# The Unreasonable Effectiveness of Fully-Connected Layers for Low-Data Regimes
# — Supplementary material —

**Peter Kocsis**
Technical University of Munich
peter.kocsis@tum.de

**Peter Súkeník**
Technical University of Munich
peter.sukenik@trojsten.sk

**Guillem Brasó**
Technical University of Munich
guillem.braso@tum.de

**Matthias Niessner**
Technical University of Munich
niessner@tum.de

**Laura Leal-Taixé**
Technical University of Munich
leal.taixe@tum.de

**Ismail Elezi**
Technical University of Munich
ismail.elezi@tum.de

peter-kocsis.github.io/LowDataGeneralization/

In this appendix, we start by providing more implementation details in Section A. In Subsection A.1, we detail the hyperparameters. In Subsection A.2, we give further information with regards to the used datasets. In Table 2, we show the number of extra parameters used during training with our method for all our experiments. In Subsection A.3, we provide the libraries used for the trainings of MLPMixer and ViT. In Subsection A.4, we provide more information for the Active Learning experiments.

In Section B, we do extra experiments with the VGG networks which already uses fully-connected layers. In Section C, we evaluate the robustness of our method on the CIFAR10-C dataset. Then, in section D, we evaluate our method on the Caltech101 dataset without any pretraining. Finally, in Section E, we provide the exact numbers (mean and standard deviation) for all the plots in the paper.

## A Training details

### A.1 Hyperparameters

Doing hyperparameter optimization in low-data regimes is extremely difficult. This is because in different cycles we have different datasets. Optimizing on them independently yields different optimal hyperparameters, which makes the results difficult to compare. Furthermore, that leads to several trainings for each cycle, which is costly. In addition, datasets like CIFAR and Caltech have public testing set, which means that an extensive hyperparameter optimization might lead to an overfitting of the testing set, and unreliable generalization results. To avoid all these issues, we decide to follow the work of LLAL [1] and use a single hyperparameter set across all the experiments and comparison. More details about the datasets and the used hyperparameters can be found in Table 1

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Table 1: **Summary of the used datasets and specific hyperparameters**. The first block until the dashed line describes the datasets. The second block shows the used data augmentations, followed by the training hyperparameters. During learning rate scheduling, we divide the learning rate by 10 after 80% epochs.

| | CIFAR10 | CIFAR100 | Caltech101 | Caltech256 |
|---|---|---|---|---|
| Number of classess | 10 | 100 | 102 | 257 |
| Number of training samples | 50000 | 50000 | 6117 | 21531 |
| Number of test samples | 10000 | 10000 | 2560 | 9076 |
| Official test split | ✓ | ✓ | ✗ | ✗ |
| Image size | 32x32 | 32x32 | Various | Various |
| Class balance | ✓ | ✓ | ✗ | ✗ |
| Random horizontal flip | ✓ | ✓ | ✓ | ✓ |
| Random crop | 32x32, p=4 | 32x32, p=4 | 224x224, p=16 | 224x224, p=16 |
| Normalization | ✓ | ✓ | ✓ | ✓ |
| Size of initial labeled pool | 1000 | 5000 | 1000 | 5000 |
| Samples labeled in stage | 1000 | 1000 | 1000 | 1000 |
| Number of stages | 10 | 10 | 6 | 10 |
| Feature size | 64 | 64 | 256 | 256 |
| Optimizer | SGD | SGD | SGD | SGD |
| Learning rate | 0.1 | 0.1 | 0.001 | 0.001 |
| Learning rate scheduler | ✓ | ✓ | ✓ | ✓ |
| Momentum | 0.9 | 0.9 | 0.9 | 0.9 |
| Weight decay | 5e-4 | 5e-4 | 5e-4 | 5e-4 |
| Number of epochs | 200 | 200 | 100 | 100 |

### A.2 Datasets and the number of labels.

For all supervised experiments, we use the same pre-defined datasplits. We obtained the datasplits with incremental random sampling. This ensures that the comparisons are fair, and subject to only the network architecture. Doing otherwise might result with some method being favored by having a better training split.

### A.3 MLPMixer [2] and ViT [3].

MLPMixer [2] and ViT [3] were originally developed for ImageNet. In order to achieve their best performance we compared multiple architectures specifically tailored for the CIFAR datasets, and chose the best performing ones, specifically, MLPMixer-Nano [4] and ViT-CIFAR10 [5]

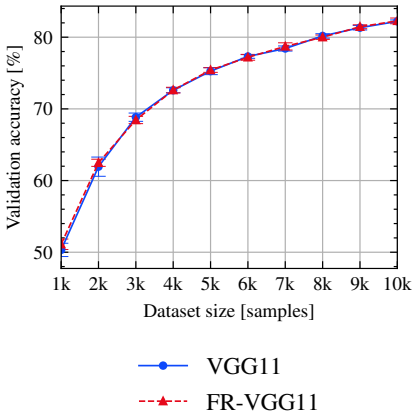
### A.4 Active Learning.

To achieve comparable results, we closely followed the active learning setup of LLAL [1]. We split the dataset into two parts: labeled and unlabaled pool. During training, we access only the labeled pool. In each cycle, we train a model on the labeled pool and use its predictions on the unlabeled pool to select samples to be included in the labeled pool for the next cycle. In each cycle, we reinitialize the model weights.

During the experiments, we noted that there is a difference between our and the officially reported LLAL [1] results. We found that the reason behind that is the mismatch between their and our PyTorch versions. They used PyTorch 1.1, while we use more modern PyTorch 1.7. Downgrading to the older version yields comparable results to the published ones. While we believe that reproducibility and comparability is crucial in research, using outdated libraries is not an ideal solution. Consequently, we compare their method and ours using PyTorch 1.7. We think that this is both fair, and allows an easier reproducibility for future research.

Table 2: **Extra parameters.** Number of extra parameters used during training for all experiments.

| | CIFAR | Caltech |
|---|---|---|
| ResNet18 | 11173962 | 11228325 |
| FR-ResNet18 | +42058 (0.38%) | +289893 (2.58%) |
| ResNet34 | 21282122 | ✗ |
| FR-ResNet34 | +42058 (0.20%) | ✗ |
| DenseNet121 | 6964096 | ✗ |
| FR-DenseNet121 | +74826 (1.07%) | ✗ |
| EfficientNetB3 | 10711602 | ✗ |
| FR-EfficientNetB3 | +82660 (0.77%) | ✗ |




(a) CIFAR10 plot

| #Samples | VGG11 | FR-VGG11 |
|---|---|---|
| 1000 | 50.33±0.93 | **51.06**±0.78 |
| 2000 | 61.94±1.34 | **62.43**±0.47 |
| 3000 | **68.84**±0.58 | 68.42±0.49 |
| 4000 | 72.60±0.35 | **72.62**±0.35 |
| 5000 | 75.27±0.49 | **75.40**±0.34 |
| 6000 | **77.31**±0.26 | 77.17±0.44 |
| 7000 | 78.42±0.36 | **78.66**±0.52 |
| 8000 | **80.18**±0.31 | 79.98±0.32 |
| 9000 | 81.33±0.32 | **81.46**±0.29 |
| 10000 | 82.21±0.20 | **82.22**±0.44 |

(b) CIFAR10 numerical results

Figure 1: **VGG experiment.** We evaluate our method with the VGG architecture. We can achieve highly similar performance with $67\%$ fewer parameters.

## B VGG experiment

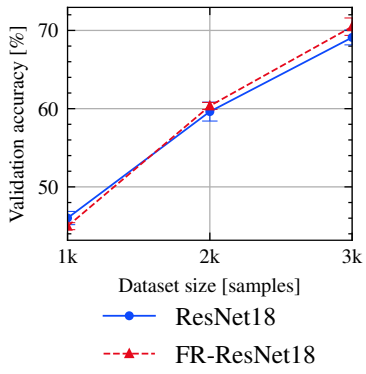
We conduct an experiment with an old-fashioned architecture that uses final fully-connected layers, the VGG11 [6]. Since this architecture follows our proposed solution, we are unable to improve upon its performance. However, we show that our light-weight Feature Refiner (FR) is able to achieve the same results, but using only $9,262,282$ parameters instead of $28,144,010$, resulting with a $67\%$ reduction in the number of weights. We present the results on Figure 1.

## C Robustness benchmark

We evaluate the robustness of ResNet18 and our model (FR-ResNet18) on the CIFAR10-C dataset [7]. CIFAR10-C contains a total of 95 perturbed test sets under 19 different corruptions with 5 severity levels. We present average results of 5 independent runs (Figure 5) over the different severity levels (Figure 4) and over the different corruption types (Figure 4). Our method consistently outperforms the baseline in cases of lower corruption severity. In cases of higher corruption severity (severity 3 and severity 4), in the very low-data regime, our method significantly outperforms the baseline. However, with more added data, the baseline starts outperforming our method.

## D Caltech without pre-training

Since the Caltech datasets are more complex, and also to show that our method works with pretrained models as well, we used Imagenet-pretrained backbone for the Caltech experiments in the main paper. Now, we conduct a minimal experiment to evaluate our method without pretraining. We run our method on the Caltech101 dataset without pretraining in the first three datasplits. As it can be seen in Figure 2, in the first cycle, we are worse than ResNet18 by $1pp$, similarly as with ImageNet

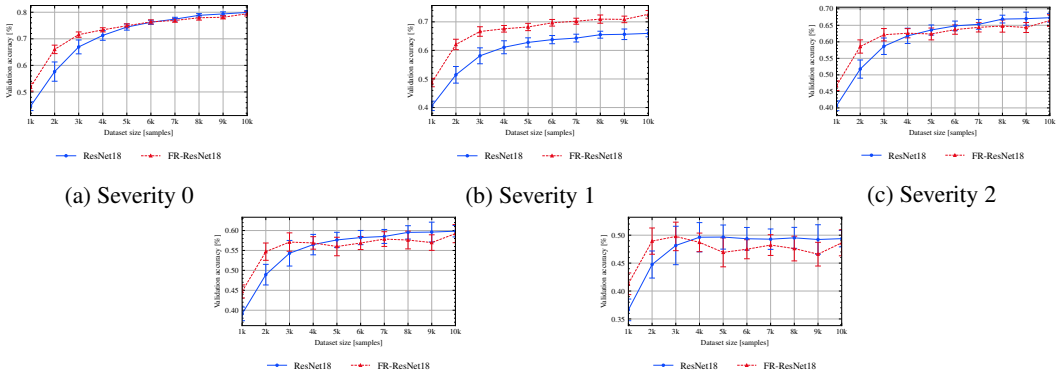



(a) Caltech101 plot

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | **46.02**±0.85 | 44.98±0.45 |
| 2000 | 59.62±1.21 | **60.35**±0.45 |
| 3000 | 69.10±0.95 | **70.48**±1.11 |

(b) Caltech101 numerical results

Figure 2: **Caltech101 w/o pretraining.** We evaluate our method on the first three splits of the Caltech101 dataset without Imagenet-pretraining. While both ResNet18 and our method reach significantly lower results than when we use ImageNet pretraining, the relative improvement of our method compared to ResNet18 remains.



(a) Severity 0 (b) Severity 1 (c) Severity 2

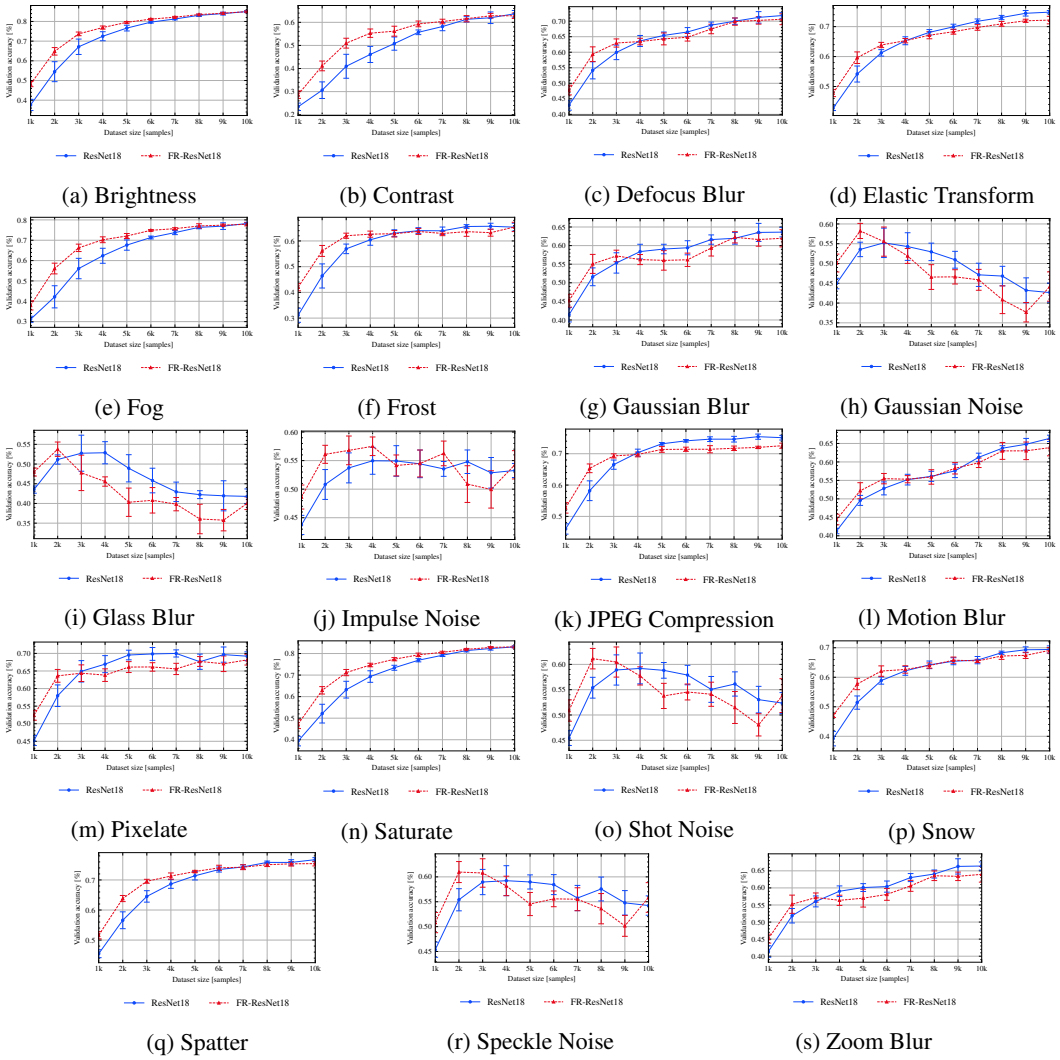
(d) Severity 3 (e) Severity 4

Figure 3: **Robustness benchmark - Severity.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score for all severity levels. Our method consistently outperforms the baseline in cases of lower corruption severity. In cases of higher corruption severity (3d and 3e), in the very low-data regime, our method significantly outperforms the baseline. However, with more added data, the baseline starts outperforming our method.

pretraining, where the difference was 0.9pp. In the second cycle, we outperform ResNet18 by $0.7pp$, while with ImageNet pretraining we outperformed them by $0.5pp$. In the third cycle, we outperform ResNet18 by $1.4pp$, while with ImageNet pretraining we outperformed them by $0.8pp$. In this way, we show that while both ResNet18 and our method reach significantly lower results than when we use ImageNet pretraining, the relative improvement of our method compared to ResNet18 remains.

## E Numerical results

In this section, we provide the exact numbers of all our results presented in the paper. Table 3, 4, 5, 6, 7, 8, 9 summarize the results provided in Figure 3, 4, 5, 6, 7, 8, 9. We provide the mean and standard deviation obtained by running the same experiment five times with different model initializations.




(a) Brightness (b) Contrast (c) Defocus Blur (d) Elastic Transform

(e) Fog (f) Frost (g) Gaussian Blur (h) Gaussian Noise

(i) Glass Blur (j) Impulse Noise (k) JPEG Compression (l) Motion Blur

(m) Pixelate (n) Saturate (o) Shot Noise (p) Snow

(q) Spatter (r) Speckle Noise (s) Zoom Blur

Figure 4: **Robustness benchmark - Corrpution types.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score for all corruption types. Our method consistently outperforms the baseline in the earlier iterations.

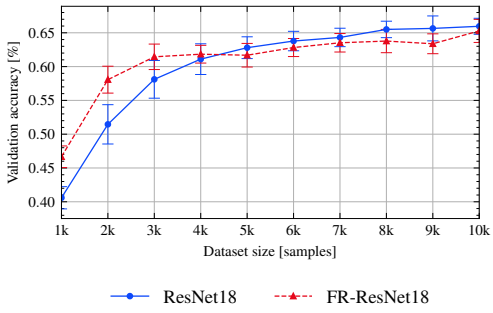



Figure 5: **Robustness benchmark - Overall.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score over all datasets. Our method performs better in the earlier iterations.

Table 3: **Comparisons with ResNet18 [8].** We compare our approach to our main baseline network, to ResNet18 [8] on the CIFAR10, CIFAR100, Caltech101 and Caltech256 datasets in supervised learning. Our method significantly outperforms the baseline.

(a) CIFAR10

| #Samples | ResNet18 [8] | FR-ResNet18 |
|---|---|---|
| 1000 | 46.18±2.81 | **53.70**±1.18 |
| 2000 | 60.36±1.64 | **70.46**±0.59 |
| 3000 | 69.22±3.21 | **76.81**±0.50 |
| 4000 | 75.45±2.29 | **79.66**±0.21 |
| 5000 | 80.23±0.85 | **82.27**±0.33 |
| 6000 | 82.52±0.59 | **83.65**±0.34 |
| 7000 | 84.08±0.38 | **84.48**±0.26 |
| 8000 | 85.51±0.51 | **85.88**±0.21 |
| 9000 | **86.59**±0.34 | 86.42±0.31 |
| 10000 | **87.31**±0.46 | 87.20±0.29 |

(b) CIFAR100

| #Samples | ResNet18 [8] | FR-ResNet18 |
|---|---|---|
| 5000 | 38.07±0.92 | **43.73**±1.36 |
| 6000 | 41.72±0.61 | **48.52**±0.78 |
| 7000 | 46.29±2.07 | **51.44**±0.45 |
| 8000 | 50.06±0.60 | **54.51**±0.40 |
| 9000 | 53.25±0.32 | **56.06**±0.40 |
| 10000 | 55.84±0.90 | **57.50**±0.56 |
| 11000 | 57.68±0.46 | **58.90**±0.09 |
| 12000 | 58.94±0.80 | **60.15**±0.33 |
| 13000 | 60.06±0.48 | **60.89**±0.55 |
| 14000 | 61.22±0.48 | **61.70**±0.56 |

(c) Caltech101

| #Samples | ResNet18 [8] | FR-ResNet18 |
|---|---|---|
| 1000 | **84.33**±0.70 | 83.46±0.53 |
| 2000 | 90.87±0.38 | **91.34**±0.17 |
| 3000 | 92.78±0.20 | **93.56**±0.35 |
| 4000 | 94.31±0.25 | **94.49**±0.17 |
| 5000 | 94.94±0.28 | **95.21**±0.20 |
| 6000 | 95.42±0.38 | **95.86**±0.21 |

(d) Caltech256

| #Samples | ResNet18 [8] | FR-ResNet18 |
|---|---|---|
| 5000 | 74.75±0.32 | **75.90**±0.20 |
| 6000 | 76.34±0.13 | **77.30**±0.15 |
| 7000 | 77.37±0.23 | **78.42**±0.26 |
| 8000 | 78.06±0.24 | **79.39**±0.25 |
| 9000 | 78.98±0.16 | **79.69**±0.22 |
| 10000 | 79.65±0.21 | **80.24**±0.09 |
| 11000 | 80.02±0.25 | **80.84**±0.15 |
| 12000 | 80.57±0.17 | **81.25**±0.28 |
| 13000 | 80.91±0.20 | **81.58**±0.12 |
| 14000 | 81.28±0.26 | **81.85**±0.13 |

Table 4: **Comparisons with MLPMixer [2] and ViT [3].** We compare our method to different modern architectures on the CIFAR10 and CIFAR100 datasets in supervised learning.

(a) CIFAR10

| #Samples | ResNet18 [8] | FR-ResNet18 | MLPMixer [2] | ViT-B16 [3] |
|---|---|---|---|---|
| 1000 | 46.18±2.81 | **53.70**±1.18 | 49.30±0.57 | 47.92±0.36 |
| 2000 | 60.36±1.64 | **70.46**±0.59 | 59.82±5.40 | 53.95±0.24 |
| 3000 | 69.22±3.21 | **76.81**±0.50 | 61.39±0.51 | 57.05±0.17 |
| 4000 | 75.45±2.29 | **79.66**±0.21 | 64.48±0.93 | 60.54±0.52 |
| 5000 | 80.23±0.85 | **82.27**±0.33 | 66.97±0.57 | 62.60±0.39 |
| 6000 | 82.52±0.59 | **83.65**±0.34 | 68.40±0.71 | 64.68±0.39 |
| 7000 | 84.08±0.38 | **84.48**±0.26 | 69.91±0.67 | 66.17±0.26 |
| 8000 | 85.51±0.51 | **85.88**±0.21 | 71.48±0.37 | 67.41±0.34 |
| 9000 | **86.59**±0.34 | 86.42±0.31 | 73.26±0.55 | 69.01±0.46 |
| 10000 | **87.31**±0.46 | 87.20±0.29 | 73.98±0.62 | 69.29±0.53 |

(b) CIFAR100

| #Samples | ResNet18 [8] | FR-ResNet18 | MLPMixer [2] | ViT-B16 [3] |
|---|---|---|---|---|
| 5000 | 38.07±0.92 | **43.73**±1.36 | 29.95±0.23 | 29.28±0.44 |
| 6000 | 41.72±0.61 | **48.52**±0.78 | 32.08±0.33 | 31.60±0.37 |
| 7000 | 46.29±2.07 | **51.44**±0.45 | 34.49±0.24 | 33.03±0.21 |
| 8000 | 50.06±0.60 | **54.51**±0.40 | 37.03±0.36 | 35.06±0.61 |
| 9000 | 53.25±0.32 | **56.06**±0.40 | 38.17±0.10 | 37.08±0.31 |
| 10000 | 55.84±0.90 | **57.50**±0.56 | 39.62±0.16 | 37.57±0.49 |
| 11000 | 57.68±0.46 | **58.90**±0.09 | 40.58±0.19 | 39.87±0.29 |
| 12000 | 58.94±0.80 | **60.15**±0.33 | 41.64±0.19 | 41.68±0.67 |
| 13000 | 60.06±0.48 | **60.89**±0.55 | 42.71±0.18 | 42.36±0.45 |
| 14000 | 61.22±0.48 | **61.70**±0.56 | 43.56±0.15 | 43.62±0.24 |

Table 5: **Comparisons with Knowledge Distillation baselines.** We compare to several KD methods. Our method outperforms the baselines in the earlier iterations with a large margin.

(a) CIFAR10

| #Samples | ResNet18 | FR-ResNet18 | DML | KDCL | KD-Student | KD-Teacher |
|---|---|---|---|---|---|---|
| 1000 | 46.18±2.81 | **53.70**±1.18 | 39.46±4.60 | 44.19±1.97 | 46.91±1.09 | 42.15±1.20 |
| 2000 | 60.36±1.64 | **70.46**±0.59 | 61.96±1.20 | 58.87±4.45 | 59.53±3.19 | 57.32±6.20 |
| 3000 | 69.22±3.21 | **76.81**±0.50 | 71.35±1.08 | 69.68±1.31 | 67.18±3.75 | 66.94±1.42 |
| 4000 | 75.45±2.29 | **79.66**±0.21 | 76.83±1.43 | 75.98±1.81 | 75.31±0.64 | 73.28±3.07 |
| 5000 | 80.23±0.85 | **82.27**±0.33 | 80.92±0.92 | 81.14±0.65 | 78.16±2.87 | 77.89±1.42 |
| 6000 | 82.52±0.59 | 83.65±0.34 | **83.67**±0.34 | 83.57±0.25 | 82.17±0.88 | 81.22±1.61 |
| 7000 | 84.08±0.38 | 84.48±0.26 | **84.98**±0.26 | 84.66±0.72 | 83.79±0.37 | 83.80±0.23 |
| 8000 | 85.51±0.51 | 85.88±0.21 | **86.28**±0.34 | 86.17±0.39 | 85.46±0.37 | 84.81±0.47 |
| 9000 | 86.59±0.34 | 86.42±0.31 | 86.74±0.53 | **86.98**±0.28 | 86.20±0.28 | 85.67±0.73 |
| 10000 | 87.31±0.46 | 87.20±0.29 | 87.52±0.37 | **87.55**±0.14 | 87.42±0.27 | 86.14±0.78 |

(b) CIFAR100

| #Samples | ResNet18 | FR-ResNet18 | DML | KDCL | KD-Student | KD-Teacher |
|---|---|---|---|---|---|---|
| 5000 | 38.07±0.92 | **43.73**±1.36 | 38.79±1.53 | 39.60±0.91 | 36.45±2.18 | 35.77±2.79 |
| 6000 | 41.72±0.61 | **48.52**±0.78 | 42.17±1.19 | 44.91±1.16 | 40.39±0.55 | 39.74±2.31 |
| 7000 | 46.29±2.07 | **51.44**±0.45 | 47.28±0.90 | 48.42±1.18 | 44.56±1.60 | 44.37±2.08 |
| 8000 | 50.06±0.60 | **54.51**±0.40 | 51.88±1.20 | 52.63±0.31 | 48.66±1.55 | 48.11±2.57 |
| 9000 | 53.25±0.32 | 56.06±0.40 | 55.13±1.27 | **56.25**±1.09 | 52.35±1.38 | 50.94±2.09 |
| 10000 | 55.84±0.90 | 57.50±0.56 | **58.08**±0.37 | 57.69±0.72 | 56.46±1.23 | 54.32±1.39 |
| 11000 | 57.68±0.46 | 58.90±0.09 | 58.88±0.81 | **60.30**±0.81 | 57.92±0.73 | 54.95±1.65 |
| 12000 | 58.94±0.80 | 60.15±0.33 | 60.30±1.06 | **61.81**±1.00 | 59.82±0.75 | 58.38±1.02 |
| 13000 | 60.06±0.48 | 60.89±0.55 | 62.58±0.55 | **63.39**±0.73 | 61.19±0.45 | 59.59±1.29 |
| 14000 | 61.22±0.48 | 61.70±0.56 | 63.55±0.55 | **64.56**±0.56 | 62.36±1.07 | 60.80±1.58 |

Table 6: **Comparison with SimSiam [4].** We compare our method against the CIFAR10 version of SimSiam [4]. Our method significantly outperforms SimSiam in the low-data regime.

| #Samples | ResNet18 | FR-ResNet18 | SimSiam-kNN | SimSiam-Linear |
|---|---|---|---|---|
| 1000 | 46.18±2.81 | **53.70**±1.18 | 25.09±1.10 | 36.67±0.46 |
| 2000 | 60.36±1.64 | **70.46**±0.59 | 36.53±0.74 | 45.01±0.11 |
| 3000 | 69.22±3.21 | **76.81**±0.50 | 48.12±0.33 | 54.58±0.08 |
| 4000 | 75.45±2.29 | **79.66**±0.21 | 55.69±0.52 | 61.39±0.50 |
| 5000 | 80.23±0.85 | **82.27**±0.33 | 60.96±0.16 | 65.55±0.40 |
| 6000 | 82.52±0.59 | **83.65**±0.34 | 65.83±0.12 | 68.22±0.57 |
| 7000 | 84.08±0.38 | **84.48**±0.26 | 69.45±0.15 | 70.53±0.35 |
| 8000 | 85.51±0.51 | **85.88**±0.21 | 72.34±0.26 | 73.75±0.11 |
| 9000 | **86.59**±0.34 | 86.42±0.31 | 74.99±0.25 | 76.57±0.17 |
| 10000 | **87.31**±0.46 | 87.20±0.29 | 76.85±0.56 | 79.20±0.15 |

Table 7: **Active Learning.** We evaluate the performance of our method on a typical low-data setiing, on active learning. We compare to standard maximum entropy-based acquisition and also to two state-of-the-art approaches, to core-set and LLAL [1]. Our method clearly outperforms all the other approaches, especially in the earlier stages.

(a) CIFAR10

| #Samples | ResNet18-Entropy | FR-ResNet18-Entropy | LLAL [1] | ResNet18-core-set |
|---|---|---|---|---|
| 1000 | 46.28±2.54 | **53.45**±1.30 | 46.53±1.57 | 46.52±2.64 |
| 2000 | 60.26±2.95 | **69.33**±2.13 | 60.96±3.10 | 62.04±2.57 |
| 3000 | 69.76±2.69 | **77.64**±1.00 | 73.03±2.68 | 69.06±2.07 |
| 4000 | 77.62±2.25 | **81.86**±0.62 | 80.83±0.83 | 77.33±2.25 |
| 5000 | 81.27±3.14 | 84.10±0.31 | **84.14**±0.56 | 80.04±1.80 |
| 6000 | 85.13±0.98 | **86.33**±0.40 | 85.95±0.50 | 84.04±0.84 |
| 7000 | 87.05±0.68 | **87.77**±0.36 | 87.40±0.49 | 86.24±0.57 |
| 8000 | 88.33±0.64 | **88.91**±0.29 | 88.55±0.59 | 87.56±0.49 |
| 9000 | **89.83**±0.60 | 89.59±0.23 | 89.18±0.41 | 88.67±0.55 |
| 10000 | **90.40**±0.33 | 90.36±0.23 | 89.78±0.42 | 89.86±0.45 |

(b) CIFAR100

| #Samples | ResNet18-Entropy | FR-ResNet18-Entropy | LLAL [1] | ResNet18-core-set |
|---|---|---|---|---|
| 5000 | 38.73±0.82 | **44.22**±1.09 | 37.36±0.92 | 38.60±0.95 |
| 6000 | 41.95±2.20 | **48.91**±0.66 | 43.73±0.76 | 43.62±0.45 |
| 7000 | 45.99±0.74 | **51.21**±0.79 | 48.26±0.62 | 48.10±0.86 |
| 8000 | 50.05±0.69 | **53.80**±0.58 | 51.62±0.60 | 50.26±1.61 |
| 9000 | 53.70±0.47 | **56.02**±0.63 | 54.20±0.43 | 54.57±1.44 |
| 10000 | 55.96±0.54 | **57.61**±0.61 | 56.18±0.83 | 56.35±1.41 |
| 11000 | 58.00±1.19 | **59.73**±0.53 | 57.91±0.57 | 58.80±0.44 |
| 12000 | 59.90±1.08 | **60.64**±0.49 | 59.34±0.74 | 60.43±0.86 |
| 13000 | 61.39±1.37 | **62.01**±0.40 | 60.78±0.93 | 61.67±0.79 |
| 14000 | **63.78**±0.49 | 63.27±0.47 | 61.93±0.71 | 63.61±0.43 |

(c) Caltech101

| #Samples | ResNet18-Entropy | FR-ResNet18-Entropy | LLAL [1] | ResNet18-core-set |
|---|---|---|---|---|
| 1000 | **84.30**±0.69 | 83.46±0.53 | 83.89±0.33 | **84.30**±0.69 |
| 2000 | 93.27±0.37 | 93.30±0.33 | 91.88±0.50 | **93.50**±0.30 |
| 3000 | 95.24±0.27 | **95.38**±0.34 | 94.02±0.40 | 95.04±0.17 |
| 4000 | 95.50±0.15 | **95.83**±0.23 | 93.89±0.23 | 95.34±0.29 |
| 5000 | 95.42±0.34 | **95.70**±0.38 | 94.08±0.22 | 95.56±0.11 |
| 6000 | 95.54±0.17 | **95.90**±0.13 | 94.18±0.55 | 95.48±0.14 |

(d) Caltech256

| #Samples | ResNet18-Entropy | FR-ResNet18-Entropy | LLAL [1] | ResNet18-core-set |
|---|---|---|---|---|
| 5000 | 74.79±0.33 | **75.90**±0.20 | 74.06±0.45 | 74.79±0.33 |
| 6000 | 76.03±0.31 | **77.11**±0.07 | 74.47±0.21 | 76.52±0.36 |
| 7000 | 77.53±0.08 | **78.29**±0.19 | 75.21±0.33 | 77.44±0.17 |
| 8000 | 78.89±0.28 | **79.53**±0.29 | 75.92±0.36 | 78.12±0.22 |
| 9000 | 79.81±0.27 | **80.34**±0.42 | 76.56±0.19 | 79.20±0.26 |
| 10000 | 80.50±0.26 | **81.34**±0.34 | 76.82±0.36 | 79.93±0.16 |
| 11000 | 81.37±0.39 | **81.60**±0.24 | 77.12±0.21 | 80.65±0.27 |
| 12000 | 81.82±0.21 | **82.44**±0.18 | 77.59±0.25 | 80.95±0.19 |
| 13000 | 82.22±0.27 | **82.63**±0.15 | 77.66±0.52 | 81.62±0.18 |
| 14000 | 82.61±0.31 | **83.07**±0.22 | 78.14±0.27 | 81.97±0.25 |

Table 8: **Backbone Agnosticism.** We evaluate or approach with ResNet34, DenseNet121 [9] and EfficientNet-B3 [10] backbones. Our method is agnostic to the backbone showing benefit in every case.

(a) ResNet34-CIFAR10

| #Samples | ResNet34 | FR-ResNet34 |
|---|---|---|
| 1000 | 44.67±1.69 | **48.88**±1.41 |
| 2000 | 60.29±2.04 | **68.61**±1.48 |
| 3000 | 68.68±2.88 | **76.28**±0.69 |
| 4000 | 76.09±1.74 | **79.73**±0.45 |
| 5000 | 79.89±1.11 | **82.26**±0.39 |
| 6000 | 82.80±0.80 | **83.83**±0.20 |
| 7000 | 83.67±0.90 | **84.99**±0.18 |
| 8000 | 85.80±0.32 | **86.13**±0.32 |
| 9000 | 86.53±0.40 | **86.76**±0.15 |
| 10000 | **87.43**±0.41 | 87.33±0.28 |

(b) ResNet34-CIFAR100

| #Samples | ResNet34 | FR-ResNet34 |
|---|---|---|
| 5000 | 32.82±1.83 | **43.57**±1.29 |
| 6000 | 36.14±1.81 | **48.69**±1.04 |
| 7000 | 43.16±1.26 | **52.30**±0.75 |
| 8000 | 47.10±1.91 | **54.29**±0.83 |
| 9000 | 49.95±2.75 | **55.92**±0.39 |
| 10000 | 53.20±1.08 | **58.11**±0.34 |
| 11000 | 55.25±1.86 | **59.58**±0.29 |
| 12000 | 57.19±1.72 | **60.92**±0.41 |
| 13000 | 59.47±1.16 | **61.64**±0.46 |
| 14000 | 60.21±1.29 | **63.07**±0.25 |

(c) EfficientNetB3-CIFAR10 [10]

| #Samples | EfficientNetB3 [10] | FR-EfficientNetB3 |
|---|---|---|
| 1000 | 25.47±8.65 | **47.31**±1.79 |
| 2000 | 34.42±8.94 | **59.55**±3.52 |
| 3000 | 50.99±3.32 | **66.67**±1.98 |
| 4000 | 59.63±4.99 | **68.81**±2.89 |
| 5000 | 63.51±3.26 | **72.00**±1.27 |
| 6000 | 66.64±5.22 | **72.96**±1.77 |
| 7000 | 67.22±2.53 | **74.22**±2.80 |
| 8000 | 68.36±5.95 | **74.76**±3.14 |
| 9000 | 69.89±4.85 | **76.61**±0.75 |
| 10000 | 69.79±11.33 | **76.24**±1.45 |

(d) EfficientNetB3-CIFAR100 [10]

| #Samples | EfficientNetB3 [10] | FR-EfficientNetB3 |
|---|---|---|
| 5000 | 13.81±5.56 | **22.45**±1.91 |
| 6000 | 17.59±3.28 | **23.46**±1.50 |
| 7000 | 20.40±2.16 | **25.22**±1.82 |
| 8000 | 22.34±2.23 | **27.57**±2.01 |
| 9000 | 22.24±2.59 | **27.61**±2.93 |
| 10000 | 26.16±2.80 | **26.89**±1.18 |
| 11000 | 24.89±3.60 | **27.92**±1.93 |
| 12000 | 24.61±4.61 | **30.50**±2.07 |
| 13000 | 25.93±2.07 | **30.87**±2.37 |
| 14000 | 26.94±2.85 | **33.41**±2.63 |

(e) DenseNet121-CIFAR10 [9]

| #Samples | DenseNet121 [9] | FR-DenseNet121 |
|---|---|---|
| 1000 | 42.03±1.44 | **46.25**±0.77 |
| 2000 | 53.43±1.15 | **58.51**±0.74 |
| 3000 | 59.58±1.61 | **65.85**±0.57 |
| 4000 | 64.85±2.26 | **70.02**±0.50 |
| 5000 | 69.23±2.42 | **73.03**±0.54 |
| 6000 | 73.35±0.83 | **75.18**±0.36 |
| 7000 | 75.47±0.60 | **76.34**±0.36 |
| 8000 | 76.96±0.59 | **77.62**±0.42 |
| 9000 | 78.01±0.83 | **78.56**±0.39 |
| 10000 | 78.82±0.65 | **79.28**±0.26 |

(f) DenseNet121-CIFAR100 [9]

| #Samples | DenseNet121 [9] | FR-DenseNet121 |
|---|---|---|
| 5000 | 27.97±0.72 | **33.38**±0.45 |
| 6000 | 30.61±0.83 | **36.67**±0.72 |
| 7000 | 34.18±0.65 | **38.99**±0.94 |
| 8000 | 35.63±0.65 | **41.79**±0.69 |
| 9000 | 38.45±0.72 | **43.85**±0.81 |
| 10000 | 40.17±0.64 | **45.54**±0.34 |
| 11000 | 42.06±1.01 | **47.10**±0.63 |
| 12000 | 42.92±0.87 | **48.52**±0.23 |
| 13000 | 44.96±0.87 | **49.09**±0.35 |
| 14000 | 46.16±1.70 | **50.34**±0.30 |

Table 9: **Feature Refiner** 9a: We apply parts of our Feature Refiner step-by-step. First, we use only a single linear layer without any extra activation (512x512 w/o Activation), then apply our dimension reduction step (512x64 w/o Activation). Finally, we evaluate the effect of the LayerNorm layer.
**OJKD ablation** 9c: Our online joint knowledge distillation enables us to utilize the advantages of our Feature Refiner without increasing the number of model parameters at the same time. We also show that we can still improve upon the baseline without our Gradient Gate, but using it gives us extra improvement.
**Number of layers ablation** 9b: We study the effect of the number of nonlinear fully-connected layers. More layers do not lead to better performance.

(a) Feature Refiner

| #Samples | ResNet18 [8] | 512x512 w/o Activation | 512x64 w/o Activation | FR w/o LayerNorm | FR-ResNet18 |
|---|---|---|---|---|---|
| 1000 | 46.18±2.81 | 51.09±3.16 | 52.69±1.01 | 53.47±1.30 | **53.70**±1.18 |
| 2000 | 60.36±1.64 | 65.84±3.38 | 66.16±2.29 | 69.22±1.04 | **70.46**±0.59 |
| 3000 | 69.22±3.21 | 73.25±2.65 | 75.28±0.98 | **77.12**±0.29 | 76.81±0.50 |
| 4000 | 75.45±2.29 | 77.39±1.87 | 79.68±0.31 | **79.77**±0.27 | 79.66±0.21 |
| 5000 | 80.23±0.85 | 81.08±0.88 | 82.03±0.29 | 82.13±0.25 | **82.27**±0.33 |
| 6000 | 82.52±0.59 | 83.27±0.90 | 83.83±0.16 | **83.94**±0.36 | 83.65±0.34 |
| 7000 | 84.08±0.38 | 84.59±0.51 | 84.95±0.38 | **84.96**±0.39 | 84.48±0.26 |
| 8000 | 85.51±0.51 | 86.02±0.22 | **86.27**±0.29 | 86.16±0.38 | 85.88±0.21 |
| 9000 | 86.59±0.34 | 86.62±0.38 | **87.13**±0.16 | 86.69±0.25 | 86.42±0.31 |
| 10000 | 87.31±0.46 | 87.65±0.30 | **87.83**±0.28 | 87.65±0.42 | 87.20±0.29 |

(b) Number of layers

| #Samples | FR | FR-2layer | FR-3layer | FR-4layer | FR-5layer |
|---|---|---|---|---|---|
| 1000 | **53.70**±1.18 | 51.69±1.64 | 48.11±2.12 | 47.90±1.64 | 41.69±3.31 |
| 2000 | **70.46**±0.59 | 68.81±1.42 | 68.18±2.48 | 67.83±2.24 | 64.37±1.73 |
| 3000 | **76.81**±0.50 | 76.31±0.57 | 75.69±1.77 | 75.01±2.68 | 73.61±1.85 |
| 4000 | **79.66**±0.21 | 79.49±0.43 | 79.29±0.35 | 78.79±0.42 | 77.33±1.12 |
| 5000 | **82.27**±0.33 | 81.81±0.42 | 81.73±0.31 | 81.36±0.52 | 80.55±0.50 |
| 6000 | **83.65**±0.34 | 83.49±0.37 | 83.30±0.48 | 83.22±0.33 | 81.89±1.01 |
| 7000 | **84.48**±0.26 | 84.39±0.27 | 84.28±0.35 | 83.97±0.37 | 83.22±0.42 |
| 8000 | **85.88**±0.21 | 85.49±0.22 | 85.11±0.41 | 85.21±0.29 | 84.25±0.27 |
| 9000 | **86.42**±0.31 | 86.06±0.17 | 85.91±0.30 | 85.85±0.25 | 85.18±0.13 |
| 10000 | **87.20**±0.29 | 86.94±0.27 | 86.64±0.40 | 86.42±0.29 | 86.09±0.25 |

(c) OJKD

| #Samples | FR-ResNet18 ExtraHead | FR-ResNet18 OriginalHead | FR-ResNet18 OriginalHead w/o GradGate |
|---|---|---|---|
| 1000 | 53.57±1.02 | **53.70**±1.18 | 48.44±1.62 |
| 2000 | 70.26±0.61 | **70.46**±0.59 | 64.89±2.00 |
| 3000 | 76.66±0.55 | **76.81**±0.50 | 71.45±2.40 |
| 4000 | 79.53±0.24 | **79.66**±0.21 | 77.35±1.26 |
| 5000 | 82.12±0.35 | **82.27**±0.33 | 80.84±0.88 |
| 6000 | 83.63±0.29 | **83.65**±0.34 | 83.06±0.55 |
| 7000 | 84.41±0.26 | **84.48**±0.26 | 83.97±0.43 |
| 8000 | 85.86±0.26 | **85.88**±0.21 | 85.67±0.31 |
| 9000 | 86.48±0.32 | 86.42±0.31 | **86.59**±0.38 |
| 10000 | **87.24**±0.26 | 87.20±0.29 | 87.18±0.38 |

Table 10: **Robustness benchmark - Overall.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score over all datasets. Our method performs better in the earlier iterations.

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 40.59±1.65 | **46.66**±1.60 |
| 2000 | 51.46±2.91 | **58.07**±1.99 |
| 3000 | 58.11±2.79 | **61.45**±1.88 |
| 4000 | 61.10±2.27 | **61.83**±1.31 |
| 5000 | **62.80**±1.61 | 61.68±1.75 |
| 6000 | **63.78**±1.44 | 62.82±1.34 |
| 7000 | **64.31**±1.36 | 63.52±1.37 |
| 8000 | **65.50**±1.22 | 63.78±1.72 |
| 9000 | **65.65**±1.85 | 63.38±1.47 |
| 10000 | **65.96**±1.19 | 65.28±1.71 |

Table 11: **Robustness benchmark - Severity.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score for all severity levels. Our method consistently outperforms the baseline in cases of lower corruption severity. In cases of higher corruption severity (4d and 4e), in the very low-data regime, our method significantly outperforms the baseline. However, with more added data, the baseline starts outperforming our method.

(a) Severity 0

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 44.64±1.64 | **51.82**±1.48 |
| 2000 | 57.66±3.63 | **66.03**±1.59 |
| 3000 | 66.95±2.61 | **71.58**±1.01 |
| 4000 | 71.29±1.85 | **73.39**±0.78 |
| 5000 | 74.42±1.15 | **74.90**±0.75 |
| 6000 | 76.22±0.78 | **76.37**±0.77 |
| 7000 | **77.42**±0.68 | 76.91±0.68 |
| 8000 | **78.84**±0.65 | 77.93±0.88 |
| 9000 | **79.36**±0.79 | 78.11±0.61 |
| 10000 | **79.94**±0.71 | 79.45±0.85 |

(b) Severity 1

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 40.59±1.65 | **48.75**±1.47 |
| 2000 | 51.46±2.91 | **62.08**±1.83 |
| 3000 | 58.11±2.79 | **66.62**±1.67 |
| 4000 | 61.10±2.27 | **67.57**±1.15 |
| 5000 | 62.80±1.61 | **68.20**±1.27 |
| 6000 | 63.78±1.44 | **69.67**±1.16 |
| 7000 | 64.31±1.36 | **70.22**±1.05 |
| 8000 | 65.50±1.22 | **70.96**±1.44 |
| 9000 | 65.65±1.85 | **70.86**±1.15 |
| 10000 | 65.96±1.19 | **72.64**±1.33 |

(c) Severity 2

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 40.65±1.48 | **46.73**±1.45 |
| 2000 | 51.75±2.75 | **58.59**±2.01 |
| 3000 | 58.67±2.49 | **62.15**±1.88 |
| 4000 | 61.81±2.31 | **62.61**±1.39 |
| 5000 | **63.58**±1.53 | 62.39±1.80 |
| 6000 | **64.83**±1.46 | 63.70±1.41 |
| 7000 | **65.30**±1.48 | 64.39±1.40 |
| 8000 | **66.88**±1.18 | 64.78±1.85 |
| 9000 | **67.01**±1.99 | 64.34±1.51 |
| 10000 | **67.30**±1.31 | 66.45±1.75 |

(d) Severity 3

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 39.08±1.71 | **44.72**±1.64 |
| 2000 | 48.93±2.59 | **54.70**±2.17 |
| 3000 | 54.30±3.21 | **57.11**±2.31 |
| 4000 | 56.47±2.56 | **56.89**±1.57 |
| 5000 | **57.64**±1.91 | 55.99±2.32 |
| 6000 | **58.24**±1.79 | 56.87±1.64 |
| 7000 | **58.51**±1.75 | 57.86±1.85 |
| 8000 | **59.54**±1.67 | 57.62±2.22 |
| 9000 | **59.62**±2.48 | 56.98±1.96 |
| 10000 | **59.83**±1.43 | 59.23±2.28 |

(e) Severity 4

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 36.62±1.94 | **41.30**±1.94 |
| 2000 | 44.75±2.42 | **48.94**±2.35 |
| 3000 | 48.16±3.44 | **49.78**±2.55 |
| 4000 | **49.63**±2.62 | 48.71±1.66 |
| 5000 | **49.66**±2.16 | 46.93±2.59 |
| 6000 | **49.36**±2.02 | 47.48±1.71 |
| 7000 | **49.29**±1.83 | 48.23±1.86 |
| 8000 | **49.52**±1.88 | 47.61±2.21 |
| 9000 | **49.23**±2.64 | 46.60±2.12 |
| 10000 | **49.38**±1.47 | 48.63±2.32 |

Table 12: **Robustness benchmark - Corrpution types.** We evaluate the robustness of our method on the CIFAR10-C dataset. We present the average score for all corruption types. Our method consistently outperforms the baseline in the earlier iterations.

(a) Brightness

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 37.59±2.84 | **48.20**±1.32 |
| 2000 | 54.49±5.07 | **64.82**±1.91 |
| 3000 | 67.14±3.91 | **73.69**±0.93 |
| 4000 | 72.49±2.30 | **76.91**±0.78 |
| 5000 | 76.63±1.46 | **79.56**±0.41 |
| 6000 | 79.65±0.50 | **81.19**±0.35 |
| 7000 | 81.26±0.44 | **82.17**±0.40 |
| 8000 | 83.11±0.56 | **83.50**±0.30 |
| 9000 | 84.00±0.74 | **84.18**±0.41 |
| 10000 | **85.03**±0.54 | 85.01±0.41 |

(b) Contrast

| ResNet18 | FR-ResNet18 |
|---|---|
| 23.42±1.61 | **28.70**±1.62 |
| 30.62±3.60 | **41.13**±2.11 |
| 40.97±5.18 | **50.98**±2.13 |
| 46.10±3.49 | **55.30**±1.82 |
| 50.80±2.87 | **56.17**±2.15 |
| 55.69±1.01 | **59.32**±1.24 |
| 58.19±1.81 | **60.35**±0.99 |
| 61.30±1.16 | **61.58**±1.35 |
| 61.99±2.51 | **62.83**±1.04 |
| **63.70**±1.43 | 62.84±1.17 |

(c) Defocus Blur

| ResNet18 | FR-ResNet18 |
|---|---|
| 42.95±1.59 | **47.82**±1.56 |
| 54.19±2.76 | **59.36**±2.37 |
| 59.96±2.34 | **62.97**±1.31 |
| **63.63**±1.77 | 63.44±1.05 |
| **65.42**±1.09 | 64.29±1.92 |
| **66.52**±1.42 | 64.81±1.22 |
| **68.72**±1.00 | 67.48±1.52 |
| **69.99**±1.03 | 69.85±1.16 |
| **71.22**±1.86 | 70.25±1.20 |
| **71.81**±0.92 | 70.62±1.74 |

(d) Elastic Transform

| ResNet18 | FR-ResNet18 |
|---|---|
| 42.78±0.80 | **47.80**±1.18 |
| 54.17±2.68 | **59.62**±1.93 |
| 61.24±1.07 | **63.86**±0.87 |
| 65.28±1.33 | **65.30**±0.73 |
| **68.09**±0.92 | 67.12±1.18 |
| **69.94**±0.87 | 68.30±0.82 |
| **71.78**±0.77 | 69.71±0.94 |
| **73.03**±0.70 | 70.92±0.81 |
| **74.45**±0.92 | 71.94±0.52 |
| **74.83**±0.55 | 72.18±1.25 |

(e) Fog

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 31.14±1.49 | **37.96**±2.24 |
| 2000 | 42.19±5.44 | **56.09**±2.64 |
| 3000 | 56.09±5.01 | **66.27**±1.82 |
| 4000 | 62.36±3.69 | **70.24**±1.44 |
| 5000 | 67.62±2.53 | **72.08**±1.22 |
| 6000 | 71.34±0.69 | **74.88**±0.36 |
| 7000 | 73.76±1.00 | **75.69**±0.48 |
| 8000 | 76.34±0.70 | **77.01**±1.10 |
| 9000 | 76.93±1.62 | **77.41**±0.49 |
| 10000 | **78.23**±0.58 | 77.86±0.80 |

(f) Frost

| ResNet18 | FR-ResNet18 |
|---|---|
| 31.10±2.75 | **42.01**±1.40 |
| 46.39±4.69 | **56.12**±2.13 |
| 56.93±1.86 | **62.03**±1.00 |
| 60.45±2.12 | **62.64**±1.19 |
| **63.01**±1.34 | 62.82±1.26 |
| **64.00**±1.15 | 63.70±1.16 |
| **63.95**±1.46 | 62.90±0.85 |
| **65.62**±0.73 | 63.62±1.82 |
| **65.71**±1.17 | 63.22±1.36 |
| 65.36±1.53 | **65.45**±1.81 |

(g) Gaussian Blur

| ResNet18 | FR-ResNet18 |
|---|---|
| 41.41±1.94 | **45.19**±1.72 |
| 51.61±2.40 | **55.08**±2.56 |
| 55.34±2.73 | **57.22**±1.55 |
| **58.39**±1.97 | 56.24±1.35 |
| **59.07**±1.28 | 56.02±2.63 |
| **59.41**±1.89 | 56.19±1.79 |
| **61.62**±1.25 | 59.34±2.14 |
| **63.54**±2.43 | 61.63±1.79 |
| **63.62**±1.28 | 61.97±2.31 |

(h) Gaussian Noise

| ResNet18 | FR-ResNet18 |
|---|---|
| 44.93±1.33 | **50.12**±2.26 |
| 53.56±1.83 | **58.23**±1.91 |
| 55.22±3.72 | **55.53**±3.69 |
| **54.29**±3.50 | 51.91±1.88 |
| **52.94**±2.21 | 46.57±3.13 |
| **50.95**±2.13 | 46.62±1.81 |
| **47.14**±2.94 | 45.87±2.64 |
| **46.82**±2.47 | 40.84±3.51 |
| **43.21**±3.16 | 37.68±2.46 |
| 42.65±2.39 | **44.18**±3.70 |

(i) Glass Blur

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 43.52±0.92 | **47.97**±1.02 |
| 2000 | 51.15±1.17 | **53.77**±1.82 |
| 3000 | **52.75**±4.57 | 47.76±4.50 |
| 4000 | **52.87**±2.81 | 45.57±1.19 |
| 5000 | **48.92**±3.46 | 40.31±3.59 |
| 6000 | **45.82**±3.12 | 40.80±3.23 |
| 7000 | **42.95**±2.47 | 39.82±1.69 |
| 8000 | **42.23**±1.01 | 36.06±3.74 |
| 9000 | **41.94**±3.82 | 35.76±2.72 |
| 10000 | **41.80**±1.89 | 40.06±1.61 |

(j) Impulse Noise

| ResNet18 | FR-ResNet18 |
|---|---|
| 43.73±1.68 | **48.62**±2.10 |
| 50.82±2.62 | **56.10**±1.58 |
| 53.72±2.62 | **56.79**±2.51 |
| 54.97±2.36 | **57.51**±1.63 |
| **54.94**±2.68 | 54.15±1.83 |
| 54.43±2.42 | **54.49**±2.36 |
| 53.54±1.30 | **56.26**±2.16 |
| **54.80**±2.06 | 50.89±3.21 |
| **52.86**±2.68 | 49.95±3.26 |
| 53.27±1.23 | **54.26**±2.48 |

(k) JPEG Compression

| ResNet18 | FR-ResNet18 |
|---|---|
| 45.99±1.64 | **52.72**±1.68 |
| 58.20±3.15 | **65.35**±1.39 |
| 66.52±1.47 | **69.30**±0.66 |
| **70.34**±1.12 | 69.76±0.75 |
| **73.16**±0.40 | 71.36±0.92 |
| **74.11**±0.41 | 71.42±0.67 |
| **74.65**±0.75 | 71.43±1.14 |
| **74.64**±0.91 | 71.74±0.71 |
| **75.44**±0.83 | 72.02±0.36 |
| **75.12**±0.74 | 72.55±0.82 |

(l) Motion Blur

| ResNet18 | FR-ResNet18 |
|---|---|
| 41.28±0.73 | **44.41**±1.43 |
| 49.58±1.35 | **52.20**±2.18 |
| 52.84±1.76 | **55.48**±1.40 |
| 55.21±1.45 | **55.32**±0.97 |
| **56.10**±1.37 | 55.93±1.96 |
| **58.28**±1.55 | 57.58±1.83 |
| **61.26**±1.11 | 59.77±1.28 |
| **63.72**±0.68 | 62.99±2.29 |
| **64.79**±1.58 | 63.01±2.38 |
| **66.39**±0.85 | 63.87±2.22 |

(m) Pixelate

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 45.22±1.42 | **52.40**±1.40 |
| 2000 | 57.97±3.07 | **63.62**±1.82 |
| 3000 | **64.90**±3.04 | 64.38±2.37 |
| 4000 | **66.95**±2.42 | 63.82±1.79 |
| 5000 | **69.54**±1.37 | 66.11±1.55 |
| 6000 | **69.85**±1.83 | 66.15±1.20 |
| 7000 | **69.99**±1.02 | 65.58±1.60 |
| 8000 | 67.64±2.17 | **67.71**±1.40 |
| 9000 | **69.66**±2.17 | 67.06±2.22 |
| 10000 | **69.21**±1.25 | 68.16±1.47 |

(n) Saturate

| ResNet18 | FR-ResNet18 |
|---|---|
| 39.36±2.22 | **47.23**±2.06 |
| 52.11±4.35 | **62.93**±1.72 |
| 63.23±3.88 | **71.16**±1.44 |
| 69.32±2.68 | **74.67**±0.82 |
| 73.39±1.15 | **77.36**±0.74 |
| 76.89±0.75 | **79.39**±0.81 |
| 79.20±0.59 | **80.58**±0.46 |
| 81.37±0.62 | **81.92**±0.44 |
| 82.31±0.73 | **82.83**±0.37 |
| 83.09±0.47 | **83.09**±0.35 |

(o) Shot Noise

| ResNet18 | FR-ResNet18 |
|---|---|
| 45.43±1.46 | **50.87**±2.12 |
| 55.39±2.06 | **61.13**±2.02 |
| 58.90±2.99 | **60.46**±2.98 |
| **59.20**±3.05 | 57.72±1.80 |
| **58.82**±1.55 | 53.77±2.49 |
| **57.89**±1.92 | 54.55±1.57 |
| **55.05**±2.56 | 54.10±2.38 |
| **56.13**±2.39 | 51.48±3.16 |
| **53.03**±2.62 | 48.05±2.20 |
| **53.83**±3.34 | 52.35±2.08 |

(p) Snow

| ResNet18 | FR-ResNet18 |
|---|---|
| 39.21±2.46 | **47.02**±0.67 |
| 51.42±2.25 | **57.71**±1.91 |
| 58.89±1.22 | **62.09**±1.77 |
| 62.17±1.50 | **62.64**±1.26 |
| **64.21**±1.29 | 63.95±0.89 |
| 65.42±1.18 | **65.79**±1.05 |
| **65.80**±1.23 | 65.52±0.56 |
| **68.31**±0.55 | 67.19±1.13 |
| **69.31**±0.99 | 67.42±1.05 |
| **69.41**±1.23 | 69.12±0.91 |

(q) Spatter

| #Samples | ResNet18 | FR-ResNet18 |
|---|---|---|
| 1000 | 45.47±1.32 | **51.61**±1.21 |
| 2000 | 56.59±2.80 | **63.81**±1.02 |
| 3000 | 64.52±1.91 | **69.56**±0.66 |
| 4000 | 68.67±1.46 | **71.30**±1.04 |
| 5000 | 71.39±1.39 | **72.80**±0.38 |
| 6000 | 73.47±0.84 | **74.01**±0.88 |
| 7000 | **74.33**±0.71 | 74.24±0.82 |
| 8000 | **75.83**±0.45 | 75.07±0.62 |
| 9000 | **75.83**±0.91 | 75.37±0.75 |
| 10000 | **76.75**±0.67 | 75.43±0.77 |

(r) Speckle Noise

| esNet18 | FR-ResNet18 |
|---|---|
| 45.36±1.52 | **50.82**±2.03 |
| 55.40±2.23 | **60.97**±2.11 |
| 58.96±2.54 | **60.78**±2.86 |
| **59.23**±3.01 | 58.18±1.95 |
| **58.98**±1.42 | 54.54±2.32 |
| **58.44**±2.01 | 55.58±1.58 |
| **55.74**±2.54 | 55.50±2.30 |
| **57.57**±2.40 | 53.61±3.04 |
| **54.78**±2.46 | 50.15±2.10 |
| 54.24±1.95 | **55.86**±2.98 |

(s) Zoom Blur

| ResNet18 | FR-ResNet18 |
|---|---|
| 41.37±1.69 | **45.15**±1.31 |
| 51.85±2.12 | **55.22**±2.67 |
| 56.05±1.60 | **57.18**±1.35 |
| **59.03**±1.56 | 56.37±1.49 |
| **60.11**±1.15 | 56.99±2.60 |
| **60.39**±1.63 | 58.10±1.72 |
| **63.03**±1.18 | 60.61±1.61 |
| **64.07**±1.18 | 63.56±1.41 |
| **66.30**±2.24 | 63.39±1.24 |
| **66.41**±1.19 | 63.98±2.29 |

## References


[1] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *CVPR*, 2019. 1, 2, 8

[2] Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In *NeurIPS*, 2021. 2, 6

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 6

[4] Tianyu Hua et al. Simsiam. https://github.com/PatrickHua/SimSiam, 2021. 2, 7

[5] Mingjian Zhu, Kai Han, Yehui Tang, and Yunhe Wang. Visual transformer pruning. *CoRR*, abs/2104.08500, 2021. URL https://arxiv.org/abs/2104.08500. 2

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 3

[7] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HJz6tiCqYm. 3

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6, 10

[9] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 9

[10] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 9