

## SUPPLEMENTARY MATERIAL

### A TRAINING METHODS

Cifar10 models were trained using code based on AutoAugment code<sup>2</sup> using the following choices:

1. Learning rate was decayed following a cosine decay schedule, starting with a value of 0.1
2. 78050 training steps were used, with data shuffled after every epoch.
3. As implemented in the AutoAugment code, the WRN-28-2 model was used with stochastic gradient descent and momentum. The optimizer used cross entropy loss with  $\ell_2$  weight decay of 0.0005.
4. Before selecting the validation set, the full training set was shuffled and balanced such that the subset selected for training was balanced across classes.
5. Validation set was the last 5000 samples of the shuffled CIFAR-10 training data.
6. Models were trained using Python 2.7 and TensorFlow 1.13 .

A training time of 78k steps was chosen because it showed reasonable convergence with the standard data augmentation of `FlipLR`, `Crop`, and `Cutout`. In the clean baseline case, test accuracy actually reached its peak much earlier than 78k steps.

With CIFAR-10, experiments were also performed for training dataset sizes of 1024, 4096, and 16384. At smaller dataset sizes, the impact of augmentation and the Switch-off Lift tended to be larger. These results are not shown in this paper.

ImageNet models were ResNet-50 trained using the Cloud TPU codebase<sup>3</sup>. Models were trained for 112.6k steps with a weight decay rate of  $1e-4$ , and a learning rate of 0.2, which was decayed by 10 at epochs 30, 60, and 80. Batch size was set to be 1024.

### B DETAILS OF AUGMENTATION

#### B.1 CIFAR-10

On CIFAR-10, both color and affine transforms were tested, as given in the full results (see Sec. [removed for anonymization]). Most augmentations were as defined in Cubuk et al. (2018) and additional conventions for augmentations as labeled in Fig. 7 are defined here. For `Rotate`, *fixed* means each augmented image was rotated by exactly the stated amount, with a randomly-chosen direction. *Variable* means an augmented image was rotated a random amount up to the given value in a randomly-chosen direction. `Shear` is defined similarly. `Rotate(square)` means that an image was rotated by an amount chosen randomly from  $[0^\circ, 90^\circ, 180^\circ, 270^\circ]$ .

`Crop` included a padding before the random-location crop so that the final image remained  $32 \times 32$  in size. The magnitude given for `Crop` is the number of pixels that were added in each dimension. The magnitude given in the label for `Cutout` is the size, in pixels, of each dimension of the square cutout.

`PatchGaussian` was defined as in Lopes et al. (2019), with the patch specified to be contained entirely within the image domain. In Fig. 7 it is labeled by two hyperparameters: the size of the square patch (in pixels) that was applied and  $\sigma_{max}$ , which is the maximum standard deviation of the noise that could be selected for any given patch. Here, “fixed” means the patch size was always the same.

Since `FlipLR`, `Crop`, and `Cutout` are part of standard pipelines for CIFAR-10 (Hernández-García & König, 2018a; Goodfellow et al., 2013; Springenberg et al., 2014), we tested combinations of the three augmentations (varying probabilities of each) as well as these three augmentations plus an single additional augmentation. As in standard processing of CIFAR-10 images, the first augmentation applied was anything that is not one of `FlipLR`, `Crop`, or `Cutout`. After that, augmentations were applied in the order `Crop`, then `FlipLR`, then `Cutout`.

<sup>2</sup>available at [github.com/tensorflow/models/tree/master/research/autoaugment](https://github.com/tensorflow/models/tree/master/research/autoaugment)

<sup>3</sup>available at <https://github.com/tensorflow/tpu/tree/master/models/official/resnet>

Finally, we tested the CIFAR-10 AutoAugment policy (Cubuk et al. 2018), RandAugment (Cubuk et al. 2019), and mixup (Zhang et al. 2017). The hyperparameters for these augmentations followed the guidelines described in the respective papers.

These augmentations are labeled in Fig. 7.

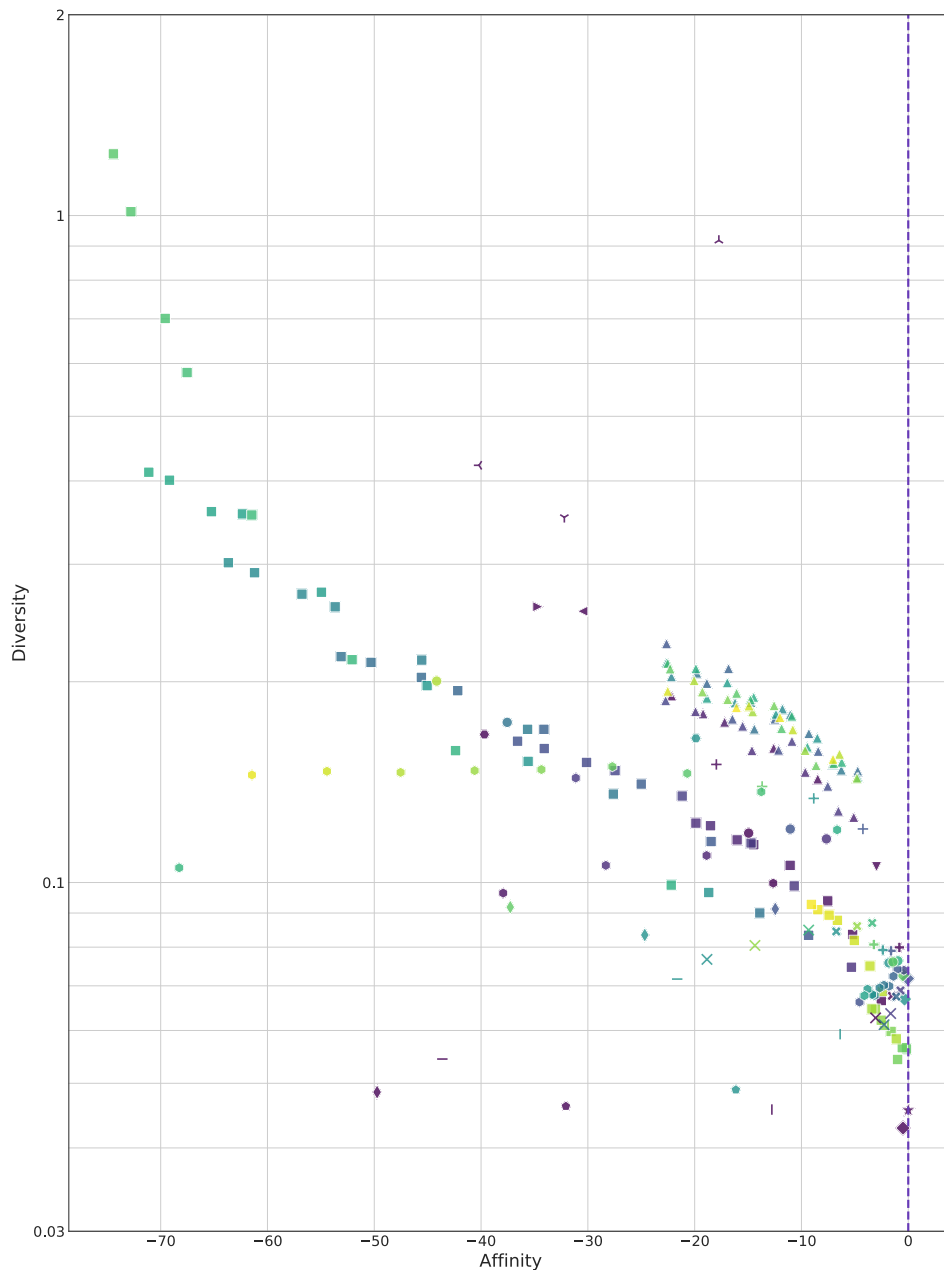


Figure 7: CIFAR-10: Labeled map of tested augmentations on the plane of Affinity and Diversity. Color distinguishes different hyperparameters for a given transform. Legend is below.

★ Clean	✳ Invert(100%)	Equalize(100%)
⤴ Autoaugment	✳ Invert(50%)	Equalize(50%)
⤵ Mixup	✳ ShearX(variable, 0.1, 100%)	▼ FlipLR(50%) + Crop(4,100%)
⤵ Randaug	✳ ShearX(variable, 0.1, 50%)	▲ FlipLR(100%) + Crop(4,100%) + Cutout(16,100%)
■ PatchGaussian(fixed, 12, 0.1, 100%)	✳ ShearX(variable, 0.1, 75%)	▲ FlipLR(100%) + Crop(4,100%) + Cutout(16,25%)
■ PatchGaussian(fixed, 12, 0.2, 100%)	✳ ShearX(variable, 0.3, 100%)	▲ FlipLR(100%) + Crop(4,100%) + Cutout(16,50%)
■ PatchGaussian(fixed, 12, 0.3, 100%)	✳ ShearX(variable, 0.3, 50%)	▲ FlipLR(100%) + Crop(4,100%) + Cutout(16,75%)
■ PatchGaussian(fixed, 12, 0.5, 100%)	✳ ShearX(variable, 0.3, 75%)	▲ FlipLR(100%) + Crop(4,25%) + Cutout(16,100%)
■ PatchGaussian(fixed, 12, 0.8, 100%)	✳ ShearX(fixed, 0.1, 100%)	▲ FlipLR(100%) + Crop(4,25%) + Cutout(16,25%)
■ PatchGaussian(fixed, 12, 1.0, 100%)	✳ ShearX(fixed, 0.1, 50%)	▲ FlipLR(100%) + Crop(4,25%) + Cutout(16,50%)
■ PatchGaussian(fixed, 12, 1.5, 100%)	✳ ShearX(fixed, 0.1, 75%)	▲ FlipLR(100%) + Crop(4,25%) + Cutout(16,75%)
■ PatchGaussian(fixed, 12, 2.0, 100%)	✳ ShearX(fixed, 0.3, 100%)	▲ FlipLR(100%) + Crop(4,50%) + Cutout(16,100%)
■ PatchGaussian(fixed, 16, 0.1, 100%)	✳ ShearX(fixed, 0.3, 50%)	▲ FlipLR(100%) + Crop(4,50%) + Cutout(16,25%)
■ PatchGaussian(fixed, 16, 0.2, 100%)	✳ ShearX(fixed, 0.3, 75%)	▲ FlipLR(100%) + Crop(4,50%) + Cutout(16,50%)
■ PatchGaussian(fixed, 16, 0.3, 100%)	● Rotate(variable, 20deg, 100%)	▲ FlipLR(100%) + Crop(4,50%) + Cutout(16,75%)
■ PatchGaussian(fixed, 16, 0.5, 100%)	● Rotate(variable, 20deg, 50%)	▲ FlipLR(100%) + Crop(4,75%) + Cutout(16,100%)
■ PatchGaussian(fixed, 16, 0.8, 100%)	● Rotate(variable, 20deg, 75%)	▲ FlipLR(100%) + Crop(4,75%) + Cutout(16,25%)
■ PatchGaussian(fixed, 16, 1.0, 100%)	● Rotate(variable, 45, 100%)	▲ FlipLR(100%) + Crop(4,75%) + Cutout(16,50%)
■ PatchGaussian(fixed, 16, 1.5, 100%)	● Rotate(variable, 5deg, 100%)	▲ FlipLR(100%) + Crop(4,75%) + Cutout(16,75%)
■ PatchGaussian(fixed, 16, 2.0, 100%)	● Rotate(variable, 5deg, 50%)	▲ FlipLR(25%) + Crop(4,100%) + Cutout(16,100%)
■ PatchGaussian(fixed, 20, 0.1, 100%)	● Rotate(variable, 5deg, 75%)	▲ FlipLR(25%) + Crop(4,100%) + Cutout(16,25%)
■ PatchGaussian(fixed, 20, 0.2, 100%)	● Rotate(variable, 60deg, 100%)	▲ FlipLR(25%) + Crop(4,100%) + Cutout(16,50%)
■ PatchGaussian(fixed, 20, 0.3, 100%)	● Rotate(fixed, 15deg, 50%)	▲ FlipLR(25%) + Crop(4,100%) + Cutout(16,75%)
■ PatchGaussian(fixed, 20, 0.5, 100%)	● Rotate(fixed, 20deg, 100%)	▲ FlipLR(25%) + Crop(4,25%) + Cutout(16,100%)
■ PatchGaussian(fixed, 20, 0.8, 100%)	● Rotate(fixed, 20deg, 50%)	▲ FlipLR(25%) + Crop(4,25%) + Cutout(16,25%)
■ PatchGaussian(fixed, 20, 1.0, 100%)	● Rotate(fixed, 20deg, 75%)	▲ FlipLR(25%) + Crop(4,25%) + Cutout(16,50%)
■ PatchGaussian(fixed, 20, 1.5, 100%)	● Rotate(fixed, 45deg, 50%)	▲ FlipLR(25%) + Crop(4,25%) + Cutout(16,75%)
■ PatchGaussian(fixed, 20, 2.0, 100%)	● Rotate(fixed, 5deg, 100%)	▲ FlipLR(25%) + Crop(4,50%) + Cutout(16,100%)
■ PatchGaussian(fixed, 24, 0.1, 100%)	● Rotate(fixed, 5deg, 100%)	▲ FlipLR(25%) + Crop(4,50%) + Cutout(16,25%)
■ PatchGaussian(fixed, 24, 0.2, 100%)	● Rotate(fixed, 5deg, 20%)	▲ FlipLR(25%) + Crop(4,50%) + Cutout(16,50%)
■ PatchGaussian(fixed, 24, 0.3, 100%)	● Rotate(fixed, 5deg, 30%)	▲ FlipLR(25%) + Crop(4,50%) + Cutout(16,75%)
■ PatchGaussian(fixed, 24, 0.5, 100%)	● Rotate(fixed, 5deg, 40%)	▲ FlipLR(25%) + Crop(4,75%) + Cutout(16,100%)
■ PatchGaussian(fixed, 24, 0.8, 100%)	● Rotate(fixed, 5deg, 50%)	▲ FlipLR(25%) + Crop(4,75%) + Cutout(16,25%)
■ PatchGaussian(fixed, 24, 1.0, 100%)	● Rotate(fixed, 5deg, 60%)	▲ FlipLR(25%) + Crop(4,75%) + Cutout(16,50%)
■ PatchGaussian(fixed, 24, 1.5, 100%)	● Rotate(fixed, 5deg, 70%)	▲ FlipLR(25%) + Crop(4,75%) + Cutout(16,75%)
■ PatchGaussian(fixed, 24, 2.0, 100%)	● Rotate(fixed, 5deg, 75%)	▲ FlipLR(50%) + Crop(4,100%) + Cutout(16,100%)
■ PatchGaussian(fixed, 28, 0.1, 100%)	● Rotate(fixed, 5deg, 80%)	▲ FlipLR(50%) + Crop(4,100%) + Cutout(16,25%)
■ PatchGaussian(fixed, 28, 0.2, 100%)	● Rotate(fixed, 5deg, 90%)	▲ FlipLR(50%) + Crop(4,100%) + Cutout(16,50%)
■ PatchGaussian(fixed, 28, 0.3, 100%)	● Rotate(fixed, 60deg, 10%)	▲ FlipLR(50%) + Crop(4,100%) + Cutout(16,75%)
■ PatchGaussian(fixed, 28, 0.5, 100%)	● Rotate(fixed, 60deg, 100%)	▲ FlipLR(50%) + Crop(4,25%) + Cutout(16,100%)
■ PatchGaussian(fixed, 28, 0.8, 100%)	● Rotate(fixed, 60deg, 20%)	▲ FlipLR(50%) + Crop(4,25%) + Cutout(16,25%)
■ PatchGaussian(fixed, 28, 1.0, 100%)	● Rotate(fixed, 60deg, 30%)	▲ FlipLR(50%) + Crop(4,25%) + Cutout(16,50%)
■ PatchGaussian(fixed, 28, 1.5, 100%)	● Rotate(fixed, 60deg, 40%)	▲ FlipLR(50%) + Crop(4,25%) + Cutout(16,75%)
■ PatchGaussian(fixed, 28, 2.0, 100%)	● Rotate(fixed, 60deg, 50%)	▲ FlipLR(50%) + Crop(4,50%) + Cutout(16,100%)
■ PatchGaussian(fixed, 32, 0.1, 100%)	● Rotate(fixed, 60deg, 60%)	▲ FlipLR(50%) + Crop(4,50%) + Cutout(16,25%)
■ PatchGaussian(fixed, 32, 0.2, 100%)	● Rotate(fixed, 60deg, 70%)	▲ FlipLR(50%) + Crop(4,50%) + Cutout(16,50%)
■ PatchGaussian(fixed, 32, 0.3, 100%)	● Rotate(fixed, 60deg, 80%)	▲ FlipLR(50%) + Crop(4,50%) + Cutout(16,75%)
■ PatchGaussian(fixed, 32, 0.5, 100%)	● Rotate(fixed, 60deg, 90%)	▲ FlipLR(50%) + Crop(4,75%) + Cutout(16,100%)
■ PatchGaussian(fixed, 32, 0.8, 100%)	● Rotate(square, 100%)	▲ FlipLR(50%) + Crop(4,75%) + Cutout(16,25%)
■ PatchGaussian(fixed, 32, 1.0, 100%)	● Rotate(square, 50%)	▲ FlipLR(50%) + Crop(4,75%) + Cutout(16,50%)
■ PatchGaussian(fixed, 32, 1.5, 100%)	— Blur(100%)	▲ FlipLR(50%) + Crop(4,75%) + Cutout(16,75%)
■ PatchGaussian(fixed, 32, 2.0, 100%)	— Blur(50%)	▲ FlipLR(75%) + Crop(4,100%) + Cutout(16,100%)
■ PatchGaussian(fixed, 4, 0.1, 100%)	◆ FlipLR(100%)	▲ FlipLR(75%) + Crop(4,100%) + Cutout(16,25%)
■ PatchGaussian(fixed, 4, 0.2, 100%)	◆ FlipLR(25%)	▲ FlipLR(75%) + Crop(4,100%) + Cutout(16,50%)
■ PatchGaussian(fixed, 4, 0.3, 100%)	◆ FlipLR(50%)	▲ FlipLR(75%) + Crop(4,100%) + Cutout(16,75%)
■ PatchGaussian(fixed, 4, 0.5, 100%)	◆ FlipLR(75%)	▲ FlipLR(75%) + Crop(4,25%) + Cutout(16,100%)
■ PatchGaussian(fixed, 4, 0.8, 100%)	◆ FlipUD(100%)	▲ FlipLR(75%) + Crop(4,25%) + Cutout(16,25%)
■ PatchGaussian(fixed, 4, 1.0, 100%)	◆ FlipUD(25%)	▲ FlipLR(75%) + Crop(4,25%) + Cutout(16,50%)
■ PatchGaussian(fixed, 4, 1.5, 100%)	◆ FlipUD(50%)	▲ FlipLR(75%) + Crop(4,25%) + Cutout(16,75%)
■ PatchGaussian(fixed, 4, 2.0, 100%)	◆ FlipUD(75%)	▲ FlipLR(75%) + Crop(4,50%) + Cutout(16,100%)
■ PatchGaussian(fixed, 8, 0.1, 100%)	✳ Crop(4, 25%)	▲ FlipLR(75%) + Crop(4,50%) + Cutout(16,25%)
■ PatchGaussian(fixed, 8, 0.2, 100%)	✳ Crop(4, 50%)	▲ FlipLR(75%) + Crop(4,50%) + Cutout(16,50%)
■ PatchGaussian(fixed, 8, 0.3, 100%)	✳ Crop(4, 75%)	▲ FlipLR(75%) + Crop(4,50%) + Cutout(16,75%)
■ PatchGaussian(fixed, 8, 0.5, 100%)	✳ Crop(4, 100%)	▲ FlipLR(75%) + Crop(4,75%) + Cutout(16,100%)
■ PatchGaussian(fixed, 8, 0.8, 100%)	✳ Cutout(16, 100%)	▲ FlipLR(75%) + Crop(4,75%) + Cutout(16,25%)
■ PatchGaussian(fixed, 8, 1.0, 100%)	✳ Cutout(16, 25%)	▲ FlipLR(75%) + Crop(4,75%) + Cutout(16,50%)
■ PatchGaussian(fixed, 8, 1.5, 100%)	✳ Cutout(16, 50%)	▲ FlipLR(75%) + Crop(4,75%) + Cutout(16,75%)
■ PatchGaussian(fixed, 8, 2.0, 100%)	✳ Cutout(16, 75%)	▲ FlipLR(50%) + Crop(4, 100%) + Cutout(16, 100%) + Equalize(50%)
		▲ FlipLR(50%) + Crop(4, 100%) + Cutout(16, 100%) + Rotate(fixed, 15deg, 50%)

## B.2 IMAGENET

On ImageNet, we experimented with PatchGaussian, Cutout, operations from the PIL imaging library<sup>4</sup> and techniques from the AutoAugment code, as described above for CIFAR-10. In addition to PatchGaussian(fixed), we also tested PatchGaussian(variable), where the patch size was uniformly sampled up to a maximum size. The implementation here did not constrain the patch to be entirely contained within the image. Additionally, we experimented with SolarizeAdd. SolarizeAdd is similar to Solarize from the PIL library, but has an additional hyperparameter which determines how much value was added to each pixel that is below the threshold. Finally, we also experimented with Full Gaussian and Random Erasing on ImageNet. Full Gaussian adds Gaussian noise to the whole image. Random Erasing is similar to Cutout, but randomly

<sup>4</sup><https://pillow.readthedocs.io/en/5.1.x/>

samples the values of the pixels in the patch (Zhong et al., 2017) (whereas `Cutout` sets them to a constant, gray pixel).

These augmentations are labeled in Fig. 8.

We also experimented with combinations of transformations, using augmentations used in the AutoAugment policy. These are labeled `RandomMultiAug(x, y, z)`, where  $x$  is the number of augmentations per subpolicy,  $y$  is the number of subpolicies, and  $z$  is 0, or 1, indicating two randomized policies ran.

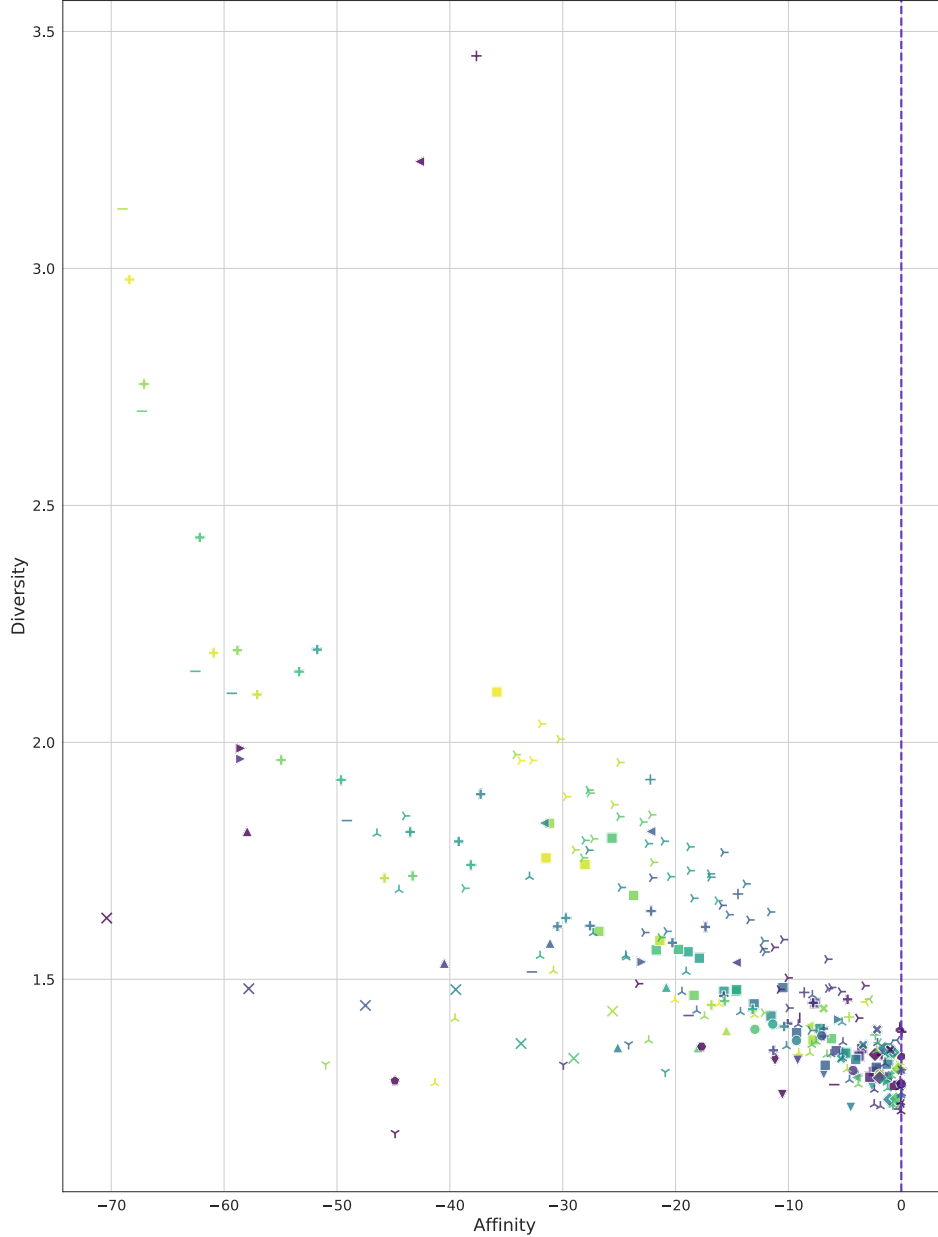


Figure 8: ImageNet: Labeled map of tested augmentations on the plane of Affinity and Diversity. Color distinguishes different hyperparameters for a given transform. Legend is below.

AutoContrast(100%)	Patch Gaussian(fixed, 200, 1.0, 100%)	Solarize Add(-127, 100, 100%)
Brightness(0.1, 100%)	Patch Gaussian(fixed, 200, 2.0, 100%)	Solarize Add(-127, 150, 100%)
Brightness(0.2, 100%)	Patch Gaussian(fixed, 250, 0.2, 100%)	Solarize Add(-127, 200, 100%)
Brightness(0.3, 100%)	Patch Gaussian(fixed, 250, 0.5, 100%)	Solarize Add(-127, 250, 100%)
Brightness(0.4, 100%)	Patch Gaussian(fixed, 250, 0.8, 100%)	Solarize Add(0023, 050, 100%)
Brightness(0.5, 100%)	Patch Gaussian(fixed, 250, 1.0, 100%)	Solarize Add(0023, 100, 100%)
Brightness(0.6, 100%)	Patch Gaussian(fixed, 250, 2.0, 100%)	Solarize Add(0023, 150, 100%)
Brightness(0.7, 100%)	Patch Gaussian(fixed, 300, 0.2, 100%)	Solarize Add(0023, 200, 100%)
Clean	Patch Gaussian(fixed, 300, 0.5, 100%)	Solarize Add(0023, 250, 100%)
Color(0.1, 100%)	Patch Gaussian(fixed, 300, 0.8, 100%)	Solarize Add(0048, 050, 100%)
Color(0.2, 100%)	Patch Gaussian(fixed, 300, 1.0, 100%)	Solarize Add(0048, 100, 100%)
Color(0.3, 100%)	Patch Gaussian(fixed, 300, 2.0, 100%)	Solarize Add(0048, 150, 100%)
Color(0.4, 100%)	Patch Gaussian(fixed, 350, 0.2, 100%)	Solarize Add(0048, 200, 100%)
Color(0.5, 100%)	Patch Gaussian(fixed, 350, 0.5, 100%)	Solarize Add(0048, 250, 100%)
Color(0.6, 100%)	Patch Gaussian(fixed, 350, 0.8, 100%)	Solarize Add(0073, 100, 100%)
Color(0.7, 100%)	Patch Gaussian(fixed, 350, 1.0, 100%)	Solarize Add(0073, 150, 100%)
Contrast(0.1, 100%)	Patch Gaussian(fixed, 350, 2.0, 100%)	Solarize Add(0073, 200, 100%)
Contrast(0.2, 100%)	Patch Gaussian(fixed, 400, 0.2, 100%)	Solarize Add(0073, 250, 100%)
Contrast(0.3, 100%)	Patch Gaussian(fixed, 400, 0.5, 100%)	Solarize Add(0098, 100, 100%)
Contrast(0.4, 100%)	Patch Gaussian(fixed, 400, 0.8, 100%)	Solarize Add(0098, 150, 100%)
Contrast(0.5, 100%)	Patch Gaussian(fixed, 400, 1.0, 100%)	Solarize Add(0098, 200, 100%)
Contrast(0.6, 100%)	Patch Gaussian(fixed, 400, 2.0, 100%)	Solarize Add(0098, 250, 100%)
Contrast(0.7, 100%)	Posterize(0, 100%)	Solarize Add(0123, 150, 100%)
Cutout(variable, 448, 100%)	Posterize(1, 100%)	Solarize Add(0123, 200, 100%)
Cutout(fixed, 120, 100%)	Posterize(2, 100%)	Solarize Add(0123, 250, 100%)
Cutout(fixed, 150, 100%)	Posterize(3, 100%)	TranslateX(0, 100%)
Cutout(fixed, 180, 100%)	Posterize(4, 100%)	TranslateX(10, 100%)
Cutout(fixed, 30, 100%)	Posterize(5, 100%)	TranslateX(20, 100%)
Cutout(fixed, 60, 100%)	Posterize(6, 100%)	TranslateX(30, 100%)
Cutout(fixed, 90, 100%)	Posterize(7, 100%)	TranslateX(40, 100%)
Equalize(100%)	Random Erasing(variable, 448, 100%)	TranslateX(50, 100%)
FlipUD(100%)	Random Erasing(fixed, 120, 100%)	TranslateX(60, 100%)
FullGaussian(0.1, 100%)	Random Erasing(fixed, 150, 100%)	TranslateX(70, 100%)
FullGaussian(0.2, 100%)	Random Erasing(fixed, 180, 100%)	TranslateX(80, 100%)
FullGaussian(0.3, 100%)	Random Erasing(fixed, 30, 100%)	TranslateX(90, 100%)
FullGaussian(0.5, 100%)	Random Erasing(fixed, 60, 100%)	RandomMultiAug(1, 1, 0)
FullGaussian(0.8, 100%)	Random Erasing(fixed, 90, 100%)	RandomMultiAug(1, 1, 1)
FullGaussian(1.0, 100%)	Rotate(variable, 10deg, 100%)	RandomMultiAug(1, 3, 0)
FullGaussian(1.5, 100%)	Rotate(variable, 15deg, 100%)	RandomMultiAug(1, 3, 1)
FullGaussian(2.0, 100%)	Rotate(variable, 20deg, 100%)	RandomMultiAug(1, 7, 0)
Rotate(square, 100%)	Rotate(variable, 25deg, 100%)	RandomMultiAug(1, 7, 1)
Invert(100%)	Rotate(variable, 30deg, 100%)	RandomMultiAug(1, 10, 0)
Patch Gaussian(variable, 100, 0.2, 100%)	Rotate(variable, 30deg, 100%)	RandomMultiAug(1, 10, 1)
Patch Gaussian(variable, 100, 0.5, 100%)	Rotate(variable, 5deg, 100%)	RandomMultiAug(1, 15, 0)
Patch Gaussian(variable, 100, 0.8, 100%)	Sharpness(0.1, 100%)	RandomMultiAug(1, 15, 1)
Patch Gaussian(variable, 100, 1.0, 100%)	Sharpness(0.2, 100%)	RandomMultiAug(1, 20, 0)
Patch Gaussian(variable, 100, 2.0, 100%)	Sharpness(0.3, 100%)	RandomMultiAug(1, 20, 1)
Patch Gaussian(variable, 150, 0.2, 100%)	Sharpness(0.4, 100%)	RandomMultiAug(2, 1, 0)
Patch Gaussian(variable, 150, 0.5, 100%)	Sharpness(0.5, 100%)	RandomMultiAug(2, 1, 1)
Patch Gaussian(variable, 150, 0.8, 100%)	Sharpness(0.6, 100%)	RandomMultiAug(2, 3, 0)
Patch Gaussian(variable, 150, 1.0, 100%)	Sharpness(0.7, 100%)	RandomMultiAug(2, 3, 1)
Patch Gaussian(variable, 150, 2.0, 100%)	ShearX(variable, 0.1, 100%)	RandomMultiAug(2, 7, 0)
Patch Gaussian(variable, 200, 0.2, 100%)	ShearX(variable, 0.2, 100%)	RandomMultiAug(2, 7, 1)
Patch Gaussian(variable, 200, 0.5, 100%)	ShearX(variable, 0.3, 100%)	RandomMultiAug(2, 10, 0)
Patch Gaussian(variable, 200, 0.8, 100%)	ShearX(variable, 0.4, 100%)	RandomMultiAug(2, 10, 1)
Patch Gaussian(variable, 200, 1.0, 100%)	ShearX(variable, 0.5, 100%)	RandomMultiAug(2, 15, 0)
Patch Gaussian(variable, 200, 2.0, 100%)	Solarize(0, 100%)	RandomMultiAug(2, 15, 1)
Patch Gaussian(variable, 250, 0.1, 100%)	Solarize(100, 100%)	RandomMultiAug(2, 20, 0)
Patch Gaussian(variable, 250, 0.2, 100%)	Solarize(150, 100%)	RandomMultiAug(2, 20, 1)
Patch Gaussian(variable, 250, 0.3, 100%)	Solarize(200, 100%)	RandomMultiAug(3, 1, 0)
Patch Gaussian(variable, 250, 0.5, 100%)	Solarize(250, 100%)	RandomMultiAug(3, 1, 1)
Patch Gaussian(variable, 250, 0.8, 100%)	Solarize(50, 100%)	RandomMultiAug(3, 3, 0)
Patch Gaussian(variable, 250, 1.0, 100%)	Solarize Add(-002, 000, 100%)	RandomMultiAug(3, 3, 1)
Patch Gaussian(variable, 250, 1.5, 100%)	Solarize Add(-002, 050, 100%)	RandomMultiAug(3, 7, 0)
Patch Gaussian(variable, 250, 2.0, 100%)	Solarize Add(-002, 100, 100%)	RandomMultiAug(3, 7, 1)
Patch Gaussian(variable, 300, 0.2, 100%)	Solarize Add(-002, 150, 100%)	RandomMultiAug(3, 10, 0)
Patch Gaussian(variable, 300, 0.5, 100%)	Solarize Add(-002, 200, 100%)	RandomMultiAug(3, 10, 1)
Patch Gaussian(variable, 300, 0.8, 100%)	Solarize Add(-002, 250, 100%)	RandomMultiAug(3, 15, 0)
Patch Gaussian(variable, 300, 1.0, 100%)	Solarize Add(-027, 000, 100%)	RandomMultiAug(3, 15, 1)
Patch Gaussian(variable, 300, 2.0, 100%)	Solarize Add(-027, 050, 100%)	RandomMultiAug(3, 20, 0)
Patch Gaussian(variable, 350, 0.2, 100%)	Solarize Add(-027, 100, 100%)	RandomMultiAug(3, 20, 1)
Patch Gaussian(variable, 350, 0.5, 100%)	Solarize Add(-027, 150, 100%)	RandomMultiAug(4, 1, 0)
Patch Gaussian(variable, 350, 0.8, 100%)	Solarize Add(-027, 200, 100%)	RandomMultiAug(4, 1, 1)
Patch Gaussian(variable, 350, 1.0, 100%)	Solarize Add(-027, 250, 100%)	RandomMultiAug(4, 3, 0)
Patch Gaussian(variable, 350, 2.0, 100%)	Solarize Add(-052, 000, 100%)	RandomMultiAug(4, 3, 1)
Patch Gaussian(variable, 400, 0.2, 100%)	Solarize Add(-052, 050, 100%)	RandomMultiAug(4, 7, 0)
Patch Gaussian(variable, 400, 0.5, 100%)	Solarize Add(-052, 100, 100%)	RandomMultiAug(4, 7, 1)
Patch Gaussian(variable, 400, 0.8, 100%)	Solarize Add(-052, 150, 100%)	RandomMultiAug(4, 10, 0)
Patch Gaussian(variable, 400, 1.0, 100%)	Solarize Add(-052, 200, 100%)	RandomMultiAug(4, 10, 1)
Patch Gaussian(variable, 400, 2.0, 100%)	Solarize Add(-052, 250, 100%)	RandomMultiAug(4, 15, 0)
Patch Gaussian(fixed, 100, 0.0, 100%)	Solarize Add(-077, 000, 100%)	RandomMultiAug(4, 15, 1)
Patch Gaussian(fixed, 100, 0.2, 100%)	Solarize Add(-077, 050, 100%)	RandomMultiAug(4, 20, 0)
Patch Gaussian(fixed, 100, 0.5, 100%)	Solarize Add(-077, 100, 100%)	RandomMultiAug(4, 20, 1)
Patch Gaussian(fixed, 100, 0.8, 100%)	Solarize Add(-077, 150, 100%)	RandomMultiAug(5, 1, 0)
Patch Gaussian(fixed, 100, 1.0, 100%)	Solarize Add(-077, 200, 100%)	RandomMultiAug(5, 1, 1)
Patch Gaussian(fixed, 100, 2.0, 100%)	Solarize Add(-077, 250, 100%)	RandomMultiAug(5, 3, 0)
Patch Gaussian(fixed, 150, 0.2, 100%)	Solarize Add(-102, 000, 100%)	RandomMultiAug(5, 3, 1)
Patch Gaussian(fixed, 150, 0.5, 100%)	Solarize Add(-102, 050, 100%)	RandomMultiAug(5, 7, 0)
Patch Gaussian(fixed, 150, 0.8, 100%)	Solarize Add(-102, 100, 100%)	RandomMultiAug(5, 7, 1)
Patch Gaussian(fixed, 150, 1.0, 100%)	Solarize Add(-102, 150, 100%)	RandomMultiAug(5, 10, 0)
Patch Gaussian(fixed, 150, 2.0, 100%)	Solarize Add(-102, 200, 100%)	RandomMultiAug(5, 15, 0)
Patch Gaussian(fixed, 200, 0.2, 100%)	Solarize Add(-102, 250, 100%)	RandomMultiAug(5, 15, 1)
Patch Gaussian(fixed, 200, 0.5, 100%)	Solarize Add(-127, 000, 100%)	RandomMultiAug(5, 20, 0)
Patch Gaussian(fixed, 200, 0.8, 100%)	Solarize Add(-127, 050, 100%)	RandomMultiAug(5, 20, 1)

Each augmentation was applied with a certain probability (given as a percentage in the label). Each time an image was pulled for training, the given image was augmented with that probability.

## C ERROR ANALYSIS

All of the CIFAR-10 experiments were repeated with 10 different initialization. In most cases, the resulting standard error on the mean (SEM) is too small to show as error bars on plots. The error on each measurement is given in the full results (see Sec. [removed for anonymization]).

Affinity and Switch-off Lift both were computed from differences between runs that share the same initialization. For Affinity, the same trained model was used for inference on clean validation data and on augmented validation data. Thus, the variance of Affinity for the clean baseline is not independent of the variance of Affinity for a given augmentation. The difference between the augmentation case and the clean baseline case was taken on a per-experiment basis (for each initialization of the clean baseline model) before the error was computed.

In the switching experiments, the final training *without* augmentation was completed starting from a given checkpoint in the model that was trained *with* augmentation. Thus, each switching experiment shared an initialization with an experiment that had no switching. Again, in this case the difference was taken on a per-experiment basis before the error (based on the standard deviation) was computed.

All ImageNet experiments shown are with one initialization. Thus, there are not statistics from which to analyze the error.

## D SWITCHING OFF AUGMENTATIONS

For CIFAR-10, switching times were tested in increments of approximately 5k steps between  $\sim 25k$  and  $\sim 75k$  steps. The best point for switching was determined by the final validation accuracy.

On ImageNet, we tested turning augmentation off at 50, 60, 70, and 80 epochs. Total training took 90 epochs. The best point for switching was determined by the final test accuracy.

The Switch-off Lift was derived from the experiment at the best switch-off point for each augmentation.

For CIFAR-10, there are some augmentations where the validation accuracy was best at 25k, which means that further testing is needed to find if the actual optimum switch off point is lower or if the best case is to not train at all with the given augmentation. Some of the best augmentations have a small negative Switch-off Lift, indicating that it is better to train the entire time with the given augmentations. For completeness we have listed the constant and optimal switching performance for all augmentations studied in the table below. The error is the standard error on the mean of the difference between the two over ten runs.

Aug name long	Constant Aug	Turn Aug Off	Switching Lift Err
Clean	0.90		
Blur(100%)	0.54	0.89	1.1e-02
Blur(50%)	0.89	0.90	1.6e-03
Crop(4, 25%)	0.93	0.93	2.7e-04
Crop(4, 50%)	0.93	0.93	2.5e-04
Crop(4, 75%)	0.93	0.93	3.1e-04
Crop(4,100%)	0.93	0.93	3.3e-04
Cutout(16, 100%)	0.92	0.92	5.1e-04
Cutout(16, 25%)	0.92	0.92	2.5e-04
Cutout(16, 50%)	0.92	0.92	6.8e-04
Cutout(16, 75%)	0.92	0.92	6.5e-04
Equalize(100%)	0.81	0.90	4.8e-03
Equalize(50%)	0.89	0.90	1.5e-03
FlipLR(100%)	0.89	0.92	1.4e-03
FlipLR(25%)	0.93	0.93	3.0e-04
FlipLR(50%)	0.93	0.93	1.9e-04
FlipLR(50%) + Crop(4, 100%) + Cutout(16, 100%)...	0.95	0.95	4.0e-04
FlipLR(50%) + Crop(4, 100%) + Cutout(16, 100%)...	0.95	0.95	4.4e-04
FlipLR(50%) + Crop(4,100%)	0.95	0.95	4.8e-04
FlipLR(75%)	0.93	0.93	2.0e-04
FlipUD(100%)	0.40	0.90	2.1e-03

FlipUD(25%)	0.90	0.91	6.5e-04
FlipUD(50%)	0.90	0.91	7.3e-04
FlipUD(75%)	0.89	0.91	1.0e-03
Invert(100%)	0.58	0.90	4.3e-03
Invert(50%)	0.88	0.90	2.0e-03
Rotate(fixed, 15deg, 50%)	0.93	0.93	2.9e-04
Rotate(fixed, 20deg, 100%)	0.85	0.92	2.1e-03
Rotate(fixed, 20deg, 50%)	0.93	0.93	3.0e-04
Rotate(fixed, 20deg, 75%)	0.92	0.93	3.1e-04
Rotate(fixed, 45deg, 50%)	0.92	0.92	3.7e-04
Rotate(fixed, 5deg, 75%)	0.92	0.92	1.6e-04
Rotate(square, 100%)	0.90	0.92	9.5e-04
Rotate(square, 50%)	0.91	0.92	7.5e-04
Rotate(variable, 20deg, 100%)	0.93	0.93	3.2e-04
Rotate(variable, 20deg, 50%)	0.93	0.93	4.8e-04
Rotate(variable, 20deg, 75%)	0.93	0.93	2.5e-04
Rotate(variable, 45, 100%)	0.92	0.93	1.7e-03
Rotate(variable, 5deg, 100%)	0.92	0.92	2.0e-04
Rotate(variable, 5deg, 50%)	0.92	0.92	2.7e-04
Rotate(variable, 5deg, 75%)	0.92	0.92	1.6e-04
Rotate(variable, 60deg, 100%)	0.92	0.93	5.3e-04
ShearX(fixed, 0.1, 100%)	0.91	0.92	8.0e-04
ShearX(fixed, 0.1, 50%)	0.92	0.92	3.2e-04
ShearX(fixed, 0.1, 75%)	0.92	0.92	3.4e-04
ShearX(fixed, 0.3, 100%)	0.90	0.91	8.1e-04
ShearX(fixed, 0.3, 50%)	0.92	0.92	4.2e-04
ShearX(fixed, 0.3, 75%)	0.92	0.92	2.6e-04
ShearX(variable, 0.1, 100%)	0.92	0.92	3.2e-04
ShearX(variable, 0.1, 50%)	0.92	0.92	3.0e-04
ShearX(variable, 0.1, 75%)	0.92	0.92	2.9e-04
ShearX(variable, 0.3, 100%)	0.92	0.92	3.6e-04
ShearX(variable, 0.3, 50%)	0.92	0.92	2.4e-04
ShearX(variable, 0.3, 75%)	0.92	0.92	3.4e-04
FlipLR(25%) + Crop(4,25%) + Cutout(16,25%)	0.95	0.95	2.3e-04
FlipLR(25%) + Crop(4,25%) + Cutout(16,50%)	0.95	0.95	3.7e-04
FlipLR(25%) + Crop(4,25%) + Cutout(16,75%)	0.95	0.95	3.5e-04
FlipLR(25%) + Crop(4,25%) + Cutout(16,100%)	0.95	0.95	4.6e-04
FlipLR(25%) + Crop(4,50%) + Cutout(16,25%)	0.95	0.95	3.7e-04
FlipLR(25%) + Crop(4,50%) + Cutout(16,50%)	0.95	0.95	4.7e-04
FlipLR(25%) + Crop(4,50%) + Cutout(16,75%)	0.95	0.95	3.6e-04
FlipLR(25%) + Crop(4,50%) + Cutout(16,100%)	0.95	0.95	3.0e-04
FlipLR(25%) + Crop(4,75%) + Cutout(16,25%)	0.95	0.95	2.4e-04
FlipLR(25%) + Crop(4,75%) + Cutout(16,50%)	0.95	0.95	3.6e-04
FlipLR(25%) + Crop(4,75%) + Cutout(16,75%)	0.95	0.95	3.3e-04
FlipLR(25%) + Crop(4,75%) + Cutout(16,100%)	0.95	0.95	5.4e-04
FlipLR(25%) + Crop(4,100%) + Cutout(16,25%)	0.95	0.95	3.2e-04
FlipLR(25%) + Crop(4,100%) + Cutout(16,50%)	0.95	0.95	3.3e-04
FlipLR(25%) + Crop(4,100%) + Cutout(16,75%)	0.95	0.95	2.5e-04
FlipLR(25%) + Crop(4,100%) + Cutout(16,100%)	0.95	0.95	3.4e-04
FlipLR(50%) + Crop(4,25%) + Cutout(16,25%)	0.95	0.95	2.3e-04
FlipLR(50%) + Crop(4,25%) + Cutout(16,50%)	0.95	0.95	2.2e-04
FlipLR(50%) + Crop(4,25%) + Cutout(16,75%)	0.95	0.95	4.7e-04
FlipLR(50%) + Crop(4,25%) + Cutout(16,100%)	0.95	0.95	3.5e-04
FlipLR(50%) + Crop(4,50%) + Cutout(16,25%)	0.95	0.95	3.1e-04
FlipLR(50%) + Crop(4,50%) + Cutout(16,50%)	0.95	0.95	3.8e-04
FlipLR(50%) + Crop(4,50%) + Cutout(16,75%)	0.95	0.95	3.7e-04
FlipLR(50%) + Crop(4,50%) + Cutout(16,100%)	0.95	0.95	5.6e-04
FlipLR(50%) + Crop(4,75%) + Cutout(16,25%)	0.95	0.95	3.2e-04
FlipLR(50%) + Crop(4,75%) + Cutout(16,50%)	0.95	0.95	4.3e-04
FlipLR(50%) + Crop(4,75%) + Cutout(16,75%)	0.96	0.95	6.1e-04
FlipLR(50%) + Crop(4,75%) + Cutout(16,100%)	0.95	0.95	4.2e-04
FlipLR(50%) + Crop(4,100%) + Cutout(16,25%)	0.95	0.95	3.7e-04
FlipLR(50%) + Crop(4,100%) + Cutout(16,50%)	0.95	0.96	3.6e-04
FlipLR(50%) + Crop(4,100%) + Cutout(16,75%)	0.95	0.95	2.8e-04
FlipLR(50%) + Crop(4,100%) + Cutout(16,100%)	0.96	0.96	5.0e-04

FlipLR(75%) + Crop(4,25%) + Cutout(16,25%)	0.95	0.95	2.7e-04
FlipLR(75%) + Crop(4,25%) + Cutout(16,50%)	0.95	0.95	5.0e-04
FlipLR(75%) + Crop(4,25%) + Cutout(16,75%)	0.95	0.95	3.9e-04
FlipLR(75%) + Crop(4,25%) + Cutout(16,100%)	0.95	0.95	3.3e-04
FlipLR(75%) + Crop(4,50%) + Cutout(16,25%)	0.95	0.95	2.6e-04
FlipLR(75%) + Crop(4,50%) + Cutout(16,50%)	0.95	0.95	3.7e-04
FlipLR(75%) + Crop(4,50%) + Cutout(16,75%)	0.95	0.95	3.4e-04
FlipLR(75%) + Crop(4,50%) + Cutout(16,100%)	0.95	0.95	3.5e-04
FlipLR(75%) + Crop(4,75%) + Cutout(16,25%)	0.95	0.95	3.6e-04
FlipLR(75%) + Crop(4,75%) + Cutout(16,50%)	0.95	0.95	3.6e-04
FlipLR(75%) + Crop(4,75%) + Cutout(16,75%)	0.95	0.95	3.4e-04
FlipLR(75%) + Crop(4,75%) + Cutout(16,100%)	0.95	0.95	3.0e-04
FlipLR(75%) + Crop(4,100%) + Cutout(16,25%)	0.95	0.95	4.3e-04
FlipLR(75%) + Crop(4,100%) + Cutout(16,50%)	0.95	0.95	2.7e-04
FlipLR(75%) + Crop(4,100%) + Cutout(16,75%)	0.95	0.95	4.5e-04
FlipLR(75%) + Crop(4,100%) + Cutout(16,100%)	0.95	0.95	3.5e-04
FlipLR(100%) + Crop(4,25%) + Cutout(16,25%)	0.93	0.94	7.9e-04
FlipLR(100%) + Crop(4,25%) + Cutout(16,50%)	0.94	0.94	4.7e-04
FlipLR(100%) + Crop(4,25%) + Cutout(16,75%)	0.94	0.94	7.1e-04
FlipLR(100%) + Crop(4,25%) + Cutout(16,100%)	0.94	0.94	5.3e-04
FlipLR(100%) + Crop(4,50%) + Cutout(16,25%)	0.94	0.94	5.8e-04
FlipLR(100%) + Crop(4,50%) + Cutout(16,50%)	0.94	0.94	5.2e-04
FlipLR(100%) + Crop(4,50%) + Cutout(16,75%)	0.94	0.94	5.0e-04
FlipLR(100%) + Crop(4,50%) + Cutout(16,100%)	0.94	0.94	5.8e-04
FlipLR(100%) + Crop(4,75%) + Cutout(16,25%)	0.94	0.94	5.0e-04
FlipLR(100%) + Crop(4,75%) + Cutout(16,50%)	0.94	0.94	7.5e-04
FlipLR(100%) + Crop(4,75%) + Cutout(16,75%)	0.94	0.95	4.4e-04
FlipLR(100%) + Crop(4,75%) + Cutout(16,100%)	0.94	0.95	6.6e-04
FlipLR(100%) + Crop(4,100%) + Cutout(16,25%)	0.94	0.94	5.3e-04
FlipLR(100%) + Crop(4,100%) + Cutout(16,50%)	0.94	0.94	4.7e-04
FlipLR(100%) + Crop(4,100%) + Cutout(16,75%)	0.94	0.95	6.5e-04
FlipLR(100%) + Crop(4,100%) + Cutout(16,100%)	0.94	0.95	4.9e-04
Rotate(fixed, 5deg, 10%)	0.92	0.92	1.9e-04
Rotate(fixed, 5deg, 20%)	0.92	0.92	3.6e-04
Rotate(fixed, 5deg, 30%)	0.92	0.92	3.1e-04
Rotate(fixed, 5deg, 40%)	0.92	0.92	1.9e-04
Rotate(fixed, 5deg, 50%)	0.92	0.92	3.0e-04
Rotate(fixed, 5deg, 60%)	0.92	0.92	2.6e-04
Rotate(fixed, 5deg, 70%)	0.92	0.92	2.9e-04
Rotate(fixed, 5deg, 80%)	0.92	0.92	2.3e-04
Rotate(fixed, 5deg, 90%)	0.92	0.92	3.4e-04
Rotate(fixed, 5deg, 100%)	0.92	0.92	4.5e-04
Rotate(fixed, 60deg, 10%)	0.91	0.91	4.3e-04
Rotate(fixed, 60deg, 20%)	0.91	0.91	6.0e-04
Rotate(fixed, 60deg, 30%)	0.91	0.92	4.7e-04
Rotate(fixed, 60deg, 40%)	0.91	0.92	5.0e-04
Rotate(fixed, 60deg, 50%)	0.92	0.92	1.9e-03
Rotate(fixed, 60deg, 60%)	0.91	0.91	5.3e-04
Rotate(fixed, 60deg, 70%)	0.91	0.92	1.4e-03
Rotate(fixed, 60deg, 80%)	0.91	0.92	1.9e-03
Rotate(fixed, 60deg, 90%)	0.90	0.91	1.5e-03
Rotate(fixed, 60deg, 100%)	0.41	0.91	3.9e-03
PatchGaussian(fixed,4, 0.1, 100%)	0.90	0.90	4.3e-04
PatchGaussian(fixed,4, 0.2, 100%)	0.90	0.90	3.5e-04
PatchGaussian(fixed,4, 0.3, 100%)	0.90	0.90	3.0e-04
PatchGaussian(fixed,4, 0.5, 100%)	0.91	0.91	2.6e-04
PatchGaussian(fixed,4, 0.8, 100%)	0.91	0.91	3.6e-04
PatchGaussian(fixed,4, 1.0, 100%)	0.91	0.91	2.8e-04
PatchGaussian(fixed,4, 1.5, 100%)	0.91	0.91	4.2e-04
PatchGaussian(fixed,4, 2.0, 100%)	0.91	0.91	4.4e-04
PatchGaussian(fixed,8, 0.1, 100%)	0.90	0.90	7.6e-04
PatchGaussian(fixed,8, 0.2, 100%)	0.91	0.91	2.8e-04
PatchGaussian(fixed,8, 0.3, 100%)	0.91	0.91	5.5e-04
PatchGaussian(fixed,8, 0.5, 100%)	0.91	0.91	3.0e-04
PatchGaussian(fixed,8, 0.8, 100%)	0.92	0.91	3.6e-04



PatchGaussian(fixed,8, 1.0, 100%)	0.92	0.91	4.6e-04
PatchGaussian(fixed,8, 1.5, 100%)	0.92	0.92	4.6e-04
PatchGaussian(fixed,8, 2.0, 100%)	0.92	0.92	5.8e-04
PatchGaussian(fixed,12, 0.1, 100%)	0.90	0.90	2.3e-04
PatchGaussian(fixed,12, 0.2, 100%)	0.91	0.91	4.3e-04
PatchGaussian(fixed,12, 0.3, 100%)	0.91	0.91	5.2e-04
PatchGaussian(fixed,12, 0.5, 100%)	0.91	0.91	4.4e-04
PatchGaussian(fixed,12, 0.8, 100%)	0.91	0.91	6.7e-04
PatchGaussian(fixed,12, 1.0, 100%)	0.92	0.92	4.9e-04
PatchGaussian(fixed,12, 1.5, 100%)	0.92	0.91	4.0e-04
PatchGaussian(fixed,12, 2.0, 100%)	0.92	0.92	7.3e-04
PatchGaussian(fixed,16, 0.1, 100%)	0.90	0.90	5.1e-04
PatchGaussian(fixed,16, 0.2, 100%)	0.91	0.90	6.1e-04
PatchGaussian(fixed,16, 0.3, 100%)	0.91	0.90	7.4e-04
PatchGaussian(fixed,16, 0.5, 100%)	0.91	0.90	7.7e-04
PatchGaussian(fixed,16, 0.8, 100%)	0.90	0.91	1.4e-03
PatchGaussian(fixed,16, 1.0, 100%)	0.90	0.91	1.3e-03
PatchGaussian(fixed,16, 1.5, 100%)	0.90	0.90	7.5e-04
PatchGaussian(fixed,16, 2.0, 100%)	0.90	0.91	1.6e-03
PatchGaussian(fixed,20, 0.1, 100%)	0.90	0.90	1.3e-03
PatchGaussian(fixed,20, 0.2, 100%)	0.89	0.90	1.1e-03
PatchGaussian(fixed,20, 0.3, 100%)	0.89	0.90	1.6e-03
PatchGaussian(fixed,20, 0.5, 100%)	0.89	0.90	1.6e-03
PatchGaussian(fixed,20, 0.8, 100%)	0.88	0.90	1.8e-03
PatchGaussian(fixed,20, 1.0, 100%)	0.88	0.91	1.1e-03
PatchGaussian(fixed,20, 1.5, 100%)	0.87	0.90	1.8e-03
PatchGaussian(fixed,20, 2.0, 100%)	0.88	0.90	1.4e-03
PatchGaussian(fixed,24, 0.1, 100%)	0.89	0.90	9.7e-04
PatchGaussian(fixed,24, 0.2, 100%)	0.88	0.90	1.4e-03
PatchGaussian(fixed,24, 0.3, 100%)	0.87	0.90	1.3e-03
PatchGaussian(fixed,24, 0.5, 100%)	0.86	0.90	2.0e-03
PatchGaussian(fixed,24, 0.8, 100%)	0.84	0.90	1.6e-03
PatchGaussian(fixed,24, 1.0, 100%)	0.84	0.90	1.7e-03
PatchGaussian(fixed,24, 1.5, 100%)	0.83	0.90	2.4e-03
PatchGaussian(fixed,24, 2.0, 100%)	0.82	0.90	9.6e-04
PatchGaussian(fixed,28, 0.1, 100%)	0.88	0.90	1.0e-03
PatchGaussian(fixed,28, 0.2, 100%)	0.85	0.90	1.4e-03
PatchGaussian(fixed,28, 0.3, 100%)	0.83	0.90	1.8e-03
PatchGaussian(fixed,28, 0.5, 100%)	0.81	0.90	1.7e-03
PatchGaussian(fixed,28, 0.8, 100%)	0.79	0.90	2.4e-03
PatchGaussian(fixed,28, 1.0, 100%)	0.77	0.89	1.7e-03
PatchGaussian(fixed,28, 1.5, 100%)	0.74	0.89	2.4e-03
PatchGaussian(fixed,28, 2.0, 100%)	0.72	0.90	3.3e-03
PatchGaussian(fixed,32, 0.1, 100%)	0.88	0.90	1.8e-03
PatchGaussian(fixed,32, 0.2, 100%)	0.85	0.90	1.6e-03
PatchGaussian(fixed,32, 0.3, 100%)	0.82	0.90	1.8e-03
PatchGaussian(fixed,32, 0.5, 100%)	0.79	0.90	1.8e-03
PatchGaussian(fixed,32, 0.8, 100%)	0.79	0.89	2.5e-03
PatchGaussian(fixed,32, 1.0, 100%)	0.79	0.89	2.0e-03
PatchGaussian(fixed,32, 1.5, 100%)	0.81	0.89	1.5e-03
PatchGaussian(fixed,32, 2.0, 100%)	0.81	0.89	1.7e-03

## E DIVERSITY METRICS

We computed three possible diversity metrics, shown in Fig. 9: Entropy, Final Training Loss, Training Steps to Accuracy Threshold. The entropy was calculated only for augmentations that have a discrete stochasticity (such as `Rotate(fixed)`) and not for augmentations that have a continuous variation (such as `Rotate(variable)` or `PatchGaussian`). Final Training Loss is the batch statistic at the last step of training. For CIFAR-10 experiments, this was averaged across the 10 initializations. For ImageNet, it was averaged over the last 10 steps of training. Training Steps to Accuracy Threshold is the number of training steps at which the training accuracy first hits a threshold of 97%. A few of the tested augmentation (extreme versions of `PatchGaussian`) did not reach this threshold in the given time and that column is left blank in the full results.

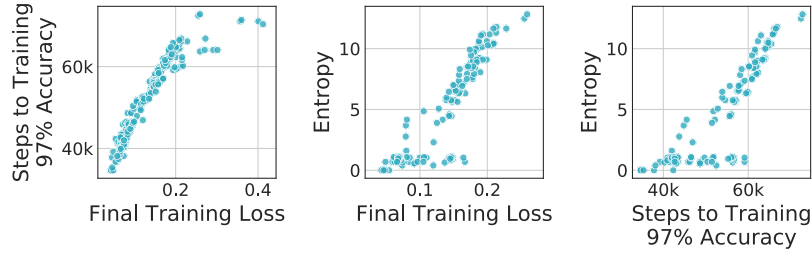


Figure 9: CIFAR-10: Three different diversity metrics are strongly correlated for high entropy augmentations. Here, the entropy is calculated only for discrete augmentations.

Entropy is unique in that it is independent of the model or dataset and it is a counting of states. However, it is difficult to compare between discrete and continuously-varying transforms and it is not clear how proper it is to compare even across different types of transforms.

Final Training Loss and Training Steps to Accuracy Threshold correlate well across the tested transforms. Entropy is highly correlated to these measures for PatchGaussian and versions of FlipLR, Crop, and Cutout where only probabilities are varying. For Rotate and Shear where magnitudes are varying as well, the correlation between Entropy and the other two measures is less clear.

One shortcoming of the loss or training step based Diversity measures is that they require fully training a new model for each augmentation policy. To ameliorate this computational cost, we also consider using the training loss, but for models trained for a significantly shorter time. In Figure 10 and Figure 11 We consider CIFAR-10 models trained for 4.5% and ImageNet models trained for 12% of the full training time. We find good correlation between this short time measure and the training loss computed at the end of training. With this alternate computationally less-expensive measure we find that 98.9% of pairs of CIFAR-10 models and 9.66% of ImageNet models satisfy Inequality 3

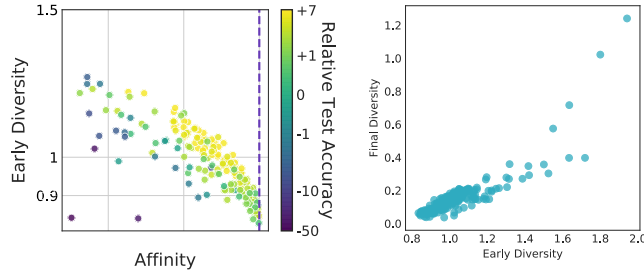


Figure 10: CIFAR-10: (Left) Diversity as the average loss over the first 10 epochs (out of 222) epochs of training (10 measurements, one per epoch). This represents an 96.5% reduction in computational cost. 98.9% of pairs satisfy Inequality 3 (Right) Early Diversity compared to Diversity computed for full training, this represents a Spearman correlation of 0.87

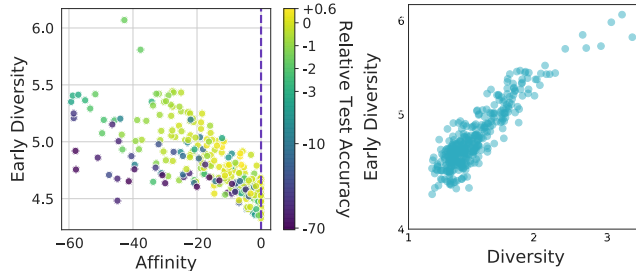


Figure 11: ImageNet: (Left) Diversity as the average loss over the first 13000 (out of 112000) steps of training (5 measurements, one per epoch), this represents an 88% reduction in computational cost. 96.6% of pairs satisfy Inequality 3 (Right) Early Diversity compared to Diversity computed for full training. This represents a Spearman correlation of 0.82.

## F COMPARING AFFINITY TO OTHER RELATED MEASURES

We gain confidence in the Affinity measure by comparing it to other potential model-dependant measures of distribution shift. In Fig 12, we show the correlation between Affinity and these two measures: the mean log likelihood of augmented test images (Grathwohl et al., 2019) (labeled as “logsumexp(logits)”) and the Watanabe–Akaike information criterion (labeled as “WAIC”) (Watanabe, 2010).

Like Affinity, these other two measures indicate how well a model trained on clean data comprehends augmented data.

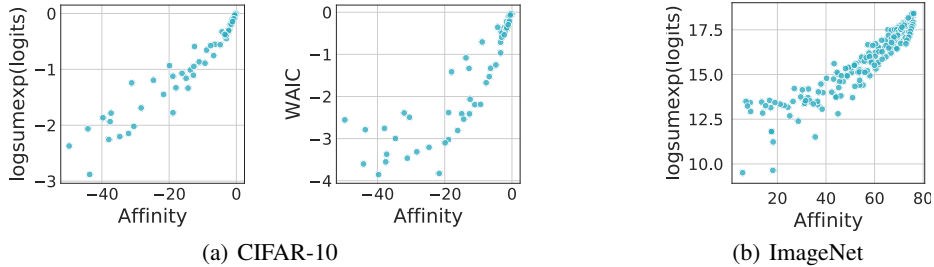


Figure 12: Affinity correlates with two other measures of how augmented images are related to a trained model’s distribution:  $\text{logsumexp}$  of the logits (left, for CIFAR-10, and right, for ImageNet) is the mean log likelihood for the image. WAIC (middle, for CIFAR-10) corrects for a possible bias in that estimate. In all three plots, numbers are referenced to the clean baseline, which is assigned a value of 0.

## G MODEL ROBUSTNESS ON THE AFFINITY-DIVERSITY PLANE

To shed light on whether Affinity and Diversity inform model robustness we evaluated our ImageNet and CIFAR-10 models on ImageNet-V2 and CIFAR-10.1 (Recht et al., 2019; 2018) as well as on CIFAR-10-C (Hendrycks & Dietterich, 2019). The former two datasets attempt to recreate the CIFAR-10 and ImageNet test sets as closely as possible, however model performance typically suffers on these sets indicating some degree of distribution shift. CIFAR-10-C is a dataset designed to test model robustness to a variety of image corruptions. We display accuracy for each individual corruption in Figure 16 and the overall CIFAR-10-C score in Figure 15. We find that for both ImageNet-V2 and CIFAR-10.1, and the robustness gap (difference between clean and robust accuracy) show consistent trends with higher diversity and higher affinity augmentations leading to more robust performance. For some of the CIFAR-10-C corruptions we see a similar trend. For other augmentations, (e.g. Gaussian noise, impulse noise, speckle and shot noise) we see that models trained with patch Gaussian augmentations (red-circles) outperform this general trend. This is perhaps consistent with the picture that models trained with augmentations well paired with a specific corruption perform well (Yin et al., 2019; Anonymous, 2021).

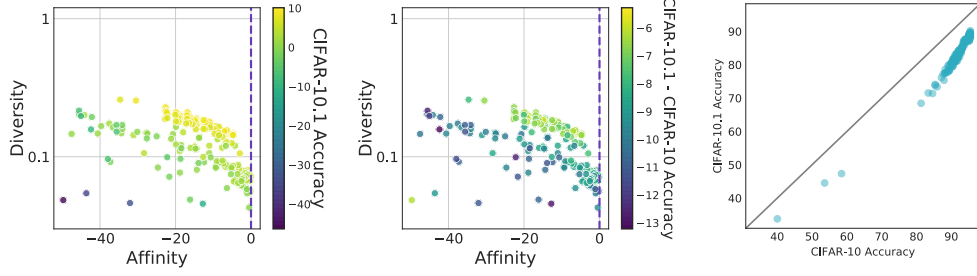


Figure 13: (Left) Augmented model performance on CIFAR-10.1 relative to a clean baseline (acc 80.1). We see that robust accuracy is higher for models trained with more diverse higher affinity augmentation policies. (Middle) Gap in performance between CIFAR-10.1 and CIFAR-10 accuracy. (Right) CIFAR-10.1 accuracy versus CIFAR-10 accuracy.

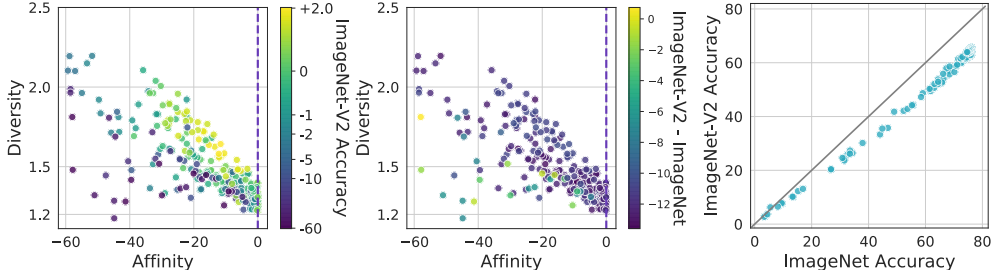


Figure 14: (Left) Augmented model performance on ImageNet-V2 relative to a clean baseline (acc 63.7). We see that robust accuracy is higher for models trained with more diverse higher affinity augmentation policies. (Middle) Gap in performance between ImageNet-V2 and ImageNet accuracy. The gap diminishes for models trained with high diversity, high affinity augmentations. (Right) ImageNet-V2 accuracy versus ImageNet accuracy.

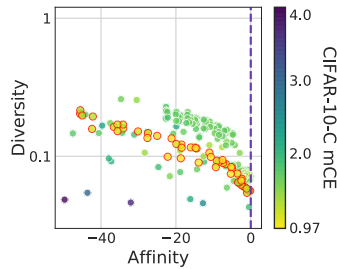


Figure 15: Mean corruption error on CIFAR-10-C (lower is better): Augmentations with high Affinity and high Diversity generally lead to lower mean corruption scores, and thus better robustness performance. Patch Gaussian augmentations (red circles), however, outperform this trend, potentially due to the similarity between some of the corruption transformations and Gaussian noise.

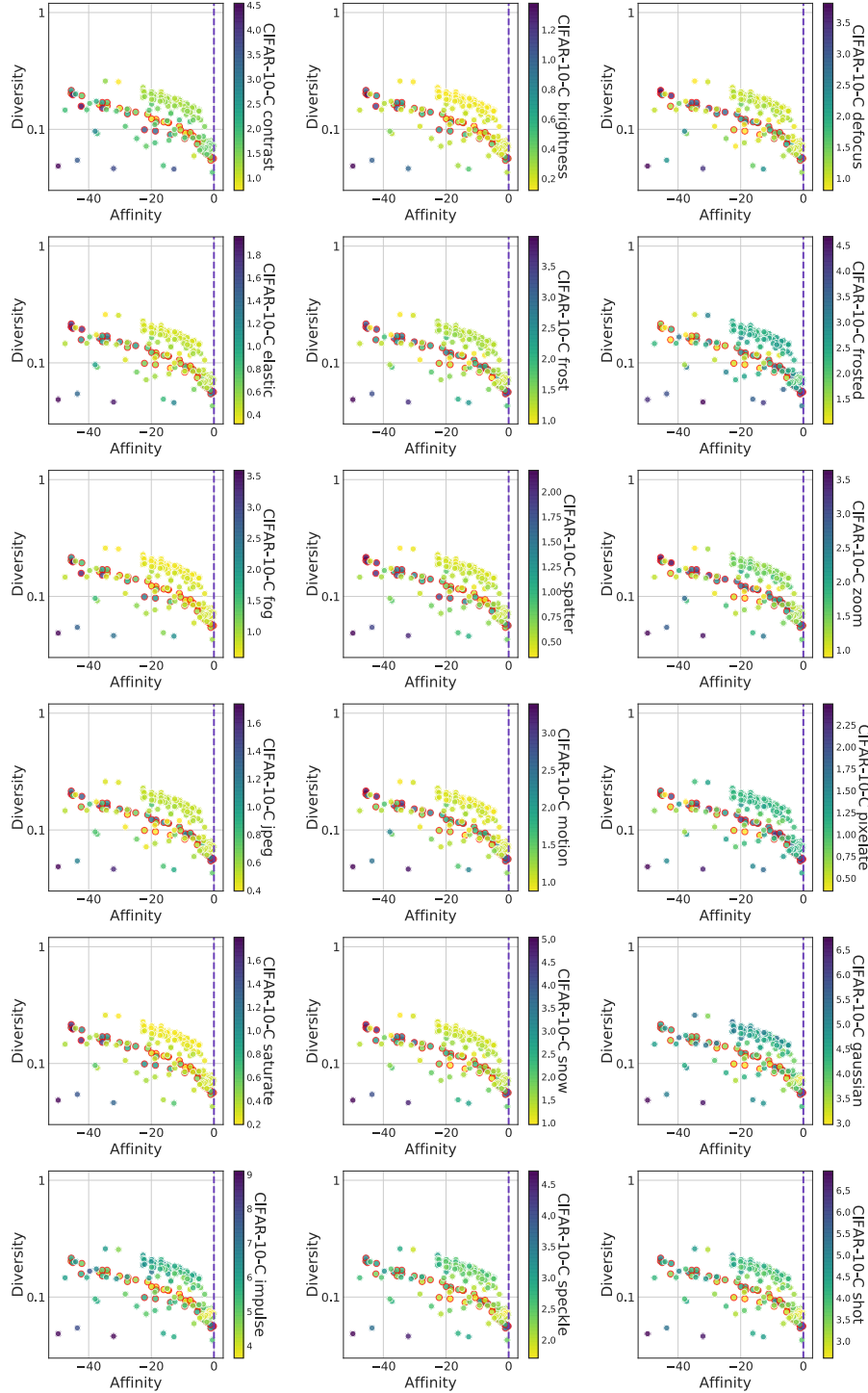


Figure 16: Robustness performance on CIFAR-10-C: Figures show corruption error for 18 different corruption types averaged over five corruption strengths and compared to the baseline model, AlexNet, used by [Hendrycks & Dietterich \(2019\)](#). Robust performance for some corruptions, such as brightness, is maximal for models trained with high Affinity high Diversity augmentations. For other corruptions, such as Gaussian noise or speckle, we do not see this. Red circles indicate models trained with patch Gaussian augmentation which are particularly robust to paired families of augmentations.

## H ADDITIONAL FIGURES FOR REVIEWERS

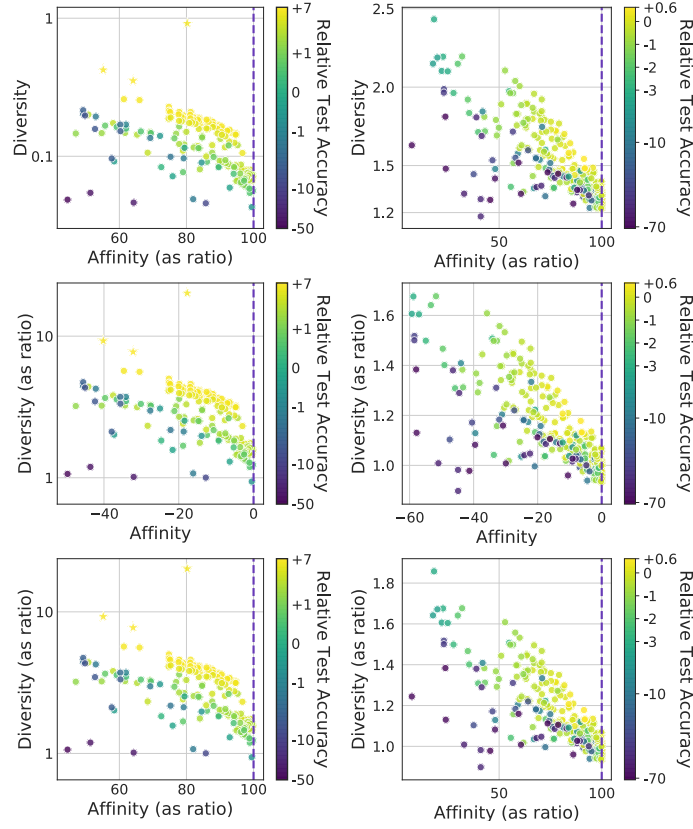


Figure 17: Here we present potential alternate versions of Figure 3 corresponding to different normalization conventions for Affinity and Diversity. Here, (as ratio) indicates that the augmentation Affinity and/or Diversity has been normalized by the Affinity or Diversity of the model trained without augmentation.

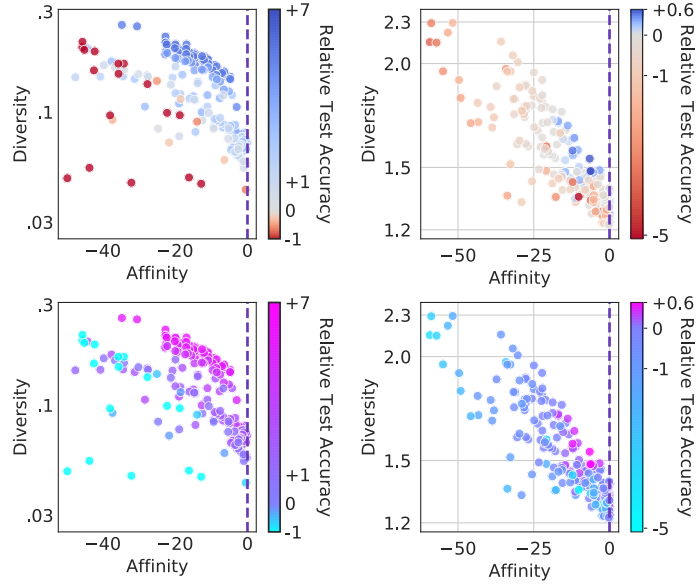


Figure 18: Versions of Fig. 1 with two alternative colorschemes. Top: *coolwarm* diverging colorscheme. Bottom: *cool*. Although not diverging, it is composed of two colors, which effectively highlights positive and negative performance of augmentations with respect to the baseline. CIFAR-10 data is shown in the left column and ImageNet on the right.

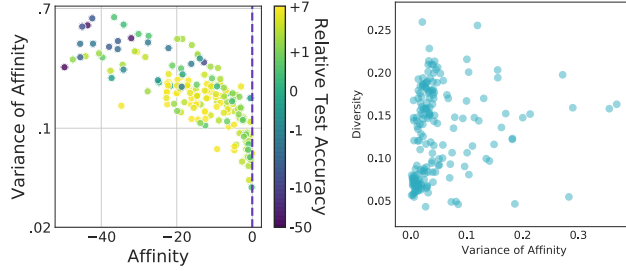


Figure 19: Replacing Diversity with Variance of Affinity, calculated over 5 model reinitializations on CIFAR-10. (Left) Relative test accuracy plotted against Affinity and Variance of Affinity. (Right) Variance of Affinity over model initializations does not correlate well with Diversity.

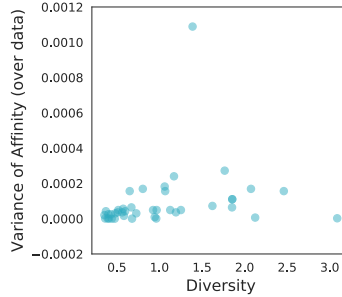


Figure 20: Replacing Diversity with Variance of Affinity, calculated over dataset batches on CIFAR-10. Variance of Affinity over dataset batches does not correlate well with Diversity.

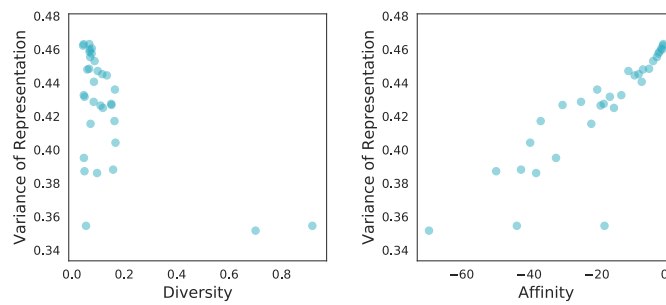


Figure 21: Replacing Diversity with Variance of Representation, calculated using the last BatchNorm output on CIFAR-10. Variance of Representation does not correlate well with Diversity. It does seem well correlated with Affinity.