Appendix:
TeachAugment: Data Augmentation Optimization Using Teacher Knowledge

A. Training setups

A.1. Image classification

We summarize the hyperparamters for training in Tab. 1. Each model was trained with Nesterov's accelerated gradient method [12] in the stochastic gradient descent. The cross entropy loss between the model prediction and the ground truth label was used as the loss function. We gradually warmed up the learning rate for five epochs until it reached the predefined learning rate shown in Tab. 1. As baseline augmentation, we used random horizontal flipping and random cropping with a crop size of 32 for CIFAR-10 and CIFAR-100 and 224 for ImageNet. In addition, we also used Cutout with a crop size of 16 for CIFAR-10 and CIFAR-100, and random color distortion for ImageNet.[1]

A.2. Semantic segmentation

We trained all models using the stochastic gradient descent with momentum of 0.9 for 300 epochs. The cross entropy loss between the model prediction and the ground truth label was used as the loss function. We set the initial learning rate to 5e-3 and decayed it with poly learning rate decay where the initial learning rate was multiplied by $\left(1 - \frac{iter}{max\_iter}\right)^{0.9}$. The coefficient of the auxiliary loss used in [17] was set to 0.4. As baseline augmentation, we used random horizontal flipping and random cropping with a crop size of 1024. For TeachAugment, the smoothing parameter ϵ in label smoothing was set to 0.1. For RandAugment, the number of transformations n and the magnitude m, were set to 1 and 5, which were tuned for FCN-32s using grid search.

A.3. Unsupervised representation learning

We evaluated each method following the linear evaluation setting [2].[2] We modified the objective of our method for SimSiam as follows:

$$\max_{\phi_1,\phi_2} \min_{\theta} \mathbb{E}_{x\sim\mathcal{X}}[L(f_\theta(a_{\phi_1}(x)), f_\theta(a_{\phi_2}(x))) - L(f_\theta(a_{\phi_1}(x)), f_\theta(a_{\phi_2}(x)))], \quad (1)$$

where L denotes the cosine distance. Because the non-saturating loss and label smoothing cannot be applied to the cosine distance, we omitted them in this experiment.

Note that our method with the EMA teacher cannot be simply applied to other methods, such as BYOL [4] and MoCo [1, 6], because they already integrate the EMA teacher into their training frameworks. It will be future work to investigate combinations of such methods.

As pretraining, we trained ResNet-50 for 100, 200, and 400 epochs with a batch size of 256. The momentum SGD was employed as the optimizer. The learning rate and the momentum were set to 0.05 and 0.9, respectively. After pretraining, we trained a linear classifier for 90 epochs with a batch size of 4,096. We set the hyperparameters for each of the methods to the same as the parameters for ImageNet classification. As baseline augmentation, we used the same augmentation as in SimSiam [2], namely, random cropping, random horizontal flipping, color jittering, and Gaussian blur. For more details and SimSiam's hyperparameters, please refer to [2].

B. Pseudo-Code of TeachAugment

We show the pseudo-code of TeachAugment in Algorithm 1.

C. Details of augmentation model

For the geometric augmentation, we used a three-layer perceptron. The dimension of the noise vector was 128 and the number of units in hidden layers was 512. As a nonlinear activation function, we used leaky ReLU [10] with the negative slope of 0.2. The output, A^{unnorm}, was normalized through the sigmoid function:

$$A = \lambda_{g_{\text{scale}}}(\text{sigmoid}(A^{\text{unnorm}}) - 0.5), \quad (2)$$

where $\lambda_{g_{\text{scale}}}$ controls the search range of A, and we set it to 0.5 (i.e., $A \in (-0.25, 0.25)^{2\times 3}$).

[1] The implementation of the baseline augmentation and models are based on https://github.com/kakaobrain/fast-autoaugment.

[2] The experimental code is based on https://github.com/facebookresearch/simsiam.

	WideResNet-40-2	WideResNet-28-10	Shake-Shake (26 2×96d)	PyramidNet	ResNet-50
Learning rate	0.1	0.1	0.01	0.05	0.05
Weight decay	2e-4	5e-4	1e-3	5e-5	1e-4
Epochs	200	200	1,800	1,800	270
Batch size	128	128	128	64	128
Learning rate decay	cosine	cosine	cosine	cosine	step

Table 1. Hyperparameters for classification tasks. We set parameters following [8]. Note that we show the batch size per GPU and the learning rate was multiplied by the number of GPUs (e.g., if two GPUs are used for training, the learning rate is doubled). We used a single GPU for WideResNet-40-2 and WideResNet-28-10, and four GPUs for the other models. We decayed the learning rate for ResNet-50 by 10-fold at epochs 90, 180, and 240.

Algorithm 1 Training procedure for TeachAugment

Input: A target model f_θ, a teacher model $f_{\hat{\theta}}$, dataset $\mathcal{X}$, the number of inner iterations n_{inner}, learning rate η_θ and η_ϕ, and decay rate for the EMA teacher ξ

while θ has not converged **do**
  for $i = 0, \cdots, n_{\text{inner}}$ **do**
    if $f_{\hat{\theta}}$ is the EMA teacher **then**
      Update teacher weights, $\hat{\theta} \leftarrow \xi\hat{\theta} + (1-\xi)\theta$
    end if
    Randomly sample a mini-batch, $\{x^b\}_b^B \sim \mathcal{X}$
    Compute loss for the target model, $L_\theta = \sum_b L(f_\theta(a_\phi(x^b)))$
    Update θ by the gradient descent, $\theta \leftarrow \theta - \eta_\theta \partial L_\theta / \partial\theta$
  end for
  Randomly sample a mini-batch, $\{\bar{x}^b\}_b^B \sim \mathcal{X}$
  Compute loss for the augmentation model, $L_\phi = \sum_b (L(f_\theta(a_\phi(\bar{x}^b))) - L(f_{\hat{\theta}}(a_\phi(\bar{x}^b))))$
  Update ϕ by the gradient ascent, $\phi \leftarrow \phi + \eta_\phi \partial L_\phi / \partial\phi$
end while

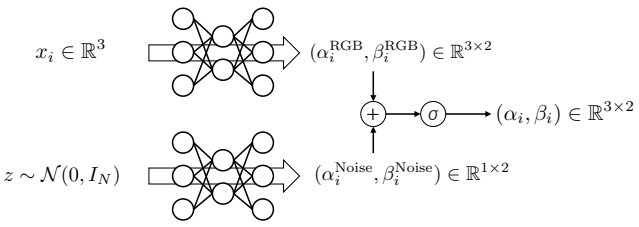


Figure 1. Illustration of the color augmentation model. σ denotes the sigmoid function.

For the color augmentation, we used two three-layer perceptrons that receive an RGB vector and a noise vector as inputs and add up their outputs. The number of units in hidden layers was 128 and 512, respectively. As the non-linear activation, leaky ReLU was used. We illustrate the computational scheme of the color augmentation model in Fig. 1. The former model outputs 3-dimensional scale and shift parameters:

$$(\alpha_i^{\text{RGB}}, \beta_i^{\text{RGB}}) \in \mathbb{R}^{3\times2}, \tag{3}$$

and the latter outputs scalar scale and shift parameters:

$$(\alpha_i^{\text{Noise}}, \beta_i^{\text{Noise}}) \in \mathbb{R}^{1\times2}. \tag{4}$$

The scale and shift parameters from the noise vector control the global brightness of images. Then, we add up these scale and shift parameters:

$$(\alpha_i^{\text{unnorm}})_j = (\alpha_i^{\text{RGB}})_j + \alpha_i^{\text{Noise}}, \tag{5}$$
$$(\beta_i^{\text{unnorm}})_j = (\beta_i^{\text{RGB}})_j + \beta_i^{\text{Noise}}, \tag{6}$$

where $(\alpha_i)_j$ denotes the j-th element of $\alpha_i \in \mathbb{R}^3$. We normalized the scale and shift parameters, $(\alpha_i^{\text{unnorm}}, \beta_i^{\text{unnorm}})$, using the sigmoid function:

$$\alpha_i = \lambda_{c_{\text{scale}}}(\text{sigmoid}(\alpha_i^{\text{unnorm}}) - 0.5) + 1, \tag{7}$$
$$\beta_i = \lambda_{c_{\text{scale}}}(\text{sigmoid}(\beta_i^{\text{unnorm}}) - 0.5), \tag{8}$$

where $\lambda_{c_{\text{scale}}}$ controls the search range of α_i and β_i, and we set it to 0.8, namely, $\alpha_i \in (0.6, 1.4)$ and $\beta_i \in (-0.4, 0.4)$.

We adopted AdamW [9] as the optimizer for the augmentation model. The learning rate and the weight decay were set to 1e-3 and 1e-2, which are the default parameters in PyTorch [13]. Dropout [14] was applied after the linear layers except for the output layer with the drop ratio of 0.8.

D. Learning pipeline of probabilities p_c and p_g

We made the decision process of applying the augmentation differentiable using weights sampled from the relaxed Bernoulli distribution defined in [11].[3] A sample from the relaxed Bernoulli, $w \sim \text{ReBern}(p, \tau)$, is obtained as follows:

$$w = \text{sigmoid}((L + \log p)/\tau),\ L \sim \text{Logistic}(0, 1), \tag{9}$$

[3]The relaxed Bernoulli distribution is referred to as the BinConcrete distribution in [11].

where τ and p denote the temperature parameter and a probability that corresponds to p_c and p_g in our case and L is a sample from the Logistic distribution, which is obtained by $L = \log(U) - \log(1 - U)$, $U \sim \mathrm{Uniform}(0, 1)$. We set τ to 0.05 following [5]. We note that the sampling procedure, Eq. 9, is differentiable with respect to the probability p.

We compute weighted sum of the parameters generated by augmentation models and parameters that make the augmentation the identity mapping:

$$\hat{\alpha}_i = w_c \alpha_i + (1 - w_c) \cdot 1, \tag{10}$$

$$\hat{\beta}_i = w_c \beta_i + (1 - w_c) \cdot 0, \tag{11}$$

$$\hat{A} = w_g A + (1 - w_g) I, \tag{12}$$

where $w_c \sim \mathrm{ReBern}(p_c, \tau)$ and $w_g \sim \mathrm{ReBern}(p_g, \tau)$. We use $(\hat{\alpha}_i, \hat{\beta}_i)$ and $\hat{A}$ to transform images, instead of (α_i, β_i) and A:

$$\tilde{x}_i = t(\hat{\alpha}_i \odot x_i + \hat{\beta}_i), \tag{13}$$

$$\hat{x} = \mathrm{Affine}(\tilde{x}, \hat{A} + I). \tag{14}$$

Because the sampling procedure from $\mathrm{ReBern}(p, \tau)$ is differentiable with respect to p, we can also update the probabilities using the gradient method.

E. Additional results

E.1. Relation between consistency regularization and TeachAugment

TeachAugment can be viewed as a method that minimizes the distance between predictions of the target model and the teacher model. We show it qualitatively.

We assume that data augmentation a_ϕ can transform an input data point to any point in the input space, and the teacher model is fixed during training. Then, solving $\max_\phi L(f_\theta(a_\phi(x))) - L(f_{\hat{\theta}}(a_\phi(x)))$ corresponds to searching the data augmentation maximizing the distance between the predictions under the constraint of $L(f_\theta(a_\phi(x))) > L(f_{\hat{\theta}}(a_\phi(x)))$. Then, the target model updates its decision boundary for minimizing the loss for the augmented data. This procedure is illustrated in Fig. 2. If the decision boundary of the target model completely corresponds to that of the teacher model, the data augmentation reaches the stationary point because the objective $L(f_\theta(a_\phi(x))) - L(f_{\hat{\theta}}(a_\phi(x)))$ is 0 for all data points.

However, the target model would not match the teacher model in practice, because above analysis lacks certain points: (1) The data augmentation cannot transform an input data point to any point in the input. (2) We used the non-saturating loss for the image classification tasks instead of cross entropy. Thus, $L(f_\theta(a_\phi(x))) - L(f_{\hat{\theta}}(a_\phi(x))) \neq 0$ if the decision boundary of the target model is the same as that of the teacher model. (3) TeachAugment does not explicitly minimize the distance between the predictions. In particular, (3) is a more critical point compared to (1) and (2) because they may be solved by adopting a model large enough to satisfy the requirement as the augmentation model, and using an original loss function instead of the non-saturating loss. Because of the lack of explicit costs for the consistency, TeachAugment does not ensure that the target model converges with the teacher model.

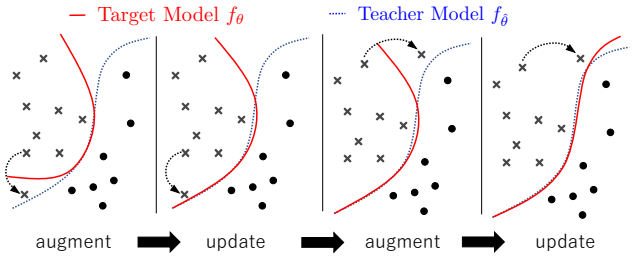


Figure 2. Illustration of the training process in TeachAugment. The data augmentation transforms a data point to the point so that $L(f_\theta(a_\phi(x))) > L(f_{\hat{\theta}}(a_\phi(x)))$. Then, the target model is updated to minimize loss for the augmented point. By repeating this process, the decision boundary of the target model is close to that of the teacher model.

However, we believe that the above analysis gives some insights, and comparing TeachAugment to consistency regularization is important. Thus, we trained model to minimize the following costs and compared the results to TeachAugment:

$$\min_\theta L(f_\theta(x)) + D(f_\theta(x), f_{\hat{\theta}}(x)), \tag{15}$$

where $D(\cdot, \cdot)$ denotes the distance function. We used the mean squared error and Kullback–Leibler divergence as $D(\cdot, \cdot)$. These functions are widely used in the consistency regularization and the knowledge distillation. We refer to the former as MSE Consistency and the later as KLD Consistency. Note that we use random horizontal flipping, random cropping, and cutout [3] as the data augmentation but omitted them in Eq. (15) because they were not optimized. As the teacher model, we used the same EMA teacher in TeachAugment.

The results are shown in Tab. 2. The consistency methods do not improve the error rates from the baseline. Thus, TeachAugment has different properties than the consistency regularization, and it works well in supervised learning.

E.2. Qualitative analysis

We show augmented images obtained by the augmentation model trained with various objective functions: TeachAugment, Adversarial AutoAugment (Adv. AA) [16], and PointAugment [7]. All methods used the same proposed augmentation model as data augmentation.

Dataset	CIFAR-10	CIFAR-100
Baseline	3.1	18.4
KLD consistency	3.1	18.2
MSE consistency	3.2	18.5
TeachAugment	**2.5**	**16.8**

Table 2. Comparison with the consistency regularization. We report the error rates of WideResNet-28-10. For training with KLD and MSE consistency, we used random horizontal flipping, random cropping, and cutout [3] as data augmentation.

The objective of Adv. AA is as follows:

$$\max_{\phi} \min_{\theta} L(f_\theta(a_\phi(x))). \tag{16}$$

Also, the objective of PointAugment is as follows:

$$\min_{\theta} L(f_\theta(a_\phi(x))), \tag{17}$$

$$\min_{\phi} L(f_\theta(a_\phi(x))) + |1 - \exp(L(f_\theta(a_\phi(x))) - \rho L(f_\theta(x)))|, \tag{18}$$

where ρ is a dynamic parameter defined as $\rho = \exp(y^T \cdot f_\theta(a_\phi(x)))$. Note that Adv. AA updates augmentation functions using the REINFORCE algorithm [15] because many functions in the search space of AutoAugment are non-differentiable. However, in our experiments, because our proposed augmentation is differentiable, we updated augmentation with the gradient descent rather than the REINFORCE algorithm.

The augmented images of CIFAR-10 and CIFAR-100 are shown in Figs. 3 and 4. The images augmented by Adv. AA obviously collapse, so the meaningful information is lost. The augmented images obtained by PointAugment and the proposed method are recognizable, but the proposed method transforms images more strongly than PointAugment so that the proposed method distorts the aspect ratio for the geometric augmentation. PointAugment binds the difficulty of the augmented images through a dynamic parameter $\rho \leq \exp(1)$, but the proposed method requires only that the augmented images are recognizable for the teacher model. As a result, the proposed method allows stronger augmentation than PointAugment.

E.3. Example of augmented images

We show example results of augmentation for ImageNet and Cityscapes in Figs. 5 and 6. As can see from Fig. 6, the augmentation obtained for FCN-32s is obviously different from the others. Because the output stride of FCN-32s (i.e., the ratio of input image spatial resolution to final output resolution) is lower than that of the others, FCN-32s will have different properties than PSPNet and Deeplav3. We believe that the difference leads the different augmentation of these models. Moreover, it also leads the degradation of mIoU for FCN-32s in RandAugment and TrivialAugment.

References

[1] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 1

[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 1

[3] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3, 4

[4] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 1

[5] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020. 3

[6] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1

[7] Ruihui Li, Xianzhi Li, Pheng-Ann Heng, and Chi-Wing Fu. Pointaugment: an auto-augmentation framework for point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6378–6387, 2020. 3

[8] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32:6665–6675, 2019. 2

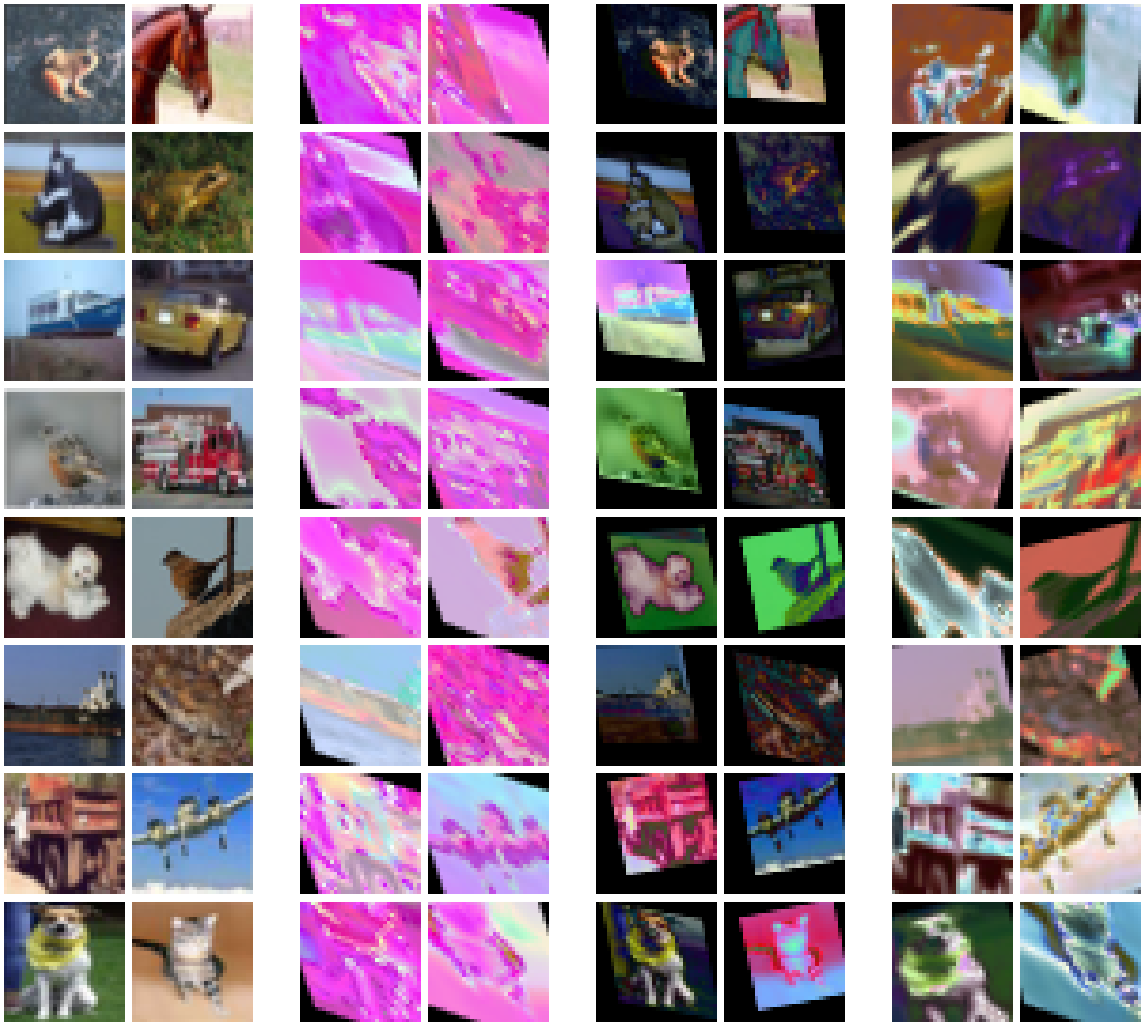
[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 2

[10] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013. 1

[11] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016. 2

[12] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983. 1

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library.



(a) Original Image (b) Adv. AA (c) PointAugment (d) Ours

Figure 3. Augmented images obtained with CIFAR-10 using various methods..

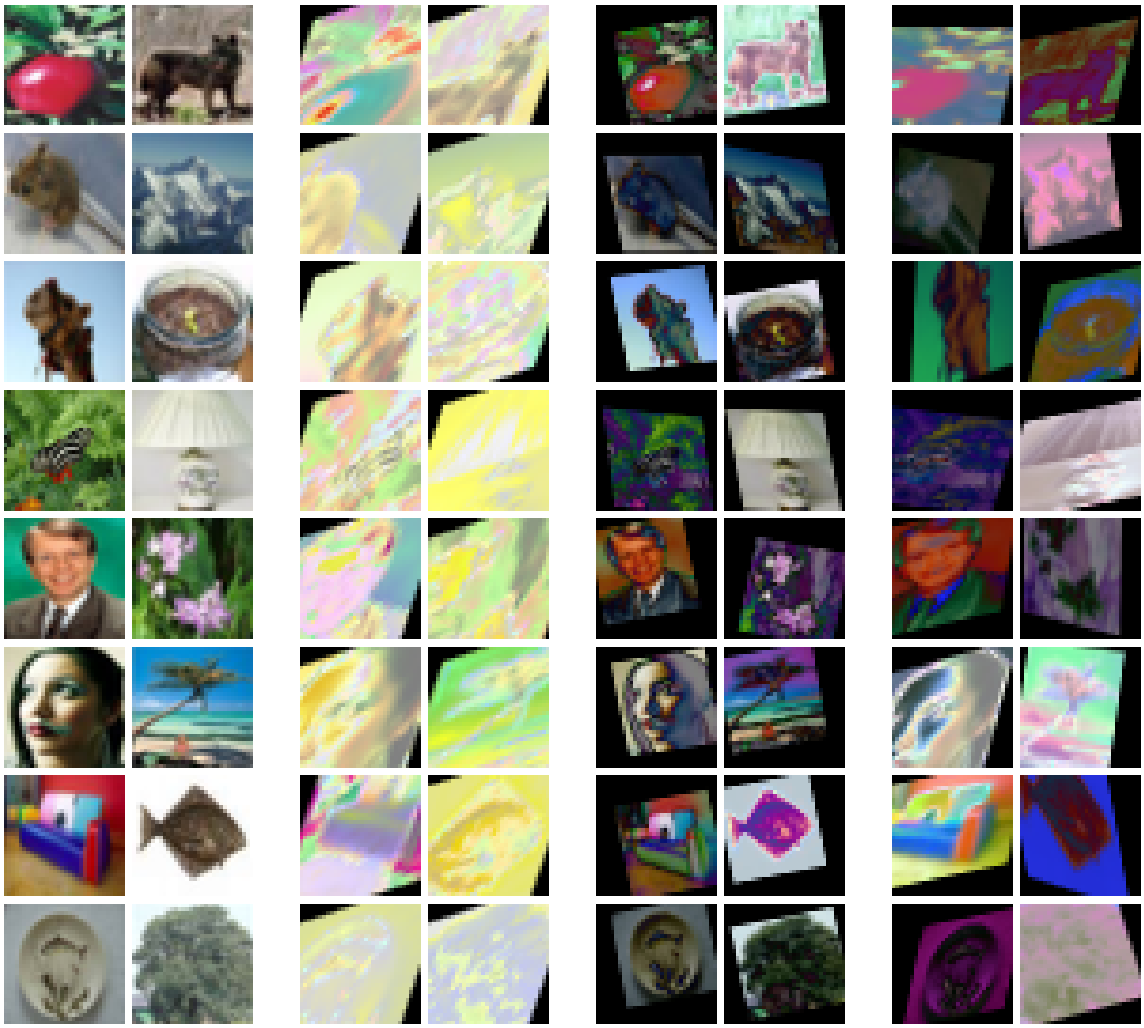
In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 2

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014. 2

[15] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992. 4

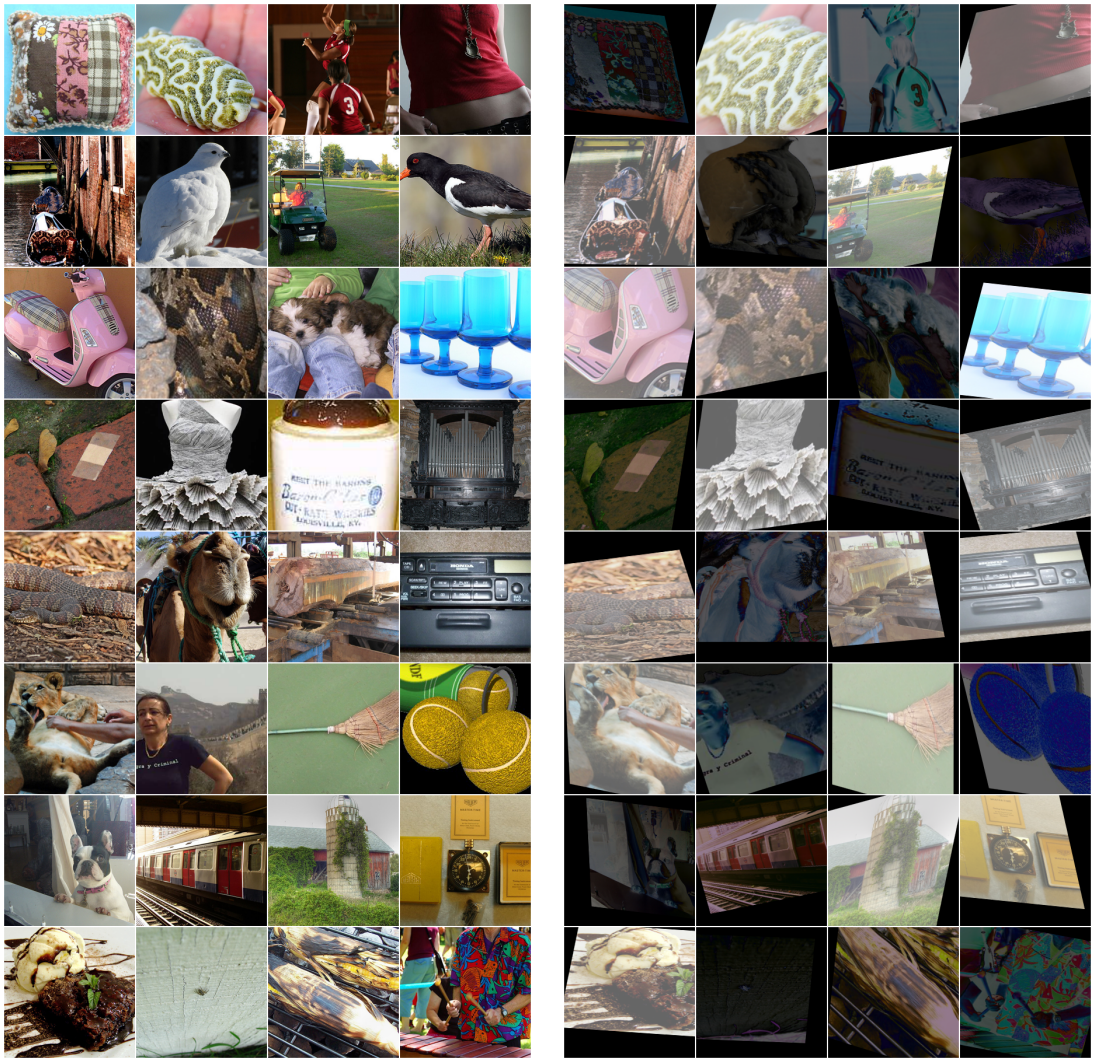
[16] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. *arXiv preprint arXiv:1912.11188*, 2019. 3

[17] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 1



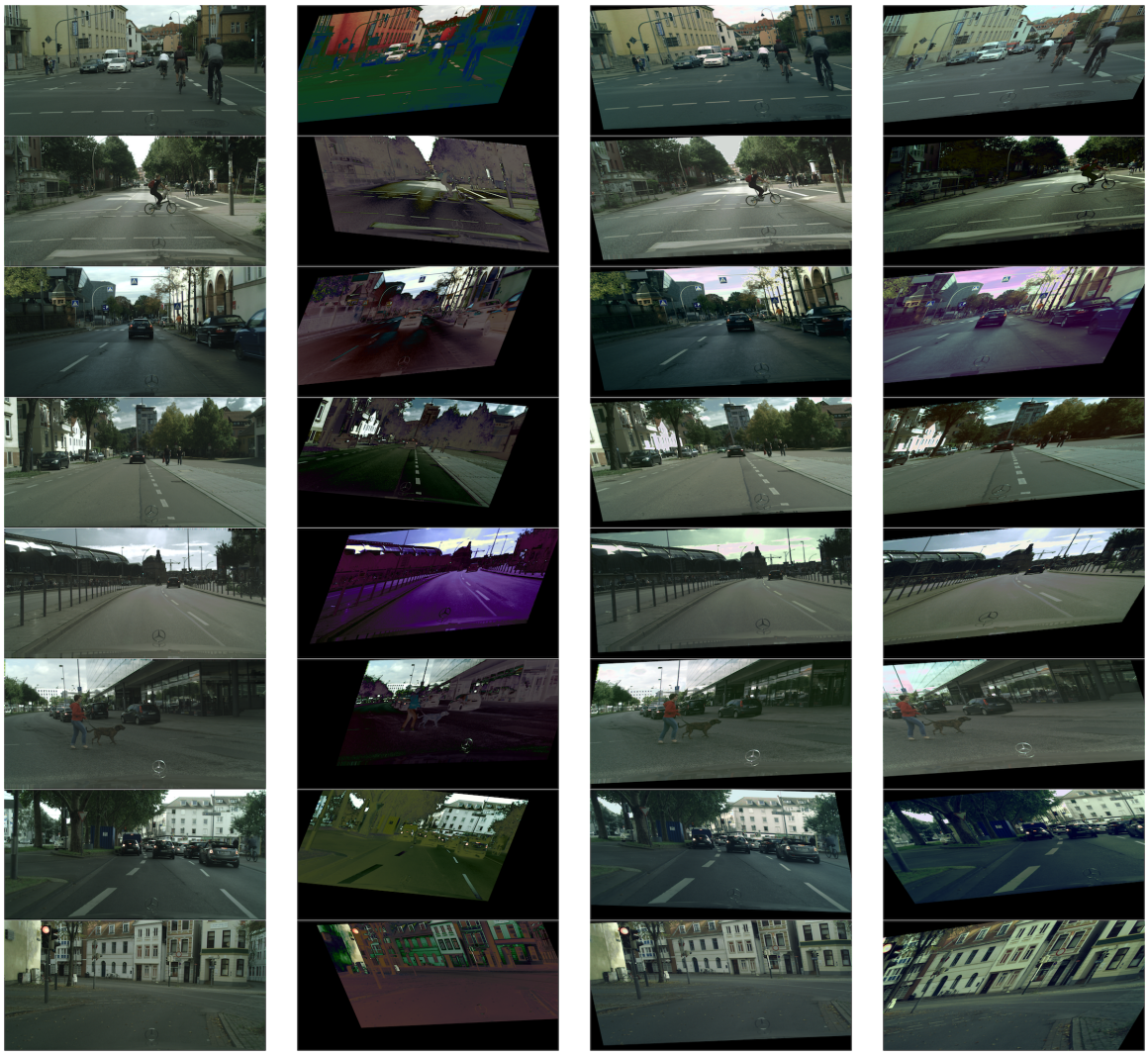
(a) Original Image (b) Adv. AA (c) PointAugment (d) Ours

Figure 4. Augmented images obtained with CIFAR-100 using various methods..



(a) Original images (b) Augmented images

Figure 5. Augmentations obtained with ImageNet.



(a) Original images (b) FCN-32s (c) PSPNet (d) Deeplabv3

Figure 6. Augmentations obtained with Cityscapes.