previous work focused on the search methodology [19, 15], our analysis shows that the search space plays a more significant role. In previous work, it was required to search for both the probability and the magnitude of each operation in the search space. Our experiments show that it is sufficient to optimize all of the operations jointly with a single distortion magnitude while setting the probability of each operation to uniform. The simplified search space vastly reduces the cost of automated augmentation. With the reduced search space, we also simplify the whole search process: we no longer need a separate expensive search phase and proxy tasks.

The reduction in parameter space is in fact so dramatic that simple grid search is sufficient to find a data augmentation policy that outperforms all learned augmentation methods that employ a separate search phase. We name our method RandAugment because it uniformly samples ops in the search space. Table 1 shows a summary of our main results: despite the fact that RandAugment is much faster thanks to having a much smaller search space. RandAugment also achieves higher accuracy on a wide range of benchmarks, thanks to its ability to adjust its distortion magnitude on the model and dataset size. Our contributions can be summarized as follows:

- We demonstrate that the optimal strength of a data augmentation distortions depends on the model size and training set size. This observation indicates that a separate optimization of an augmentation policy on a smaller proxy task may be sub-optimal for learning and transferring augmentation policies.

- We analyze AutoAugment methods and identify that the search space plays an important role in the results. We hence introduce a vastly simplified search space for data augmentation containing 2 interpretable hyper-parameters. One may employ a simple grid search to tailor the augmentation policy to a model and dataset, removing the need for a separate search process. Our change is however orthogonal to the better search methods and can be used in combination with them.

- Despite the simplifications, our method surprisingly outperforms AutoAugment and related methods. We achieve state-of-the-art results on a wide range of datasets: CIFAR, SVHN, and ImageNet. With EfficientNet-B7, we achieve an accuracy of 85.0%, a 0.6% increment over AutoAugment and 1.0% over baseline augmentation [39]. With an even larger network, EfficientNet-B8, we achieve 85.4% accuracy on ImageNet with no extra data, which matches a previous result that used an additional 3.5B Instagram images [26] (with a model that has 9.4 times more parameters than ours).

2. Systematic failures of a separate proxy task

A central premise of learned data augmentation is to construct a small, proxy task that may be reflective of a larger task [52, 53, 4]. Although this assumption is sufficient for identifying learned augmentation policies to improve performance [4, 51, 30, 19, 15], it is unclear if this assumption is overly stringent and may lead to sub-optimal data augmentation policies.

In this section, we challenge the hypothesis that formulating the problem in terms of a small proxy task is appropriate for learned data augmentation. In particular, we explore this question along two separate dimensions that are commonly restricted to achieve a small proxy task: model size and dataset size. To explore this hypothesis, we systematically measure the effects of data augmentation policies on CIFAR-10. First, we train a family of Wide-ResNet architectures [47], where the model size may be systematically altered through the *widening* parameter governing the number of convolutional filters. For each of these networks, we train the model on CIFAR-10 and measure the final accuracy compared to a baseline model trained with default data augmentations (i.e. horizontal flips and pad-and-crop) [47]. The Wide-ResNet models are trained with the additional K=14 data augmentations (see Section 3) over a range of global distortion magnitudes M parameterized on a uniform linear scale ranging from [0, 30] [1].

Figure 1a demonstrates the relative gain in accuracy of a model trained across increasing distortion magnitudes for three Wide-ResNet models. The squares indicate the distortion magnitude with which achieves the highest accuracy. Note that in spite of the measurement noise, Figure 1a demonstrates systematic trends across distortion magnitudes. In particular, plotting all Wide-ResNet architectures versus the optimal distortion magnitude highlights a clear monotonic trend across increasing network sizes (Figure 1b). Namely, larger networks demand larger data distortions for regularization. Figure 2 highlights the visual difference in the optimal distortion magnitude for differently sized models. Conversely, a policy learned on a proxy task (such as AutoAugment) provides a fixed distortion magnitude (Figure 1b, dashed line) for all architectures that is clearly sub-optimal.

A second dimension for constructing a small proxy task is to train the proxy on a small subset of the training data. Figure 1c demonstrates the relative gain in accu-

[1] Note that the range of magnitudes exceeds the specified range of magnitudes in the Methods because we wish to explore a larger range of magnitudes for this preliminary experiment. We retain the same scale as [4] for a value of 10 to maintain comparable results.

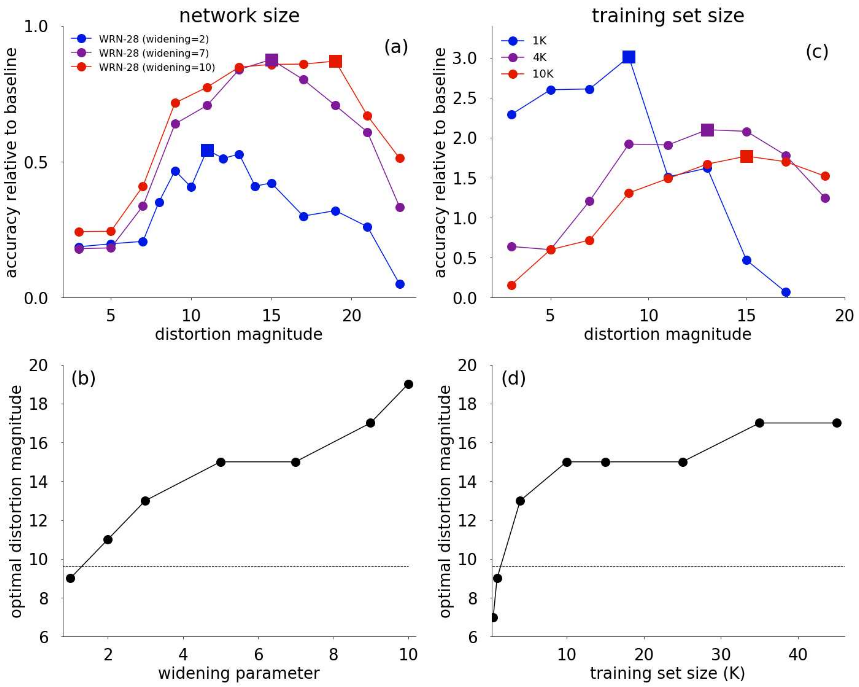

Figure 1. **Optimal magnitude of augmentation depends on the size of the model and the training set.** All results report CIFAR-10 validation accuracy for Wide-ResNet model architectures [47] averaged over 20 random initializations, where $N = 1$. (a) Accuracy of Wide-ResNet-28-2, Wide-ResNet-28-7, and Wide-ResNet-28-10 across varying distortion magnitudes. Models are trained for 200 epochs on 45K training set examples. Squares indicate the distortion magnitude that achieves the maximal accuracy. (b) Optimal distortion magnitude across 7 Wide-ResNet-28 architectures with varying widening parameters (k). (c) Accuracy of Wide-ResNet-28-10 for three training set sizes (1K, 4K, and 10K) across varying distortion magnitudes. Squares indicate the distortion magnitude that achieves the maximal accuracy. (d) Optimal distortion magnitude across 8 training set sizes. Dashed curves show the scaled expectation value of the distortion magnitude in the AutoAugment policy. [4]

racy of Wide-ResNet-28-10 trained across increasing distortion magnitudes for varying amounts of CIFAR-10 training data. The squares indicate the distortion magnitude with that achieves the highest accuracy. Note that in spite of the measurement noise, Figure 1c demonstrates systematic trends across distortion magnitudes. We first observe that models trained on smaller training sets may gain more improvement from data augmentation (e.g. 3.0% versus 1.5% in Figure 1c). Furthermore, we see that the optimal distortion magnitude is larger for models that are trained on larger datasets. At first glance, this may disagree with the expectation that smaller datasets require stronger regularization.

Figure 1d demonstrates that the optimal distortion magnitude increases monotonically with training set size. One hypothesis for this counter-intuitive behavior is that aggressive data augmentation leads to a low signal-to-noise ratio in small datasets. Regardless, this trend highlights the need for increasing the strength of data augmentation on larger datasets and the shortcomings of optimizing learned augmentation policies on a proxy task comprised of a subset of the training data. Namely, the learned augmentation may learn an augmentation strength more tailored to the proxy task instead of the larger task of interest.

The dependence of augmentation strength on the dataset and model size indicate that a small proxy task may provide a sub-optimal indicator of performance on a larger task. This empirical result suggests that a distinct strategy may be necessary for finding an optimal data augmentation policy. In particular, we propose in this work to focus on a *unified* optimization of the model weights and data augmentation policy. Figure 1 suggest that merely searching for a shared distortion magnitude M across all transformations may provide sufficient gains that exceed learned optimization methods using proxy tasks. Additionally, we see that optimizing individual magnitudes further leads to minor improvement in performance (see Section A.1.2 in Appendix).

Furthermore, Figure 1a and 1c indicate that merely sampling a few distortion magnitudes is sufficient to achieve good results. Coupled with a second free parameter N, we consider these results to prescribe an algorithm for learning an augmentation policy. In the subsequent sections, we identify two free parameters N and M specifying RandAugment through a minimal grid search and compare these results against computationally-heavy learned data augmentations based on proxy tasks.

3. Automated data augmentation without a proxy task

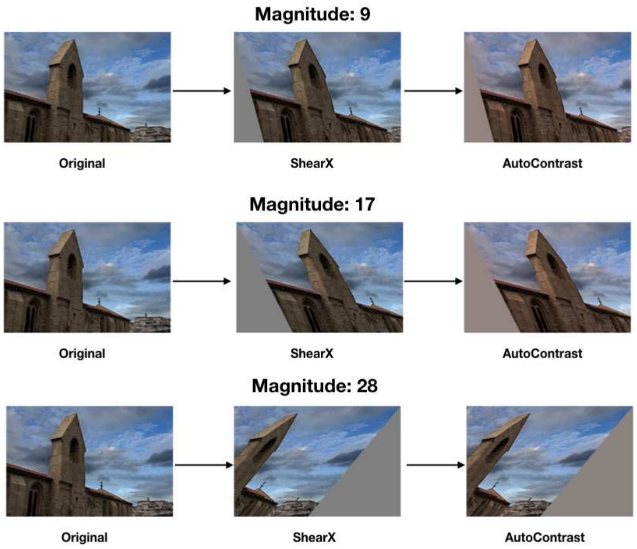

Figure 2. **Example images augmented by RandAugment.** In these examples N=2 and three magnitudes are shown corresponding to the optimal distortion magnitudes for ResNet-50, EfficientNet-B5 and EfficientNet-B7, respectively. As the distortion magnitude increases, the strength of the augmentation increases.

The primary goal of RandAugment is to remove the need for a separate search phase on a proxy task. The reason we wish to remove the search phase is because a separate search phase significantly complicates training and is computationally expensive. More importantly, the proxy task may provide sub-optimal results, as seen in the previous section. In order to remove a separate search phase, we aspire to fold the parameters for the data augmentation strategy into the hyper-parameters for training a model. Given that previous learned augmentation methods contained 30+ parameters [4, 19, 15], we focus on vastly reducing the parameter space for data augmentation.

Previous work indicates that the main benefit of learned augmentation policies arise from increasing the diversity of examples [4, 15, 19]. Indeed, previous work enumerated a policy in terms of choosing which transformations to apply out of K=14 available transformations, and probabilities for applying each transformation:

- `identity`
- `autoContrast`
- `equalize`
- `rotate`
- `solarize`
- `color`
- `posterize`
- `contrast`
- `brightness`
- `sharpness`
- `shear-x`
- `shear-y`
- `translate-x`
- `translate-y`

In order to reduce the parameter space but still maintain image diversity, we replace the learned policies and probabilities for applying each transformation with a parameter-free procedure of *always* selecting a transformation with uniform probability $\frac{1}{K}$. Given N transformations for a training image, RandAugment may thus express K^N potential policies.

```
transforms = [
'Identity', 'AutoContrast', 'Equalize', 'Rotate',
    'Solarize', 'Color',
'Posterize', 'Contrast', 'Brightness', 'Sharpness',
    'ShearX', 'ShearY',
'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to apply
        sequentially.
    M: Magnitude for all the transformations.
"""

  sampled_ops = np.random.choice(transforms, N)
  return [(op, M) for op in sampled_ops]
```

Figure 3. Python code for RandAugment based on numpy.

The final set of parameters to consider is the magnitude of the each augmentation distortion. Following [4], we employ the same linear scale for indicating the strength of each transformation. Briefly, each transformation resides on an integer scale from 0 to 10 where a value of 10 indicates the maximum scale for a given transformation. A data augmentation policy consists of identifying an integer for each augmentation [4, 19, 15]. In order to reduce the parameter space further, we observe that the learned magnitude for each transformation follows a similar schedule during training (e.g. Figure 4 in [15]) and postulate that a *single* global distortion M may suffice for parameterizing all transformations. We experimented with four methods for the schedule of M during training: constant magnitude, random magnitude, a linearly increasing magnitude, and a random magnitude with increasing upper bound. The details of this experiment can be found in Appendix A.1.1.

The resulting algorithm contains two parameters N and M and may be expressed simply in two lines of Python code (Figure 3). Both parameters are human-interpretable such that larger values of N and M increase regularization strength. Standard methods may be employed to efficiently perform hyperparameter optimization [38], however given the extremely small search space we find that naive grid search is quite effective (Section 2). We justify all of the choices of this proposed algorithm in the subsequent sections by comparing the efficacy of our simple method to previous learned data augmentation methods.

	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**

Table 2. **Test accuracy (%) on CIFAR-10, CIFAR-100, SVHN and SVHN core set**. Comparisons across default data augmentation (baseline), Population Based Augmentation (PBA) [15] and Fast AutoAugment (Fast AA) [19], AutoAugment (AA) [4] and proposed RandAugment (RA). Note that baseline and AA are replicated in this work. SVHN core set consists of 73K examples. The Shake-Shake model [10] employed a 26 2×96d configuration, and the PyramidNet model used the ShakeDrop regularization [45]. Results reported by us are averaged over 10 independent runs. Bold indicates best results.

4. Experiments

To explore the space of data augmentations, we experiment with core image classification and object detection tasks. In particular, we focus on CIFAR-10, CIFAR-100, SVHN, and ImageNet datasets as well as COCO object detection so that we may compare with previous work. For all of these datasets, we replicate the corresponding architectures and set of data transformations. Our goal is to demonstrate the relative benefits of employing this method over previous learned augmentation methods.

4.1. CIFAR-10 and SVHN

CIFAR-10 has been extensively studied with previous data augmentation methods and we first test this proposed method on this data. The default augmentations for all methods include flips, pad-and-crop and Cutout [7]. N and M were selected based on the validation performance on 5K held out examples from the training set for 1 and 5 settings for N and M, respectively. Results indicate that RandAugment achieves either competitive (i.e. within 0.1%) or state-of-the-art on CIFAR-10 across four network architectures (Table 2). As a more challenging task, we additionally compare the efficacy of RandAugment on CIFAR-100 for Wide-ResNet-28-2 and Wide-ResNet-28-10. On the held out 5K dataset, we sampled 2 and 4 settings for N and M, respectively (i.e. N=$\{1, 2\}$ and M=$\{2, 6, 10, 14\}$). For Wide-ResNet-28-2 and Wide-ResNet-28-10, we find that N=1, M=2 and N=2, M=14 achieves best results, respectively. Again, RandAugment achieves competitive or superior results across both architectures (Table 2).

Because SVHN is composed of numbers instead of natural images, the data augmentation strategy for SVHN may differ substantially from CIFAR-10. Indeed, [4] identified a qualitatively different policy for CIFAR-10 than SVHN. Likewise, in a semi-supervised setting for CIFAR-10, a policy learned from CIFAR-10 performs better than a policy learned from SVHN [44].

SVHN has a core training set of 73K images [28]. In addition, SVHN contains 531K less difficult "extra" images to augment training. We compare the performance of the augmentation methods on SVHN with and without the extra data on Wide-ResNet-28-2 and Wide-ResNet-28-10 (Table 2). In spite of the large differences between SVHN and CIFAR, RandAugment consistently matches or outperforms previous methods with no alteration to the list of transformations employed. Notably, for Wide-ResNet-28-2, applying RandAugment to the core training dataset improves performance more than augmenting with 531K additional training images (98.3% vs. 98.2%). For, Wide-ResNet-28-10, RandAugment is competitive with augmenting the core training set with 531K training images (i.e. within 0.2%). Nonetheless, Wide-ResNet-28-10 with RandAugment matches the previous state-of-the-art accuracy on SVHN which used a more advanced model [4].

4.2. Image classification with ImageNet dataset

Data augmentation methods that improve CIFAR-10 and SVHN models do not always improve large-scale tasks such as ImageNet. For instance, Cutout substantially improves CIFAR and SVHN performance [7], but fails to improve ImageNet [25]. Likewise, AutoAugment does not increase the performance on ImageNet as much as other tasks [4], especially for large networks (e.g. +0.4% for AmoebaNet-C [4] and +0.1% for EfficientNet-B5 [41]). One plausible reason for the lack of strong gains is that the small proxy task was particularly impoverished by restricting the task to $\sim$10% of the 1000 ImageNet classes.

Table 3 compares the performance of RandAugment to other learned augmentation approaches on ImageNet. RandAugment matches the performance of AutoAugment and Fast AutoAugment on the smallest model (ResNet-50), but on larger models RandAugment significantly outperforms other methods achieving increases of up to +1.3% above the baseline. For instance, on EfficientNet-B7, the resulting model achieves 85.0% – a new state-of-the-art accuracy – exhibiting a 1.0% improvement over the baseline augmentation. These systematic gains are similar to the improvements achieved with engineering new architectures [53, 22], however these gains arise without incurring additional computational cost at inference time.

4.3. Object detection with COCO dataset

To further test the generality of this approach, we next explore a related task of large-scale object detection on the

	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**

Table 3. **ImageNet results.** Top-1 and Top-5 accuracies (%) on ImageNet. Baseline and AutoAugment (AA) results on ResNet-50 are from [4]. Fast AutoAugment (Fast AA) results are from [19]. EfficientNet results with and without AutoAugment are from [41]. Highest accuracy for each model is presented in bold. Note that Population Based Augmentation (PBA) [15] has not been implemented on ImageNet.

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

Table 4. **Results on object detection.** Mean average precision (mAP) on COCO detection task. Search space size is reported as the order of magnitude of the number of possible augmentation policies. Models are trained for 300 epochs from random initialization following [51].

COCO dataset [21]. Learned augmentation policies have improved object detection and lead to state-of-the-art results [51]. We followed previous work by training on the same architectures and following the same training schedules (see Appendix A.3). Briefly, we employed RetinaNet [20] with ResNet-101 and ResNet-200 as a backbone [12]. Models were trained for 300 epochs from random initialization.

Table 4 compares results between a baseline model, AutoAugment and RandAugment. AutoAugment leveraged additional, specialized transformations not afforded to RandAugment in order to augment the localized bounding box of an image [51]. In addition, note that AutoAugment expended ~15K GPU hours for search, where as RandAugment was tuned by on merely 6 values of the hyper-parameters (see Appendix A.3). In spite of the smaller library of specialized transformations and the lack of a separate search phase, RandAugment surpasses the baseline model and provides competitive accuracy with AutoAugment. We reserve for future work to expand the transformation library to include bounding box specific transformation to potentially improve RandAugment results even further.

4.4. Investigating the dependence on the included transformations

RandAugment achieves state-of-the-art results across different tasks and datasets using the same list of transformations. This result suggests that RandAugment is largely insensitive to the selection of transformations for different datasets. To further study the sensitivity, we experimented with RandAugment on a Wide-ResNet-28-2 trained on CIFAR-10 for randomly sampled subsets of the full list of 14 transformations. We did not use flips, pad-and-crop, or cutout to only focus on the improvements due to RandAugment with random subsets. Figure 4a suggests that the median validation accuracy due to RandAugment improves as the number of transformations is increased. However, even with only two transformations, RandAugment leads to more than 1% improvement in validation accuracy on average.

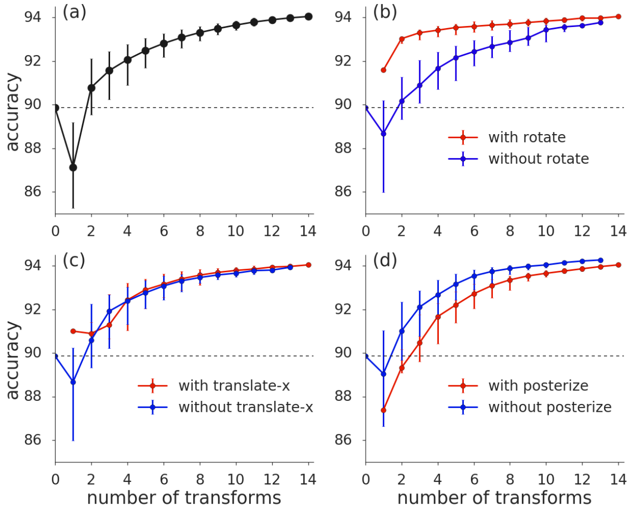

Figure 4. **Average performance improves when more transformations are included in RandAugment.** All panels report median CIFAR-10 validation accuracy for Wide-ResNet-28-2 model architectures [47] trained with RandAugment ($N = 3$, $M = 4$) using randomly sampled subsets of transformations. No other data augmentation is included in training. Error bars indicate 30th and 70th percentile. (a) Median accuracy for randomly sampled subsets of transformations. (b) Median accuracy for subsets with and without the `Rotate` transformation. (c) Median accuracy for subsets with and without the `translate-x` transformation. (d) Median accuracy for subsets with and without the `posterize` transformation. Dashed curves show the accuracy of the model trained without any augmentations.

To get a sense for the effect of individual transformations, we calculate the average improvement in validation accuracy for each transformation when they are added to a random subset of transformations. We list the transformations in order of most helpful to least helpful in Table 5. We see that while geometric transformations individually make the most difference, some of the color transformations lead to a degradation of validation accuracy on average. Note

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

Table 5. **Average improvement due to each transformation.** Average difference in validation accuracy (%) when a particular transformation is added to a randomly sampled set of transformations. For this ablation study, Wide-ResNet-28-2 models were trained on CIFAR-10 using RandAugment ($N = 3$, $M = 4$) with the randomly sampled set of transformations, with no other data augmentation.

that while Table 5 shows the average effect of adding individual transformations to randomly sampled subsets of transformations, Figure 4a shows that including all transformations together leads to a good result. The transformation `rotate` is most helpful on average, which was also observed previously [4, 51]. To see the effect of representative transformations in more detail, we repeat the analysis in Figure 4a for subsets with and without (`rotate, translate-x, and posterize`). Surprisingly, `rotate` can significantly improve performance and lower variation even when included in small subsets of RandAugment transformations, while `posterize` seems to hurt all subsets of all sizes.

4.5. Learning the probabilities for selecting image transformations

	baseline	AA	RA	+ 1st
Reduced CIFAR-10				
Wide-ResNet-28-2	82.0	**85.6**	85.3	85.5
Wide-ResNet-28-10	83.5	**87.7**	86.8	87.4
CIFAR-10				
Wide-ResNet-28-2	94.9	95.9	95.8	**96.1**
Wide-ResNet-28-10	96.1	**97.4**	97.3	**97.4**

Table 6. **Differentiable optimization for augmentation can improve RandAugment.** Test accuracy (%) from differentiable RandAugment for reduced (4K examples) and full CIFAR-10. The 1st-order approximation (1st) is based on density matching (Section 4.5). Models trained on reduced CIFAR-10 were trained for 500 epochs. CIFAR-10 models trained using the same hyperparameters as previous. Each result is averaged over 10 independent runs.

RandAugment selects all image transformations with equal probability. This opens up the question of whether learning K probabilities may improve performance further. Most of the image transformations (except posterize, equalize, and autoContrast) are differentiable, which permits backpropagation to learn the K probabilities [23]. Let us denote α_{ij} as the learned probability of selecting image transformation i for operation j. For K=14 image transformations and N=2 operations, α_{ij} constitutes 28 parameters. We initialize all weights such that each transformation is equal probability (i.e. RandAugment), and update these parameters based on how well a model classifies a held out set of validation images distorted by α_{ij}. This approach was inspired by density matching [19], but instead uses a differentiable approach in lieu of Bayesian optimization. We label this method as a 1st-order density matching approximation.

To test the efficacy of density matching to learn the probabilities of each transformation, we trained Wide-ResNet-28-2 and Wide-ResNet-28-10 on CIFAR-10 and the reduced form of CIFAR-10 containing 4K training samples. Table 6 indicates that learning the probabilities α_{ij} slightly improves performance on reduced and full CIFAR-10 (RA vs 1st). The 1st-order method improves accuracy by more than 3.0% for both models on reduced CIFAR-10 compared to the baseline of flips and pad-and-crop. On CIFAR-10, the 1st-order method improves accuracy by 0.9% on the smaller model and 1.2% on the larger model compared to the baseline. We further see that the 1st-order method always performs better than RandAugment, with the largest improvement on Wide-ResNet-28-10 trained on reduced CIFAR-10 (87.4% vs. 86.8%). On CIFAR-10, the 1st-order method outperforms AutoAugment on Wide-ResNet-28-2 (96.1% vs. 95.9%) and matches AutoAugment on Wide-ResNet-28-10 [2]. Although the density matching approach is promising, this method can be expensive as one must apply all K transformations N times to each image independently. Hence, because the computational demand of KN transformations is prohibitive for large images, we reserve this for future exploration. In summary, we take these results to indicate that learning the probabilities through density matching may improve the performance on small-scale tasks and reserve explorations to larger-scale tasks for the future.

5. Related Work

Data augmentation has played a central role in the training of deep vision models. On natural images, horizontal flips and random cropping or translations of the images are commonly used in classification and detection models [47, 17, 11]. On MNIST, elastic distortions across scale, position, and orientation have been applied to achieve impressive results [36, 3, 43, 35]. While previous examples augment the data while keeping it in the training set distribution, operations that do the opposite can also be effective in increasing generalization. Some methods randomly erase or add noise to patches of images for increased vali-

[2] As a baseline comparison, in preliminary experiments we additionally learn α_{ij} based on differentiating through a virtual training step [23]. In this approach, the 2nd-order approximation yielded consistently negative results (see Appendix A.1).

dation accuracy [7, 49], robustness [40, 46, 9], or both [25]. Mixup [48] is a particularly effective augmentation method on CIFAR-10 and ImageNet, where the neural network is trained on convex combinations of images and their corresponding labels. Object-centric cropping is commonly used for object detection tasks [24], whereas [8] adds new objects on training images by cut-and-paste.

Moving away from individual operations to augment data, other work has focused on finding optimal strategies for combining different operations. For example, Smart Augmentation learns a network that merges two or more samples from the same class to generate new data [18]. Tran et al. generate augmented data via a Bayesian approach, based on the distribution learned from the training set [42]. DeVries et al. use transformations (e.g. noise, interpolations and extrapolations) in the learned feature space to augment data [6]. Furthermore, generative adversarial networks (GAN) have been used to choose optimal sequences of data augmentation operations[32]. GANs have also been used to generate training data directly [31, 27, 50, 1, 37], however this approach does not seem to be as beneficial as learning sequences of data augmentation operations that are pre-defined [33].

Another approach to learning data augmentation strategies from data is AutoAugment [4], which originally used reinforcement learning to choose a sequence of operations as well as their probability of application and magnitude. Application of AutoAugment policies involves stochasticity at multiple levels: 1) for every image in every minibatch, a sub-policy is chosen with uniform probability. 2) operations in each sub-policy has an associated probability of application. 3) Some operations have stochasticity over direction. For example, an image can be rotated clockwise or counter-clockwise. The layers of stochasticity increase the amount of diversity that the network is trained on, which in turn was found to significantly improve generalization on many datasets. More recently, several papers used the AutoAugment search space and formalism with improved optimization algorithms to find AutoAugment policies more efficiently [15, 19]. Although the time it takes to search for policies has been reduced significantly, having to implement these methods in a separate search phase reduces the applicability of AutoAugment. For this reason, this work aims to eliminate the search phase on a separate proxy task completely.

Some of the developments in RandAugment were inspired by the recent improvements to searching over data augmentation policies. For example, PBA [15] found that the optimal magnitude of augmentations increased during the course of training, which inspired us to not search over optimal magnitudes for each transformation but have a fixed magnitude schedule, which we discuss in detail in Section 3. Furthermore, authors of Fast AutoAugment [19] found that a data augmentation policy that is trained for density matching leads to improved generalization accuracy, which inspired our first order differentiable term for improving augmentation (see Section 4.5).

6. Discussion

Data augmentation is a necessary method for achieving state-of-the-art performance [36, 17, 6, 48, 11, 30]. Learned data augmentation strategies have helped automate the design of such strategies and likewise achieved state-of-the-art results [4, 19, 15, 51]. In this work, we demonstrated that previous methods of learned augmentation suffers from systematic drawbacks. Namely, not tailoring the number of distortions and the distortion magnitude to the dataset size nor the model size leads to sub-optimal performance. To remedy this situation, we propose a simple parameterization for targeting augmentation to particular model and dataset sizes. We demonstrate that RandAugment is competitive with or outperforms previous approaches [4, 19, 15, 51] on CIFAR-10/100, SVHN, ImageNet and COCO without a separate search for data augmentation policies.

In previous work, scaling learned data augmentation to larger dataset and models have been a notable obstacle. For example, AutoAugment and Fast AutoAugment could only be optimized for small models on reduced subsets of data [4, 19]; PBA was not reported for large-scale problems [15]. The proposed method scales quite well to datasets such as ImageNet and COCO while incurring minimal computational cost (e.g. 2 hyper-parameters), but notable predictive performance gains. An open question remains how this method may improve model robustness [25, 46, 34] or semi-supervised learning [44]. Future work will study how this method applies to other machine learning domains, where data augmentation is known to improve predictive performance, such as image segmentation [2], 3-D perception [29], speech recognition [14] or audio recognition [13]. In particular, we wish to better understand if or when datasets or tasks may require a separate search phase to achieve optimal performance. Finally, an open question remains how one may tailor the set of transformations to a given tasks in order to further improve the predictive performance of a given model.

7. Acknowledgements

We thank Samy Bengio, Daniel Ho, Ildoo Kim, Jaehoon Lee, Zhaoqi Leng, Hanxiao Liu, Raphael Gontijo Lopes, Ruoming Pang, Ben Poole, Mingxing Tan, and the rest of the Brain Team for their help.

References

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[3] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649. IEEE, 2012.

[4] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[6] Terrance DeVries and Graham W Taylor. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*, 2017.

[7] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[8] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017.

[9] Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. Adversarial examples are a natural consequence of test error in noise. *arXiv preprint arXiv:1901.10513*, 2019.

[10] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.

[11] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár, and Kaiming He. Detectron, 2018.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[13] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[14] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdelrahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Brian Kingsbury, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

[15] Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, and Xi Chen. Population based augmentation: Efficient learning of augmentation policy schedules. *arXiv preprint arXiv:1905.05393*, 2019.

[16] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[18] Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation learning an optimal data augmentation strategy. *IEEE Access*, 5:5858–5869, 2017.

[19] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[22] Chenxi Liu, Barret Zoph, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. *arXiv preprint arXiv:1712.00559*, 2017.

[23] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[25] Raphael Gontijo Lopes, Dong Yin, Ben Poole, Justin Gilmer, and Ekin D Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. *arXiv preprint arXiv:1906.02611*, 2019.

[26] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. *arXiv preprint arXiv:1805.00932*, 2018.

[27] Seongkyu Mun, Sangwook Park, David K Han, and Hanseok Ko. Generative adversarial network based acoustic scene training set augmentation and selection using svm hyperplane. In *Detection and Classification of Acoustic Scenes and Events Workshop*, 2017.

[28] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[29] Jiquan Ngiam, Benjamin Caine, Wei Han, Brandon Yang, Yuning Chai, Pei Sun, Yin Zhou, Xi Yi, Ouais Alsharif, Patrick Nguyen, et al. Starnet: Targeted computation for object detection in point clouds. *arXiv preprint arXiv:1908.11069*, 2019.

[30] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[31] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[32] Alexander J Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Learning to compose domain-specific transformations for data augmentation. In *Advances in Neural Information Processing Systems*, pages 3239–3249, 2017.

[33] Suman Ravuri and Oriol Vinyals. Classification accuracy score for conditional generative models. *arXiv preprint arXiv:1905.10887*, 2019.

[34] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? *arXiv preprint arXiv:1902.10811*, 2019.

[35] Ikuro Sato, Hiroki Nishimura, and Kensuke Yokoi. Apac: Augmented pattern classification with neural networks. *arXiv preprint arXiv:1505.03229*, 2015.

[36] Patrice Y Simard, David Steinkraus, John C Platt, et al. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of International Conference on Document Analysis and Recognition*, 2003.

[37] Leon Sixt, Benjamin Wild, and Tim Landgraf. Rendergan: Generating realistic labeled data. *arXiv preprint arXiv:1611.01331*, 2016.

[38] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016.

[40] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[41] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019.

[42] Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. In *Advances in Neural Information Processing Systems*, pages 2794–2803, 2017.

[43] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, pages 1058–1066, 2013.

[44] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*, 2019.

[45] Yoshihiro Yamada, Masakazu Iwamura, and Koichi Kise. Shakedrop regularization. *arXiv preprint arXiv:1802.02375*, 2018.

[46] Dong Yin, Raphael Gontijo Lopes, Jonathon Shlens, Ekin D Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *arXiv preprint arXiv:1906.08988*, 2019.

[47] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

[48] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[49] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.

[50] Xinyue Zhu, Yifan Liu, Zengchang Qin, and Jiahong Li. Data augmentation in emotion classification using generative adversarial networks. *arXiv preprint arXiv:1711.00648*, 2017.

[51] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. *arXiv preprint arXiv:1906.11172*, 2019.

[52] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017.

[53] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.