Is Self-Supervised Contrastive Learning More Robust Than Supervised Learning?

Yuanyi Zhong [*1] **Haoran Tang** [*2] **Junkun Chen** [1] **Jian Peng** [1] **Yu-Xiong Wang** [1]

Abstract

Self-supervised contrastive learning is a powerful tool to learn visual representation without labels. Prior work has primarily focused on the recognition accuracy of contrastive pre-training algorithms, but has overlooked other *behavioral* aspects. In addition to accuracy, *distributional robustness* plays a critical role in the reliability of machine learning models. We design and conduct a series of robustness tests to quantify the behavioral differences between contrastive learning and supervised learning. These tests leverage data corruptions at multiple levels, ranging from pixel-level gamma distortion to patch-level shuffling and to dataset-level distribution shift. Our tests unveil intriguing robustness behaviors of contrastive and supervised learning. On the one hand, under downstream corruptions, we generally observe that contrastive learning is surprisingly more robust than supervised learning. On the other hand, under pre-training corruptions, we find contrastive learning vulnerable to patch shuffling and pixel intensity change, yet less sensitive to dataset-level distribution change. We attempt to explain these results through the role of data augmentation and feature space properties. Our insight has implications in improving the downstream robustness of supervised learning.

1. Introduction

In recent years, self-supervised contrastive learning (CL) has demonstrated tremendous potential in learning generalizable representations from unlabeled datasets (Chen et al., 2020a; He et al., 2020; Grill et al., 2020; Caron et al., 2020; Chen & He, 2021; Zhong et al., 2021b). Current state-of-the-art CL algorithms can learn representations from ImageNet (Deng et al., 2009) that match or even exceed the accuracy of their supervised learning (SL) counterparts in terms of ImageNet linear evaluation and downstream task performance (Chen et al., 2020a; He et al., 2020; Grill et al., 2020; Caron et al., 2020; Chen & He, 2021).

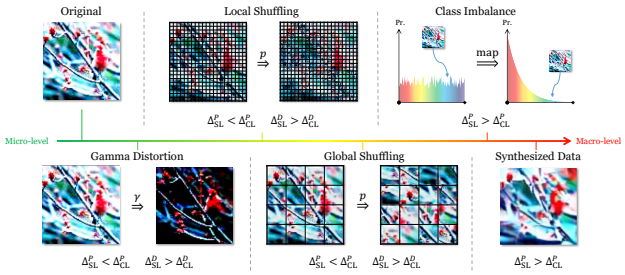

Figure 1: We conduct a series of robustness tests based on data distribution corruptions, spanning from micro to macro level, to study the behavior of contrastive and supervised learning beyond accuracy. Our results reveal that contrastive learning is usually more robust than supervised learning to *downstream* corruptions ($\Delta^D_{\text{CL}} < \Delta^D_{\text{SL}}$), while shows *opposite* behaviors to *pre-training dataset-level* corruptions ($\Delta^P_{\text{CL}} < \Delta^P_{\text{SL}}$) and *pre-training pixel- and patch-level* corruptions ($\Delta^P_{\text{CL}} > \Delta^P_{\text{SL}}$), where Δ is the accuracy drop from uncorrupted settings.

However, beyond accuracy, relatively less attention is paid on analyzing other *behavioral differences* between contrastive learning and supervised learning. Robustness is an important aspect to evaluate machine learning algorithms. For example, robustness to long-tail or noisy training data allows the learning algorithm to work well in a wide variety of imperfect real-world scenarios (Wang et al., 2017). Robustness of the model output across training iterations enables anytime early-stop (Hu et al., 2019) and smoother continual learning (Shen et al., 2020). Robustness to input corruptions at test-time plays an important role in reliable deployment of trained models in safety-critical applications, as signified by the existence of adversarial examples (Goodfellow et al., 2015; Salman et al., 2020) and the negative impact of domain shift (Zhao et al., 2019).

In this paper, we investigate whether CL and SL behave robustly to data distribution changes. In particular, how does the change in *data* affect the behavior of algorithms?

[*]Equal contribution [1]University of Illinois at Urbana-Champaign [2]University of Pennsylvania. Correspondence to: Yuanyi Zhong <yuanyiz2@illinois.edu>, Haoran Tang <thr99@seas.upenn.edu>, Junkun Chen <junkun3@illinois.edu>, Jian Peng <jianpeng@illinois.edu>, Yu-Xiong Wang <yxw@illinois.edu>.

First Workshop of Pre-training: Perspectives, Pitfalls, and Paths Forward at ICML 2022, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

Do SL and CL behave similarly? To this end, we design a wide-spectrum of corruptions as shown in Figure 1 to alter data distribution and conduct comprehensive experiments, with different backbones, CL algorithms and datasets. The corruptions are carefully selected to be multi-level, corrupt different structural information (not necessarily human-recognizable), and are rooted in prior literature.

Our main results consist of two sets of experiments: The first investigates the *downstream* robustness of pre-trained models towards corruptions of downstream data. The second studies the robustness under *pre-training* data corruptions. We deliver a set of **intriguing new discoveries**. We generally observe that CL is consistently more robust than SL to *downstream* corruptions. On the other hand, contrastive learning on corrupted *pre-training* data leads to *diverging* observations depending on the corruption type: CL is more robust to dataset-level corruption than SL, but less so to pixel- and patch-level corruptions.

To explain why CL models are more robust to downstream corruptions, our analysis of the learning dynamics through feature space metrics reveals that CL yields larger *overall* and steadily-increasing *per-class feature uniformity* (Wang & Isola, 2020) and higher stability than SL. An immediate consequence is an improvement to supervised pre-training's downstream robustness by adding a uniformity regularization term to explicitly promote intra-class variance. As for CL's vulnerability to pre-training data corruption types such as patch shuffling, we hypothesize that the interference with random crop augmentation is the main culprit, as we find switching the order of data corruption and standard augmentations recovers a substantial amount of robustness.

We summarize **our contributions** as follows. (1) We design extensive robustness tests to study the behavioral differences of CL and SL systematically. (2) We discover diverging robustness behaviors between CL and SL, and even among different CL algorithms. (3) We offer initial analyses and explanations for such observations, and show a simple way to improve the downstream robustness of supervised learning by encouraging uniformity. We claim our paper as an empirical study. We hope our findings can serve as an initial step to fully understand CL's behaviors beyond accuracy and inspire more future studies to explore such aspects through larger-scale experiments and theoretical analysis.

2. Method

2.1. Robustness Tests

Robustness Test I: Downstream data corruption. In this test, the pre-training algorithm is run on the clean version of the pre-training dataset. For a given downstream dataset, we evaluate the pre-trained model's accuracy on its original version and various corrupted versions. This assesses the robustness of the algorithm by looking at the pre-trained model's robustness behaviors.

Robustness Test II: Pre-training data corruption. This assesses the algorithm's robustness to pre-training data corruptions. We run the pre-training algorithm on the corrupted version of the dataset, and then evaluate the final model's accuracy on either the corrupted test set or the original test set. The test set can be in-domain or out-domain to the train set. Since data corruption destroys certain information by design, the model pre-trained on corrupted data is expected to yield degraded performance than the model pre-trained on the uncorrupted dataset.

Robustness Metric. In both cases, the robustness is measured by the degradation in accuracy caused by certain data corruption. An algorithm is more robust if the degradation is smaller. Denote $\mathcal{D}_{\text{original}}$ as the original dataset and $\mathcal{D}_{\text{corrupted}}$ as the corrupted dataset. We use the same notation for both pre-training and downstream corruptions. For an algorithm $\text{Alg} \in \{\text{CL}, \text{SL}\}$, we define

$$\Delta(\text{Alg}) = \frac{\text{Acc}(\text{Alg}, \mathcal{D}_{\text{original}}) - \text{Acc}(\text{Alg}, \mathcal{D}_{\text{corrupted}})}{\text{Acc}(\text{Alg}, \mathcal{D}_{\text{original}})}. \tag{1}$$

We use two methods to obtain the test accuracy in the above equation. The first is **linear evaluation**, where we train a linear classifier on top of the representation learned by the pre-training algorithm on the train split, and then evaluate the classifier on the test split. The second is **KNN evaluation**. We use the weighted KNN classifier from Wu et al. (2018), where the prediction is the exponential-distance weighted average of the K nearest neighbors in the train split of any test data point, measured by the normalized feature vectors. The KNN evaluation effectively leverages a non-parametric classifier, therefore no classifier training is required. Depending on the context, we use either linear or KNN evaluation in our experiments. The essential question we would like to ask is whether $\Delta(\text{CL})$ is consistently larger or smaller than $\Delta(\text{SL})$ across different data corruptions.

2.2. Types of Data Corruption

There is a natural hierarchy of data corruptions ranging conceptually from micro-level to macro-level. We will describe the several variations illustrated in Figure 1. Note that our data corruption is different from data augmentation. In data augmentation, the corruption is applied randomly on a per-image basis. In our case, a fixed random transformation (*e.g.*, the gamma in gamma distortion or the permutation order in shuffling) is decided first then applied consistently across all the images. We effectively transform the entire train or test dataset with the corruption method.

Pixel-Level Corruption. The pixel intensity distribution is altered, but neither the spatial layout of each image nor

the overall data distribution is changed. The popular color jittering augmentation randomly modifies the brightness, contrast, saturation, and hue of training images. Here, we deliberately pick gamma distortion, as it is not already part of the conventional data augmentation pipeline.

- **Gamma distortion:** Gamma distortion remaps each R,G,B pixel intensity ($\in [0, 255]$) according to $x \rightarrow \lfloor 255 \times (x/255)^{\gamma} \rfloor$, where $\gamma > 0$ is a tunable parameter. $\gamma = 1$ recovers the original intensity. Larger or smaller γ shifts the intensities darker or brighter, respectively. Due to quantization error, there will be part of the intensity information lost during the process.

Patch-Level Corruption. Pixel-level corruption does not change the spatial layout of pixels. The next type of corruption in the hierarchy is at patch level. Inspired by prior work (Zhang et al., 2017), we employ global and local shuffling. Note that pixel shuffling is not commonly used in the standard augmentation pipeline of visual recognition training (He et al., 2016). We are curious what kind of behaviors CL and SL will exhibit under the patch-level shuffling data corruption.

- **Global shuffling:** Global shuffling breaks down the image into patches and shuffles the patches according to a fixed random order. Specifically, suppose the image size is $s \times s$ and the size of each patch is $p \times p$, and then the image is divided into $s/p \times s/p$ patches. Global shuffling destroys the global spatial structure of an image but preserves the local structure. The image becomes less structured with a smaller patch size.

- **Local shuffling:** Inversely to global shuffling, local shuffling randomly permutes the pixels inside each local $p \times p$ patch by a fixed random order, but keeps the global ordering of patches. It damages the local image structure while preserving the overall global structure. The image becomes less structured with a larger patch size.

Dataset-Level Corruption. Finally, we consider corruptions happening at the whole dataset distribution level. The previous two types of corruptions change the images, but not the distribution of the images.

- **Synthesized data:** Synthesized data is popularizing (*e.g.*, DALL·E 2 (Ramesh et al., 2022)) and studied to replace real data (Jahanian et al., 2021). We utilize generative methods such as a conditional GAN (Karras et al., 2020) to generate a synthesized dataset $\mathcal{D}_{\text{GAN}}$ and replace $\mathcal{D}_{\text{original}}$. We can then measure and compare $\Delta(\text{Alg})$ between these two datasets of different distributions. Oftentimes, the generated distribution is not perfectly aligned with the real distribution; therefore, training with the generative data source may lead to degradation in accuracy of clean data or downstream performance.

- **Class imbalance:** Real-world data often follows a long-tail distribution of semantic classes, where head classes have many examples while tail classes have few examples (Kang et al., 2020; Samuel & Chechik, 2021). However, benchmark datasets such as CIFAR and ImageNet are curated and class-balanced. We consider the widely-used variant of ImageNet, ImageNet-LT (long-tail) (Liu et al., 2019), with maximally 1280 images and minimally 5 images per class. For comparison, we construct ImageNet-UF (uniform) which is a class-balanced subset of ImageNet that contains the same number of images as ImageNet-LT (115K). Then we test whether moving from pre-training on ImageNet-UF to ImageNet-LT leads to different behaviors between CL and SL.

2.3. Experiment Setup

Datasets. We pre-train on CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009) variants to evaluate the robustness differences between CL and SL methods subject to pre-training data corruptions. We use CIFAR-10/100 (Krizhevsky et al., 2009), STL-10 (Coates et al., 2011), and two fine-grained classification datasets, Cars (Krause et al., 2013) and Aircrafts (Maji et al., 2013), to analyze the performance of the pre-trained models on the corrupted downstream tasks. For comparable results, we fix the data augmentation for all settings that involve training.

Models and Algorithms. We benchmark a variety of self-supervised contrastive algorithms. These methods are carefully-sampled to be representative. They include contrastive learning with negatives (SimCLR-v2 (Chen et al., 2020a;b), MoCo-v2 (He et al., 2020; Chen et al., 2020d)), without negatives (SimSiam (Chen & He, 2021), and the momentum counterpart, BYOL (Grill et al., 2020)), with redundancy reduction (BarlowTwins (Zbontar et al., 2021)), and contrasting clustering assignments (DeepCluster-v2 (Caron et al., 2020), SwAV (Caron et al., 2020)). We test both CNN (standard ResNet-18/50 (He et al., 2016)) and Vision Transformer (ViT) (Dosovitskiy et al., 2021) backbones. For transformer, we leverage pre-trained models on ImageNet (Deng et al., 2009) from ViT (Dosovitskiy et al., 2021), DeiT (Touvron et al., 2021), DINO (Caron et al., 2021), MoCo-v3 (Chen et al., 2021b), and MAE (He et al., 2022).

3. Results

3.1. CL is more robust to downstream data corruption

We conduct downstream robustness tests on various datasets with frozen pre-trained ResNet-50 (He et al., 2016) in Table 1 and ViT (Dosovitskiy et al., 2021). The fully fine-tuned ResNet-50 results and the ViT results are deferred to the appendix as the observations are similar. The model checkpoints are obtained from VISSL (Goyal et al., 2021b) and

Table 1: Robustness to *downstream* pixel- and patch-level corruption with frozen ResNet-50 backbones. 'IN Acc' shows their reference ImageNet Val accuracy. Suffix '-a/b' denotes two models of the same algorithm, trained with different hyper-parameters. The Δ numbers are obtained from KNN evaluation, and are the averages of 5 downstream datasets and 6 corruption settings: gamma distortion $\gamma = 0.2, 5$, patch global shuffle and local shuffle ($p = 4$ and $p = \text{image_size}/4$). Darker shade means larger Δ. Please refer to the appendix for the detailed results. CL models generally show lower accuracy drops and therefore higher downstream robustness than SL models.

Pre-train Alg	Sup-a	Sup-b	BYOL	SimSiam	MoCo-v2-a	MoCo-v2-b	SimCLR-v2	BarlowTwins	DeepCluster-v2	SwAV-a	SwAV-b
IN Acc	76.1	75.5	72.3	68.3	66.4	71.1	71.0	73.5	75.2	72.0	74.9
Avg $\Delta \downarrow$	39.8%	40.7%	35.6%	34.4%	32.8%	35.7%	36.5%	34.7%	36.0%	33.9%	34.5%

Table 2: Robustness to *pre-training* pixel- and patch-level corruption of CIFAR10, with ResNet-18 backbones and linear evaluation. We pre-train four CL methods and SL with $p = \{4, 8\}$ patch shuffle and $\gamma = 0.2$ gamma distortion corruptions, and discover that SL is more robust than CL in this scenario.

CIFAR10	Orig	γ0.2	G4x4	G8x8	L4x4	L8x8	Avg $\Delta \downarrow$
Sup	89.53	87.36 (2.4%)	76.06 (15.0%)	65.88 (26.4%)	65.94 (26.3%)	77.49 (13.4%)	16.7%
MoCo-v2	88.73	85.84 (3.3%)	67.18 (24.3%)	60.51 (31.8%)	63.35 (28.6%)	76.90 (13.3%)	20.3%
BYOL	88.39	82.72 (6.4%)	67.47 (23.7%)	60.63 (31.4%)	62.64 (29.1%)	75.15 (15.0%)	21.1%
Barlow	88.89	80.49 (9.4%)	68.34 (23.1%)	61.13 (31.2%)	62.53 (29.7%)	75.28 (15.3%)	21.7%
DINO	84.75	69.27 (18.3%)	64.26 (24.2%)	55.83 (34.1%)	58.57 (30.9%)	68.96 (18.6%)	25.2%

Table 3: Robustness to *pre-training* pixel and patch-level corruption of ImageNet100. SL still shows higher robustness to the pixel-level and patch-level corruptions than CL, as in Table 2.

IN100	Orig	γ0.2	γ5	G2x2	G4x4	G8x8	L128x128	L64x64	L32x32	Avg $\Delta \downarrow$
Sup	77.08	73.60 (4.5%)	70.28 (8.8%)	67.26 (12.7%)	62.84 (18.5%)	58.20 (24.5%)	75.28 (2.3%)	72.68 (5.7%)	68.52 (11.1%)	6.65%
MoCo-v2	74.38	66.80 (10.2%)	62.74 (15.6%)	44.90 (39.6%)	35.84 (51.8%)	30.94 (58.4%)	69.34 (6.8%)	63.68 (14.4%)	54.24 (27.1%)	28.0%

Table 4: Robustness to *pre-training* dataset-level corruption led by substituting the real data with GAN synthesized data. KNN evaluation. MoCo demonstrates much more robustness to the distribution shift of synthesized data.

Pre-train Alg	Pre-train Data	C10 Test	C10 GAN Test	C100 Test	Avg $\Delta \downarrow$
Sup	Orig C10	87.8	88.3	16.08	-
	GAN C10	80.0 (8.88%)	82.8 (6.23%)	14.79 (8.02%)	7.71%
MoCo-v2	Orig C10	82.6	85.1	45.47	-
	GAN C10	82.2 (0.48%)	85.4 (-0.35%)	44.27 (2.64%)	0.93%

Table 5: Robustness to *pre-training* class-imbalance corruption. Linear evaluation. ImageNet-LT is a long-tail subset of ImageNet (Liu et al., 2019), ImageNet-UF is a uniform subset. MoCo shows less sensitivity to class imbalance than SL.

Alg	Pre-train	Top-1	Low	Med	Many
Sup	UF	46.37	44.85	45.88	47.52
	LT	44.90 (3.17%)	40.99 (8.61%)	43.48 (5.23%)	48.05 (-1.12%)
MoCo-v2	UF	32.36	30.63	31.66	33.84
	LT	32.13 (0.71%)	30.99 (-1.18%)	31.45 (0.66%)	33.36 (1.42%)

the official code bases. They are pre-trained on the clean version of ImageNet. We employ pixel-level and patch-level corruptions in these tests.

The CL methods have demonstrated higher robustness (lower average Δ) than the SL method. Interestingly, not all CL methods are equally robust; even within the same method, models trained with different hyper-parameters (such as epochs) exhibit different levels of robustness (*e.g.*, comparing SwAV-a and SwAV-b). With ResNet-50, we notice that SimSiam, SwAV, and BarlowTwins behave slightly more robust than others.

3.2. CLs and SL expose different degrees of robustness to different types of pre-training data corruption

Pixel-Level and Patch-Level Corruption. Table 2 demonstrates the impacts of gamma distortion and patch global/local shuffling during pre-training. The drop of accuracy of SL due to gamma distortion is 2.4%, which is smaller than all the tested CL methods. In terms of robustness to pre-training patch shuffling corruption, all CL methods behave similarly and less robustly than SL, except for the L8x8 case.

The same observation carries over to the larger-scale ImageNet-100 experiment result in Table 3, where we compare MoCo-v2 (since it appears to be the most robust on CIFAR-10) and Sup in a total of 8 corruption settings. MoCo-v2 on average yields a 28.0% degradation, while Sup only degrades 6.65%.

Dataset-Level Corruption. In Table 4, MoCo has shown more robustness on average to dataset-level corruption caused by an imperfect GAN. We adopt a class-conditional StyleGAN2-ADA (Karras et al., 2020) generator trained on CIFAR-10 to synthesize another copy of CIFAR-10. When training with synthesized data and testing on the original CIFAR-10, MoCo-v2 only has 2.58% accuracy drop, outperforming SL, which drops for 8.44%. Evaluating on a GAN-synthesized test set yields similar observation – MoCo-v2 shows almost no drop while SL drops 6%. We also conduct downstream experiments for this corruption: we test the classification accuracy of the pre-trained representation on CIFAR-100, and make a similar observation.

Table 5 shows the impact of the other type of dataset-level corruption – the class imbalance. Although there is a gap between the baseline top-1 accuracy of MoCo-v2 and SL, we observe that, the decline of MoCo resulting from pre-training on the long-tail rather than the uniform version is much smaller than SL. In fact, the MoCo performance appears to be insensitive to class balance or imbalance (the top-1 Δ is only 0.71%). This is in contrary to SL, which shows a larger drop. The difference is more salient by looking at the low-shot (<20 images per class), medium-shot, and many-shot (>100 images per class) accuracy separately. SL on ImageNet-LT sacrifices the low-shot accuracy for higher many-shot accuracy, whereas MoCo shows insignificant difference among the shots. Our observation is consistent with a contemporary work (Liu et al., 2022).

4. Analysis

CL and SL behave differently during training. The robustness discrepancy between CL (*e.g.*, MoCo) and SL

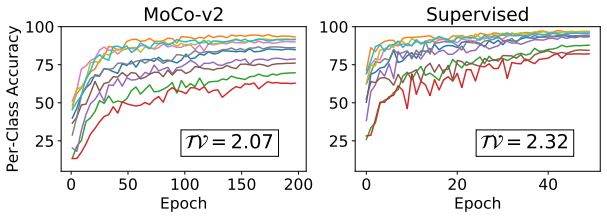

Figure 2: Class-wise test accuracy (*i.e.*, recall) of MoCo and SL on original CIFAR-10 during training. MoCo has more steady class-wise accuracy curves and smaller mean feature semantic fluctuation ($\mathcal{TV}$) than SL.

is not only reflected in the final trained models, but is in fact also attributed in the training process. To analyze how the feature space evolves during training, we measure the following three metrics.

Feature Semantic Fluctuation. We can monitor the classification ability of the feature extractor by the accuracy of a KNN probe. We define feature semantic fluctuation of class i as the total variation of per-class accuracy of class i (as a function of epoch t) averaged over all epochs: $\mathcal{TV}_i = \frac{1}{T-1}\sum_{t=0}^{T-2} |\text{Acc}_{t+1}^{(i)} - \text{Acc}_t^{(i)}|$. We further define the mean feature semantic fluctuation as the mean of $\mathcal{TV}_i$ over all classes. Larger semantic fluctuation indicates less stable feature space.

Feature Uniformity. We can measure the uniformity of all the features or class-wise features as the log-mean of Gaussian potentials of the normalized features: $U(f_t, \mathcal{D}) = -\log \mathbb{E}_{x_0, x_1 \sim \mathcal{D}} \left[e^{-2\|f_t(x_0) - f_t(x_1)\|_2^2} \right]$. Here f_t is the network at epoch t, $\mathcal{D}$ is the dataset, and x_0 and x_1 are images sampled from the dataset. The use of this measure to study contrastive learning is exemplified in Wang & Isola (2020). Intuitively, a greater U means more uniformly distributed features on the unit sphere, while a smaller value means more concentrated features.

Feature Distance. We also measure the average feature squared ℓ_2 distance between two classes. A larger distance could mean more linear separability. Denoting $\mathcal{D}_i$ and $\mathcal{D}_j$ as feature matrices of two classes, the feature distance is calculated as: $d(f_t, \mathcal{D}_i, \mathcal{D}_j) = \mathbb{E}_{x_0 \sim \mathcal{D}_i, x_1 \sim \mathcal{D}_j} \left[\|f_t(x_0) - f_t(x_1)\|_2^2 \right]$. When $\mathcal{D}_i = \mathcal{D}_j$, it actually measures the intra-class variance of class i.

We train ResNet-18 (He et al., 2016) on the original CIFAR-10 (Krizhevsky et al., 2009) train split and measure the above metrics on the test split. Figure 3 shows the dynamics of feature uniformity and distances of MoCo-v2 (He et al., 2020; Chen et al., 2020d), supervised contrastive (SupCon) (Khosla et al., 2020), and supervised learning. We are interested in SupCon, because it bridges CL and SL by leveraging a similar contrastive loss.

As illustrated, the overall feature uniformity of MoCo-v2

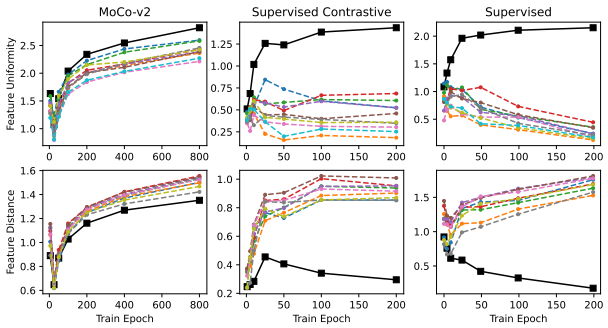

Figure 3: **Above:** Solid black line refers to the uniformity of the overall feature space. Dashed lines refer to the class-wise feature uniformities of the 10 classes. While the overall uniformity of all methods grows, the uniformity of each class (related to intra-class variance) of Sup or SupCon is shrinking as training progresses. In the end, the overall uniformity of MoCo is the largest. **Below:** Solid black line refers to $d(f_t, \mathcal{D}_0, \mathcal{D}_0)$, *i.e.*, the intra-class variance of class 0. Dashed lines refer to feature distances between $\mathcal{D}_i (i \neq 0)$ and $\mathcal{D}_0$. While the distances between classes increase in all methods, the intra-class variance behavior of MoCo (increasing) is the opposite to that of Sup or SupCon (decreasing).

(Chen et al., 2020d) is greater than 2.5 and approaching 3, while the overall feature uniformity of SupCon and supervised methods ranges from 1.25 to 2.2. This means that features from contrastive learning methods are more uniformly distributed on the unit sphere. By looking at the class-wise feature uniformity and distance, we notice that the supervised model tends to compress (and maybe over-compress) the features of each class. Figure 2 shows that the accuracy of a KNN probe during supervised learning also fluctuates more dramatically. We can interpret it as that the classes are competing with each other, and SL cannot improve the performance on all classes at the same time like CL methods. Further analysis on augmentation dependency and application of uniformity regularization to improve SL's performance are included in the appendix.

5. Conclusion

This paper systematically studies the robustness of contrastive learning and supervised learning through a diverse set of multi-level data corruption robustness tests. We discover diverging robustness behaviors of contrastive learning to different corruption settings. We hope our findings can draw more attention towards understanding the behavior differences of pre-training algorithms beyond accuracy.

Acknowledgement: This work was supported in part by NSF Grant 2106825, the Jump ARCHES endowment through the Health Care Engineering Systems Center, the New Frontiers Initiative, the National Center for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign through the NCSA Fellows program, and the IBM-Illinois Discovery Accelerator Institute.

References

Bao, H., Dong, L., and Wei, F. BEiT: BERT pre-training of image transformers. In *ICLR*, 2022.

Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *ICML*, 2020a.

Chen, T., Kornblith, S., Swersky, K., Norouzi, M., and Hinton, G. E. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020b.

Chen, T., Liu, S., Chang, S., Cheng, Y., Amini, L., and Wang, Z. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *CVPR*, 2020c.

Chen, T., Luo, C., and Li, L. Intriguing properties of contrastive losses. In *NeurIPS*, 2021a.

Chen, X. and He, K. Exploring simple siamese representation learning. In *CVPR*, 2021.

Chen, X., Fan, H., Girshick, R., and He, K. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020d.

Chen, X., Xie, S., and He, K. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021b.

Chuang, C.-Y., Hjelm, R. D., Wang, X., Vineet, V., Joshi, N., Torralba, A., Jegelka, S., and Song, Y. Robust contrastive learning against noisy views. *arXiv preprint arXiv:2201.04309*, 2022.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

Cole, E., Yang, X., Wilber, K., Mac Aodha, O., and Belongie, S. When does contrastive visual representation learning work? In *CVPR*, 2022.

da Costa, V. G. T., Fini, E., Nabi, M., Sebe, N., and Ricci, E. solo-learn: A library of self-supervised methods for visual representation learning. *JMLR*, 23(56):1–6, 2022. URL http://jmlr.org/papers/v23/21-1155.html.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Ericsson, L., Gouk, H., and Hospedales, T. M. How well do self-supervised models transfer? In *CVPR*, 2021.

Fan, L., Liu, S., Chen, P.-Y., Zhang, G., and Gan, C. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? In *NeurIPS*, 2021.

Ge, S., Mishra, S., Li, C.-L., Wang, H., and Jacobs, D. Robust contrastive learning using negative samples with diminished semantics. In *NeurIPS*, 2021.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *ICLR*, 2015.

Goyal, P., Caron, M., Lefaudeux, B., Xu, M., Wang, P., Pai, V., Singh, M., Liptchinsky, V., Misra, I., Joulin, A., and Bojanowski, P. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*, 2021a.

Goyal, P., Duval, Q., Reizenstein, J., Leavitt, M., Xu, M., Lefaudeux, B., Singh, M., Reis, V., Caron, M., Bojanowski, P., Joulin, A., and Misra, I. VISSL. https://github.com/facebookresearch/vissl, 2021b.

Goyal, P., Duval, Q., Seessel, I., Caron, M., Singh, M., Misra, I., Sagun, L., Joulin, A., and Bojanowski, P. Vision models are more robust and fair when pretrained on uncurated images without supervision. *arXiv preprint arXiv:2202.08360*, 2022.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Doersch, C., Pires, B., Guo, Z., Azar, M., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

He, K., Chen, X., xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR*, 2019.

Hendrycks, D., Mazeika, M., Kadavath, S., and Song, D. Using self-supervised learning can improve model robustness and uncertainty. In *NeurIPS*, 2019.

Hu, H., Dey, D., Hebert, M., and Bagnell, J. A. Learning anytime predictions in neural networks via adaptive loss balancing. In *AAAI*, 2019.

Jahanian, A., Puig, X., Tian, Y., and Isola, P. Generative models as a data source for multiview representation learning. *arXiv preprint arXiv:2106.05258*, 2021.

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., and Kalantidis, Y. Decoupling representation and classifier for long-tailed recognition. In *ICLR*, 2020.

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. Training generative adversarial networks with limited data. In *NeurIPS*, 2020.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., and Krishnan, D. Supervised contrastive learning. In *NeurIPS*, 2020.

Kim, M., Tack, J., and Hwang, S. J. Adversarial self-supervised contrastive learning. In *NeurIPS*, 2020.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *ICCV workshops*, 2013.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Liu, H., HaoChen, J. Z., Gaidon, A., and Ma, T. Self-supervised learning is more robust to dataset imbalance. In *ICLR*, 2022.

Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., and Yu, S. X. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *ICLR*, 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M., and Vedaldi, A. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

Nado, Z., Band, N., Collier, M., Djolonga, J., Dusenberry, M. W., Farquhar, S., Filos, A., Havasi, M., Jenatton, R., Jerfel, G., et al. Uncertainty baselines: Benchmarks for uncertainty & robustness in deep learning. *arXiv preprint arXiv:2106.04015*, 2021.

Naseer, M., Khan, S., Hayat, M., Khan, F. S., and Porikli, F. A self-supervised approach for adversarial robustness. In *CVPR*, 2020.

Neyshabur, B., Sedghi, H., and Zhang, C. What is being transferred in transfer learning? In *NeurIPS*, 2020.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. In *ICML*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

Salman, H., Ilyas, A., Engstrom, L., Kapoor, A., and Madry, A. Do adversarially robust imagenet models transfer better? In *NeurIPS*, 2020.

Samuel, D. and Chechik, G. Distributional robustness loss for long-tail learning. In *ICCV*, 2021.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

Shafahi, A., Najibi, M., Ghiasi, A., Xu, Z., Dickerson, J., Studer, C., Davis, L. S., Taylor, G., and Goldstein, T. Adversarial training for free! In *NeurIPS*, 2019.

Shen, Y., Xiong, Y., Xia, W., and Soatto, S. Towards backward-compatible representation learning. In *CVPR*, 2020.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, 2014.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The caltech-ucsd birds-200-2011 dataset. 2011.

Wang, F. and Liu, H. Understanding the behaviour of contrastive loss. In *CVPR*, 2021.

Wang, T. and Isola, P. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, 2020.

Wang, Y.-X., Ramanan, D., and Hebert, M. Learning to model the tail. In *NeurIPS*, 2017.

Wu, Z., Xiong, Y., Yu, S. X., and Lin, D. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018.

You, Y., Gitman, I., and Ginsburg, B. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.

Zhao, H., Des Combes, R. T., Zhang, K., and Gordon, G. On learning invariant representations for domain adaptation. In *ICML*, 2019.

Zhao, N., Wu, Z., Lau, R. W., and Lin, S. What makes instance discrimination good for transfer learning? In *ICLR*, 2021.

Zhong, Y., Wang, J., Peng, J., and Zhang, L. Boosting weakly supervised object detection with progressive knowledge transfer. In *ECCV*, 2020.

Zhong, Y., Wang, J., Wang, L., Peng, J., Wang, Y.-X., and Zhang, L. DAP: Detection-aware pre-training with weak supervision. In *CVPR*, 2021a.

Zhong, Y., Yuan, B., Wu, H., Yuan, Z., Peng, J., and Wang, Y.-X. Pixel contrastive-consistent semi-supervised semantic segmentation. In *ICCV*, 2021b.

A. Additional Analysis

A.1. Strong dependency on data augmentation may explain CL's non-robustness to patch shuffling

The different degree of dependency on data augmentation of CL and SL may explain why CL algorithms are less robust to pixel-level and patch-level pre-training corruptions. Contrastive learning relies heavily on well-defined data augmentation, while supervised learning can train without data augmentation as Row 1 of Table A.1. The central assumption in using data augmentation is that the augmented image falls within or close to the natural image statistics, *e.g.*, a random cropped image is still plausible. However, our data corruption such as patch shuffling destroys the structure of an image and thus renders the random resize-crop augmentation inappropriate, as the cropped image no longer has the same global structure.

To support our claim, we switch the order of the patch shuffling corruption and the standard augmentation in the MoCo experiments in Table A.1 (KNN evaluation), since shuffling after crop gives consistent image structure. We find that this switching reverts much of the accuracy drop, making MoCo comparably robust to Sup.

Table A.1: Additional pre-training corruption with Sup no-augmentation and Sup/MoCo augmentation-then-corrupt variants. SL is able to learn without data augmentation. Contrary to the corrupt-aug version in previous sections, MoCo and Sup share roughly a similar level of robustness with the aug-corrupt version.

Pre-training	Orig.	G4x4	G8x8	L4x4	L8x8	γ0.1
Sup no-aug	87.66	77.37 (11.7%)	71.86 (18.0%)	73.30 (16.4%)	82.34 (6.0%)	86.86 (0.91%)
Sup aug-corrupt	92.23	85.92 (6.8%)	80.58 (12.6%)	83.61 (9.4%)	89.96 (2.5%)	-
MoCo corrupt-aug	82.55	65.43 (17.1%)	59.49 (27.9%)	59.62 (27.8%)	70.14 (15.%)	-
MoCo aug-corrupt	82.55	77.63 (6.0%)	73.48 (11.0%)	78.12 (5.4%)	81.25 (1.6%)	-

A.2. Uniformity regularization improves downstream robustness of supervised pre-training

The analysis above shows that MoCo appears to yield a more uniformly distributed feature space and tends not to compress the intra-class variance of semantic classes. Could the larger uniformity be one reason behind the higher downstream robustness of MoCo? We introduce a small uniformity-promoting regularization term in addition to the cross-entropy loss in SL. This regularization is computed from the mini-batch itself without using the Siamese architecture or memory bank in MoCo.

In Table A.2, there is no surprise that adding (or subtracting) the uniformity regularization produces a more (or less) uniform test feature space. We also notice a correlation between the KNN evaluation accuracy and the test set uniformity under 3 corruption conditions. This experiment suggests that we could improve SL by leveraging loss functions from CL and potentially get the best of both worlds.

Table A.2: Supervised pre-training with uniformity regularization improves test-time robustness (ResNet-18, CIFAR-10, 200 epochs). The model achieves higher KNN evaluation accuracy on corrupted data while sacrificing little accuracy on the original data. Subtracting the uniformity promoting term appears to have the opposite effect.

Pre-train Alg	Metric	Orig	γ5	L4x4	G4x4
Sup	Acc, Unif	94.18, 1.98	72.85, 1.64	37.85, 0.98	39.70, 0.90
Sup+0.01Unif	Acc, Unif	94.21, 2.69	74.47, 2.03	42.22, 1.11	44.34, 1.30
Sup−0.01Unif	Acc, Unif	94.56, 1.12	71.50, 0.77	36.15, 0.41	37.88, 0.46

B. Related Work

Supervised Learning (SL). Supervised deep learning, large labeled dataset, and computation have been a success recipe for cracking many visual recognition problems (Russakovsky et al., 2015; He et al., 2016). Typically, one first collects a large-scale dataset for the target vision problem and crowdsources labels for the specific task from human annotators. A machine learning model, *e.g.*, a neural net, is then trained by minimizing a loss function defined on the prediction-label pairs.

Self-Supervised Learning (SSL) and Contrastive Learning (CL). Remarkable progress has been made in self-supervised representation learning from unlabeled datasets (Chen et al., 2020a; He et al., 2020; Grill et al., 2020; Chen & He, 2021; Caron et al., 2020). This paper focuses on a particular kind of SSL algorithm, contrastive learning, that learns augmentation invariance with a Siamese network. To prevent trivial solution, contrastive learning pushes negative examples apart (MoCo (He et al., 2020; Chen et al., 2020d; 2021b), SimCLR (Chen et al., 2020a;b)), makes use of stop-gradient operation or asymmetric predictor without using negatives (SimSiam (Chen & He, 2021), BYOL (Grill et al., 2020), DINO (Caron et al., 2021)), or leverages redundancy reduction (BarlowTwins (Zbontar et al., 2021)) and clustering (DeepCluster-v2 and SwAV (Caron et al., 2020)). In addition to augmentation invariance, generative pre-training (Ramesh et al., 2021; Bao et al., 2022; He et al., 2022) and visual-language pre-training (Radford et al., 2021) are promising ways to learn transferable representations.

There is a growing body of literature on understanding self-supervised learning. Wang & Liu (2021) decomposes the contrastive objective into alignment (between augmentations) and uniformity (across entire feature space) terms. Uniformity can be thought of as an estimate of the feature entropy, which we leverage as a metric to study the feature space dynamics during training. Wang & Isola (2020) makes connection between uniformity and the temperature parameter in a contrastive loss, and finds that a good temperature can balance uniformity and tolerance of semantically similar examples. Zhao et al. (2021) discovers that SSL transferring better than SL can be due to better low- and mid-level features, and the intra-class invariance objective in SL weakens transferability by causing more pre-training and downstream task misalignment. Ericsson et al. (2021) studies the downstream task accuracy of a variety of pre-trained models and finds that SSL outperforms SL on many tasks. Cole et al. (2022) investigates the impact of pre-training data size, domain quality, and task granularity on downstream performance. Chen et al. (2021a) identifies three intriguing properties of contrastive learning: a generalized version of the loss, learning with the presence of multiple objects, and feature suppression induced by competing augmentations. Our work falls into the same line of research that attempts to understand SSL better. However, we investigate from the angle of *robustness behavior comparison* between SSL/CL and SL.

Robustness and Data Corruption. The success of learning algorithms is often measured by some form of task accuracy, such as the top-1 accuracy for image classification (Deng et al., 2009; Krizhevsky et al., 2009; Coates et al., 2011; Wah et al., 2011), or the mean average precision for object detection (He et al., 2020; Zhong et al., 2020; 2021a). Beyond accuracy, robustness is another important measure (Hendrycks & Dietterich, 2019), and there are benchmarks and metrics proposed for SL (Nado et al., 2021). Robustness is more and more studied in SSL settings (Chuang et al., 2022; Goyal et al., 2022). Chuang et al. (2022) tries to improve CL's robustness to noisy positive views. Recent large-scale study reveals that vision models are more robust and fair when pre-trained on uncurated images without supervision (Goyal et al., 2022). We use "robustness" to refer to the ability of learning algorithms to cope with systematic train or test data corruptions. Under the supervised setting, deep models are shown to train successfully (albeit not to generalize) under pixel shuffling corruption and random labels, even though they are not human-recognizable anymore (Zhang et al., 2017).

Adversarial robustness (Szegedy et al., 2014; Goodfellow et al., 2015; Madry et al., 2018; Shafahi et al., 2019; Chen et al., 2020c) is a related but different concept, which refers to the model's ability to defend against adversarial attacks. An adversarial attack (Szegedy et al., 2014; Goodfellow et al., 2015) is a perceptually indistinguishable perturbation to a *single* image that fools the model. Adversarial training (Madry et al., 2018; Shafahi et al., 2019) is a technique to achieve adversarial robustness. Self-supervised perturbation is explored in adversarial attack and training (Naseer et al., 2020; Kim et al., 2020). Hendrycks et al. (2019) shows that SSL models possess better adversarial robustness. Fan et al. (2021) improves the adversarial robustness transferability of CL. Our definition of robustness differs from adversarial robustness – we use robustness to analyze the tolerance of learning methods to *systematic data corruptions* (rather than per-image imperceptible perturbation).

There are many types of data corruptions in prior work. The most common data corruptions, such as random resizing and cropping, flipping, and color jittering, appear as data augmentation in SL and SSL (He et al., 2016; 2020; Chen et al., 2020a). The learned representation is encouraged to be invariant to such corruptions. Hendrycks & Dietterich (2019) proposes a set of corruptions complementary to ours. Block shuffling (our image global shuffling) has been used to study what is transferred in transfer learning (Neyshabur et al., 2020) and as negative views with diminished semantics in contrastive learning (Ge et al., 2021). Cole et al. (2022) tampers data quality in SimCLR and SL training by salt-and-pepper noise, JPEG, resizing, and downsampling, and tests on clean data. We use a different set of data corruptions and test on the corrupted data as well. A recent work (Jahanian et al., 2021) also studies generative models as an alternative data source for contrastive learning. They focus on comparison with real data, while we emphasize the behavior difference of SSL and SL in response to the generative data source. Feature backward-compatibility (Shen et al., 2020) is related to our stability analysis of feature dynamics. Recently, Goyal et al. (2021a) studies the effectiveness of SSL on uncurated class-imbalanced data. Liu et al. (2022) also notices that SSL tends to be more robust to class imbalance than SL. We bring extra insights over them. We consider *both pre-training and downstream* robustness and compare *CL and SL behaviors*, while Goyal et al. (2021a) only focuses on downstream and compares dataset scale. Our investigation suggests that pre-train behavior can be *opposite* to downstream. Liu et al. (2022) only studies class imbalance, but we consider broader corruptions.

C. Additional Implementation Details

Table C.3 below lists the experiment configurations for each pre-training robustness table of the main paper. We train our own ResNets (He et al., 2016) on CIFAR-10 (Krizhevsky et al., 2009) and ImageNet variants (Deng et al., 2009). ImageNet-

LT/UF are the long-tail and Uniform-subsampled versions. ImageNet-100 is a 100 class subset of full ImageNet-1K. We mainly list Sup and MoCo-v2 (Chen et al., 2020d) hyper-parameters here. The other CL methods follow their recommended hyper-parameter values in the Solo-Learn package (da Costa et al., 2022).

Table C.3: Implementation details for the pre-training results in the main paper.

Config	Tables 2,4	Table 5	Table 3
Pretrain dataset	CIFAR-10	ImageNet-LT/UF	ImageNet-100
# of categories	10	1000	100
Train image size	32	224	224
Train data size	50K	115K	130K
Network	ResNet-18	ResNet-50	ResNet-18
Backbone out dim	512	2048	512
Sup epochs	50	200	50
Sup lr	0.1 cos	0.015 cos	0.015 cos
Sup batch size	512	128	128
MoCo epochs	200	200	200
MoCo lr	0.06 cos	0.015 cos	0.03 cos
MoCo batch size	512	128	256
MoCo dim	128	128	128
MoCo temp.	0.1	0.2	0.2
MoCo momentum	0.99	0.999	0.999
MoCo queue size	4096	65536	65536
Evaluation	Linear	Linear	Linear
Augmentation	crop+flip+ color(.4,p=.8) +gray(p=.2)	crop+flip+ color(.4,p=.8)+gray(p=.2)+ gauss(.1,.2,p=.5)	

D. Additional Results

D.1. Pre-training robustness test with Transformer backbone

In the main paper, we compare pre-training robustness with a CNN backbone, and show Vision Transformer (ViT) downstream robustness test results. Here, we supplement ViT *pre-training* robustness test results. Specifically, we leverage MoCo-v3 (Chen et al., 2021b), the ViT version of MoCo, and Supervised ViT. The results are in Table D.1. We find that the MoCo-v3 degradation is larger with patch shuffling, but smaller with gamma distortion. Interestingly, the impact of patch shuffling is much smaller than a CNN (despite the Orig performance gap between ViT and CNN). We suspect this is due to the unique patching and attention network structure of ViT. Essentially, if we do not take into consideration the data augmentation, with the right patch size, the shuffling within a small patch does not affect the learning of ViT much, and the global ordering of patches also does not matter much, because of learned positional embeddings and global attention.

Table D.1: Pre-training robustness with ViT on CIFAR10: MoCo-v3 vs. Sup. For the ViT architecture, since the input size (32x32) is smaller than that of a standard ViT, we use a customized small ViT (image size=32, patch size=4, dim=512, depth=6, heads=8, mlp dim=512, dropout=0.1, emb dropout=0.1).

Method	Orig	G4x4	G16x16	L4x4	L16x16	Avg Δ	$\gamma = 0.1$
Sup ViT 50ep	67.92	59.01	47.97	57.76	67.95	-	52.96
Δ	-	13.12%	29.37%	14.96%	-0.04%	14.35%	22.03%
MoCo-v3 200ep	62.78	53.36	41.58	53.52	61.77	-	51.41
Δ	-	15.0%	33.77%	14.75%	1.61%	16.28%	18.11%

D.2. Pre-training robustness test with longer epochs

In Table 2 of the main paper, we mostly report results of short pre-training schedules: Sup 30 epochs and CL 200 epochs, in order to make the baseline results comparable. We report CIFAR-10 longer training epochs in Table D.2. Training longer does not change our observation that MoCo appears less robust to patch and pixel-level corruptions than SL during pre-training on this dataset.

D.3. Downstream robustness test with ResNet-50: detailed KNN accuracy numbers

Table D.3 show the detailed accuracy numbers for computing the summary statistics in Table 1 of the main paper.

Table D.2: Pre-training robustness: Sup 50ep vs. MoCo-v2 400ep, ResNet-18, CIFAR-10.

Method	Orig	G4x4	G8x8	L4x4	L8x8	Avg Δ	$\gamma = 0.1$
Supervised 50ep	92.23	81.14	71.33	71.72	81.95	-	89.11
Δ	-	12.02%	22.67%	22.24%	11.15%	17.02%	3.38%
MoCo-v2 400ep	91.43	70.99	64.25	66.56	81.51	-	83.94
Δ	-	22.36%	29.73%	27.20%	10.85%	22.54%	8.19%

D.4. Downstream robustness test with ViT backbone

Table D.4: Robustness to downstream pixel- and patch-level corruption with ViT backbone (Dosovitskiy et al., 2021). We show KNN accuracies and the Δ's on three datasets. ViT-based CL models are also more robust than the two SL models, especially to gamma distortion. Comparing the generative method, MAE (He et al., 2022), to contrastive learning, despite being inferior performance on STL10, it slightly outperforms contrastive methods for patch shuffling on CIFAR10/100, but is more vulnerable to gamma distortion.

STL10	Orig	γ0.2	γ2.5	G4x4	G24x24	L4x4	L24x24	Avg Δ
ViT (Sup)	98.85	91.71 (7.2%)	91.39 (7.5%)	88.96 (10.0%)	43.69 (55.8%)	45.95 (53.5%)	70.89 (28.3%)	27.1%
DeiT (Sup)	98.64	97.58 (1.1%)	98.01 (0.6%)	92.92 (5.8%)	46.99 (52.4%)	45.60 (53.8%)	73.22 (25.8%)	23.3%
DINO	98.91	98.31 (0.6%)	98.17 (0.7%)	95.30 (3.7%)	50.36 (49.1%)	52.35 (47.1%)	79.96 (19.2%)	20.1%
MoCo-v3	97.89	97.11 (0.8%)	96.75 (1.2%)	91.24 (6.8%)	48.86 (50.1%)	47.70 (51.3%)	74.88 (23.5%)	22.3%
MAE	90.74	83.54 (7.9%)	87.42 (3.7%)	72.54 (20.1%)	46.35 (48.9%)	46.20 (49.1%)	60.15 (33.7%)	27.2%
CIFAR10	Orig	γ0.2	γ2.5	G4x4	G8x8	L4x4	L8x8	
ViT (Sup)	94.23	71.42 (24.2%)	82.37 (12.6%)	64.09 (32.0%)	52.58 (44.2%)	52.54 (44.2%)	59.63 (36.7%)	32.3%
DeiT (Sup)	95.37	90.66 (4.9%)	92.78 (2.7%)	73.24 (23.2%)	59.48 (37.6%)	53.10 (44.3%)	59.65 (37.5%)	25.0%
DINO	96.68	92.85 (4.0%)	94.65 (2.1%)	77.99 (19.3%)	64.63 (33.2%)	60.79 (37.1%)	68.04 (29.6%)	20.9%
MoCo-v3	96.16	91.90 (4.4%)	94.17 (2.1%)	75.30 (21.7%)	61.14 (36.4%)	57.60 (40.1%)	64.43 (33.0%)	22.9%
MAE	77.06	71.00 (7.9%)	72.04 (6.5%)	61.25 (20.5%)	55.06 (28.5%)	53.31 (30.8%)	56.99 (26.0%)	20.0%
CIFAR100	Orig	γ0.2	γ2.5	G4x4	G8x8	L4x4	L8x8	
ViT (Sup)	79.86	48.70 (39.0%)	60.87 (23.8%)	40.95 (48.7%)	29.84 (62.6%)	28.91 (63.8%)	35.31 (55.8%)	49.0%
DeiT (Sup)	78.23	68.98 (11.8%)	73.00 (6.7%)	49.86 (36.3%)	34.81 (55.5%)	29.49 (62.3%)	36.12 (53.8%)	37.7%
DINO	83.88	75.76 (9.7%)	79.21 (5.6%)	56.81 (32.3%)	40.80 (51.4%)	36.62 (56.3%)	44.82 (46.6%)	33.7%
MoCo-v3	82.32	73.25 (11.0%)	77.42 (6.0%)	53.07 (35.5%)	37.75 (54.1%)	33.00 (59.9%)	40.79 (50.4%)	36.2%
MAE	53.70	47.72 (11.1%)	49.46 (7.9%)	37.18 (30.8%)	30.93 (42.4%)	29.36 (45.3%)	34.18 (36.4%)	29.0%

D.5. Downstream robustness test with ResNet-50 and full fine-tuning

Table 1 in the main paper and Table D.3 above are generated with the KNN evaluation protocol. We also experiment with full fine-tuning on the downstream datasets. The results are in Table D.5. Since different pre-trained checkpoints are optimized with different optimizers (SGD for Sup, SimSiam (Chen & He, 2021), MoCo-v2 (Chen et al., 2020d), and SimCLR-v2 (Chen et al., 2020b); LARS (You et al., 2017) for BYOL (Grill et al., 2020), BarlowTwins (Zbontar et al., 2021), DeepCluster2, and SwAV (Caron et al., 2020)), we use SGD (lr 0.002 cosine) for Sup, SimSiam, MoCo, and SimCLR, and AdamW (lr 0.001 cosine) (Loshchilov & Hutter, 2019) for others during fine-tuning. All models are fine-tuned for 10 epochs. We find this strategy of using different optimizers is able to make the baseline results on original images comparable across methods. We note that fine-tuning drastically improves the accuracy on downstream datasets, while the general observation that CL methods are more robust to downstream corruption than SL still holds, except for BarlowTwins which is slightly worse than SL. Another interesting observation here is that *different CL methods actually yield different robustness behaviors*, although they are all doing some form of contrastive learning and have similar baseline accuracies.

D.6. Variance of pre-training results

We repeat MoCo-v2 on the original CIFAR-10 200ep three times: The KNN evaluation mean and std is 82.44 ± 0.18. Repeating MoCo-v2 on the global 8x8 shuffling corrupted CIFAR-10 gives KNN evaluation mean and std 59.24 ± 0.40. The linear evaluation variance is similar. The randomness has a smaller order than the gap between MoCo and Supervised results.

Table D.3: Robustness to downstream data corruption with *KNN evaluation*. This table contains the detailed top-1 accuracy numbers constituting Table 1 in the main paper. The shades of yellow in the last column indicate the size of the numbers.

Pre-train Alg	Dataset	Orig	$\gamma = 0.2$	$\gamma = 5$	G-small	G-large	L-small	L-large	Avg Δ
Sup	cifar10	86.4	76.4 (11.7%)	67.6 (21.8%)	61.2 (29.2%)	49.6 (42.6%)	46.9 (45.7%)	53.6 (38.0%)	31.5%
Sup	cifar100	65.1	52.4 (19.6%)	44.1 (32.3%)	37.4 (42.6%)	25.7 (60.5%)	23.4 (64.1%)	30.8 (52.8%)	45.3%
Sup	stl10	96.6	92.2 (4.6%)	82.6 (14.5%)	80.8 (16.3%)	40.7 (57.8%)	43.4 (55.1%)	60.3 (37.5%)	31.0%
Sup	cars196	26.8	23.3 (13.2%)	18.1 (32.5%)	12.7 (52.7%)	4.4 (83.6%)	4.1 (84.9%)	16.0 (40.4%)	51.2%
Sup	aircraft70	40.3	37.9 (6.0%)	38.8 (3.6%)	25.2 (37.4%)	8.0 (80.3%)	10.0 (75.2%)	25.5 (36.7%)	39.9%
Sup-b	cifar10	84.9	77.0 (9.3%)	68.9 (18.9%)	57.7 (32.0%)	46.5 (45.2%)	44.0 (48.2%)	51.9 (38.9%)	32.1%
Sup-b	cifar100	63.2	53.3 (15.7%)	45.4 (28.1%)	33.0 (47.8%)	21.4 (66.1%)	18.9 (70.0%)	28.0 (55.7%)	47.2%
Sup-b	stl10	96.0	92.8 (3.3%)	83.6 (12.9%)	77.9 (18.9%)	36.0 (62.5%)	41.5 (56.7%)	60.4 (37.0%)	31.9%
Sup-b	cars196	28.8	26.8 (7.0%)	22.0 (23.8%)	12.1 (58.1%)	1.8 (93.7%)	2.4 (91.6%)	15.8 (45.2%)	53.2%
Sup-b	aircraft70	46.9	47.0 (-0.3%)	46.0 (1.9%)	29.8 (36.4%)	7.7 (83.5%)	10.8 (76.9%)	29.8 (36.5%)	39.2%
BYOL	cifar10	87.5	80.3 (8.3%)	72.4 (17.3%)	64.0 (26.8%)	50.7 (42.1%)	48.4 (44.7%)	55.6 (36.5%)	29.3%
BYOL	cifar100	67.4	58.1 (13.8%)	49.8 (26.0%)	39.7 (41.1%)	26.1 (61.3%)	24.5 (63.6%)	32.1 (52.3%)	43.0%
BYOL	stl10	94.9	92.4 (2.6%)	85.0 (10.4%)	78.1 (17.7%)	41.5 (56.3%)	43.8 (53.8%)	63.0 (33.6%)	29.0%
BYOL	cars196	22.5	22.3 (0.8%)	18.7 (17.0%)	12.6 (44.1%)	2.5 (88.9%)	2.3 (89.6%)	18.6 (17.2%)	42.9%
BYOL	aircraft70	38.2	39.5 (-3.3%)	38.6 (-1.2%)	24.0 (37.2%)	8.0 (79.2%)	10.5 (72.6%)	31.1 (18.5%)	33.8%
SimSiam	cifar10	83.6	77.6 (7.2%)	68.7 (17.8%)	61.5 (26.5%)	50.4 (39.8%)	48.4 (42.2%)	55.6 (33.5%)	27.8%
SimSiam	cifar100	59.9	52.7 (12.1%)	44.8 (25.1%)	36.1 (39.7%)	24.8 (58.6%)	23.2 (61.2%)	31.1 (48.0%)	40.8%
SimSiam	stl10	92.1	89.6 (2.7%)	81.1 (12.0%)	74.2 (19.4%)	39.1 (57.6%)	43.8 (52.5%)	62.7 (31.9%)	29.3%
SimSiam	cars196	17.1	15.9 (6.9%)	13.9 (18.5%)	10.3 (39.4%)	2.0 (88.6%)	2.5 (85.2%)	15.3 (10.2%)	41.5%
SimSiam	aircraft70	32.8	34.1 (-3.7%)	32.6 (0.8%)	20.9 (36.3%)	6.8 (79.3%)	10.8 (67.2%)	27.6 (15.9%)	32.6%
MoCo	cifar10	81.4	74.0 (9.1%)	66.1 (18.9%)	60.2 (26.0%)	49.5 (39.2%)	47.2 (42.0%)	54.0 (33.7%)	28.1%
MoCo	cifar100	56.6	48.2 (14.8%)	41.9 (26.0%)	35.3 (37.7%)	23.5 (58.5%)	23.1 (59.3%)	30.0 (46.9%)	40.5%
MoCo	stl10	90.1	88.1 (2.2%)	77.5 (13.9%)	73.5 (18.5%)	39.9 (55.7%)	42.8 (52.5%)	60.0 (33.4%)	29.4%
MoCo	cars196	13.1	12.8 (2.5%)	11.3 (14.2%)	8.8 (32.9%)	2.3 (82.6%)	2.6 (80.5%)	12.1 (8.2%)	36.8%
MoCo	aircraft70	25.2	26.7 (-6.1%)	23.4 (7.2%)	17.3 (31.5%)	8.1 (67.9%)	10.0 (60.2%)	21.2 (15.7%)	29.4%
MoCo-b	cifar10	83.6	76.1 (9.0%)	67.1 (19.7%)	57.2 (31.5%)	47.2 (43.5%)	45.6 (45.5%)	51.0 (38.9%)	31.3%
MoCo-b	cifar100	59.5	49.9 (16.1%)	42.4 (28.7%)	32.9 (44.7%)	21.4 (64.0%)	21.1 (64.5%)	27.9 (53.0%)	45.2%
MoCo-b	stl10	95.3	92.8 (2.6%)	85.1 (10.6%)	76.0 (20.2%)	38.9 (59.2%)	40.4 (57.6%)	60.9 (36.1%)	31.0%
MoCo-b	cars196	13.8	13.7 (1.3%)	12.1 (12.8%)	7.9 (42.7%)	1.8 (86.9%)	1.9 (86.3%)	12.7 (8.4%)	39.7%
MoCo-b	aircraft70	26.8	28.1 (-4.9%)	26.4 (1.5%)	17.2 (35.8%)	6.9 (74.4%)	8.6 (67.8%)	23.3 (13.2%)	31.3%
SimCLR2	cifar10	85.4	79.2 (7.3%)	67.5 (21.0%)	58.0 (32.1%)	45.6 (46.6%)	45.8 (46.4%)	54.8 (35.8%)	31.5%
SimCLR2	cifar100	63.5	55.2 (13.1%)	44.6 (29.8%)	33.2 (47.7%)	21.2 (66.7%)	22.4 (64.7%)	31.4 (50.6%)	45.4%
SimCLR2	stl10	91.9	89.3 (2.9%)	81.7 (11.2%)	69.8 (24.1%)	38.4 (58.2%)	40.8 (55.6%)	61.5 (33.1%)	30.8%
SimCLR2	cars196	17.7	16.9 (4.6%)	15.6 (11.9%)	9.8 (44.5%)	1.6 (91.2%)	2.4 (86.7%)	14.3 (19.1%)	43.0%
SimCLR2	aircraft70	31.2	32.0 (-2.4%)	32.0 (-2.3%)	20.1 (35.7%)	7.1 (77.4%)	10.2 (67.2%)	26.7 (14.6%)	31.7%
BarlowTwins	cifar10	83.8	77.8 (7.1%)	70.0 (16.4%)	62.0 (26.0%)	51.6 (38.5%)	50.0 (40.3%)	56.9 (32.1%)	26.7%
BarlowTwins	cifar100	63.7	56.0 (12.1%)	48.1 (24.5%)	38.8 (39.2%)	26.5 (58.5%)	26.6 (58.2%)	34.2 (46.3%)	39.8%
BarlowTwins	stl10	94.5	91.6 (3.0%)	83.7 (11.4%)	74.6 (21.1%)	40.0 (57.7%)	44.8 (52.6%)	63.7 (32.6%)	29.7%
BarlowTwins	cars196	23.4	23.6 (-1.1%)	20.7 (11.4%)	11.8 (49.6%)	2.7 (88.4%)	2.4 (89.8%)	18.7 (19.9%)	43.0%
BarlowTwins	aircraft70	39.2	43.1 (-9.7%)	40.4 (-2.9%)	22.5 (42.7%)	7.7 (80.3%)	9.4 (76.1%)	31.4 (19.9%)	34.4%
DeepCluster	cifar10	87.2	80.5 (7.7%)	70.6 (19.0%)	64.3 (26.2%)	52.5 (39.7%)	50.3 (42.3%)	57.3 (34.3%)	28.2%
DeepCluster	cifar100	65.0	56.2 (13.6%)	47.3 (27.1%)	39.6 (39.1%)	27.7 (57.4%)	25.6 (60.6%)	33.5 (48.5%)	41.1%
DeepCluster	stl10	94.8	92.4 (2.6%)	84.6 (10.8%)	79.2 (16.5%)	41.9 (55.8%)	45.0 (52.6%)	64.0 (32.5%)	28.5%
DeepCluster	cars196	22.7	20.9 (7.8%)	19.3 (15.0%)	13.8 (39.3%)	2.9 (87.2%)	3.8 (83.2%)	16.7 (26.4%)	43.2%
DeepCluster	aircraft70	40.3	39.2 (2.8%)	37.6 (6.7%)	24.4 (39.5%)	7.3 (81.8%)	11.2 (72.2%)	27.9 (30.7%)	38.9%
SwAV	cifar10	83.5	76.8 (8.1%)	68.8 (17.7%)	63.3 (24.1%)	52.9 (36.6%)	48.6 (41.8%)	55.3 (33.8%)	27.0%
SwAV	cifar100	60.1	52.5 (12.7%)	44.3 (26.3%)	38.5 (35.9%)	27.1 (55.0%)	23.8 (60.4%)	31.0 (48.4%)	39.8%
SwAV	stl10	94.4	91.8 (2.8%)	84.0 (11.1%)	80.3 (14.9%)	43.0 (54.5%)	44.5 (52.8%)	62.7 (33.6%)	28.3%
SwAV	cars196	17.2	16.2 (5.8%)	14.6 (15.0%)	12.0 (30.4%)	3.0 (82.8%)	3.0 (82.5%)	12.6 (26.9%)	40.6%
SwAV	aircraft70	31.5	29.7 (5.6%)	30.5 (3.0%)	23.9 (24.2%)	8.3 (73.7%)	10.3 (67.2%)	22.1 (29.7%)	33.9%
SwAV-b	cifar10	84.7	78.0 (7.9%)	70.1 (17.2%)	63.4 (25.1%)	52.6 (37.8%)	51.0 (39.8%)	56.7 (33.0%)	26.8%
SwAV-b	cifar100	62.7	54.4 (13.2%)	46.5 (25.8%)	39.7 (36.6%)	28.0 (55.3%)	26.5 (57.8%)	33.2 (47.1%)	39.3%
SwAV-b	stl10	94.3	91.6 (2.9%)	83.7 (11.3%)	78.5 (16.8%)	44.6 (52.8%)	45.0 (52.4%)	60.9 (35.4%)	28.6%
SwAV-b	cars196	19.3	18.0 (6.8%)	16.4 (15.2%)	12.4 (36.1%)	3.1 (84.2%)	3.5 (82.0%)	14.7 (24.2%)	41.4%
SwAV-b	aircraft70	33.5	32.8 (2.1%)	30.5 (9.0%)	21.7 (35.2%)	8.0 (76.0%)	11.0 (67.2%)	23.9 (28.7%)	36.3%

Table D.5: Robustness to downstream data corruption with *fine-tuning*. We fine-tune the full network and linear classification layer for 10 epochs. Overall, CL methods can be more robust than Sup under this setting except for BarlowTwins.

Pre-train Alg	Dataset	Orig	$\gamma = 0.2$	$\gamma = 5$	G-small	G-large	L-small	L-large	Avg Δ
Sup	cifar10	96.7	96.8 (-0.2%)	94.0 (2.8%)	88.2 (8.8%)	77.5 (19.8%)	72.0 (25.5%)	86.0 (11.0%)	11.3%
Sup	cifar100	83.8	83.7 (0.1%)	77.4 (7.6%)	68.8 (17.9%)	53.5 (36.2%)	46.0 (45.1%)	65.2 (22.2%)	21.5%
Sup	stl10	97.7	97.2 (0.6%)	92.5 (5.4%)	92.1 (5.7%)	55.5 (43.2%)	56.5 (42.2%)	89.1 (8.8%)	17.6%
Sup	cars196	75.1	73.2 (2.6%)	58.4 (22.3%)	40.1 (46.5%)	4.1 (94.6%)	5.0 (93.4%)	56.5 (24.7%)	47.3%
Sup	aircraft70	81.5	80.4 (1.3%)	78.8 (3.3%)	66.5 (18.4%)	11.5 (85.9%)	17.6 (78.4%)	72.8 (10.6%)	33.0%
BYOL	cifar10	96.5	96.3 (0.2%)	93.9 (2.7%)	88.8 (7.9%)	80.2 (16.9%)	75.2 (22.0%)	87.4 (9.4%)	9.8%
BYOL	cifar100	83.2	82.2 (1.2%)	76.7 (7.8%)	68.3 (17.9%)	54.2 (34.8%)	46.5 (44.1%)	64.4 (22.6%)	21.4%
BYOL	stl10	96.2	95.8 (0.5%)	91.8 (4.7%)	91.3 (5.2%)	57.0 (40.8%)	56.2 (41.6%)	88.2 (8.4%)	16.9%
BYOL	cars196	80.4	77.2 (4.0%)	62.1 (22.8%)	49.0 (39.1%)	2.8 (96.6%)	3.9 (95.2%)	65.1 (19.0%)	46.1%
BYOL	aircraft70	87.7	86.5 (1.4%)	84.1 (4.2%)	76.3 (13.0%)	13.9 (84.1%)	20.2 (76.9%)	80.0 (8.8%)	31.4%
SimSiam	cifar10	95.0	95.1 (-0.1%)	92.1 (3.1%)	87.5 (7.9%)	79.8 (16.1%)	75.4 (20.7%)	86.7 (8.8%)	9.4%
SimSiam	cifar100	81.0	80.5 (0.6%)	74.1 (8.5%)	68.5 (15.5%)	56.4 (30.4%)	51.3 (36.6%)	67.1 (17.1%)	18.1%
SimSiam	stl10	94.0	93.4 (0.6%)	88.1 (6.3%)	87.1 (7.3%)	64.4 (31.5%)	59.5 (36.7%)	85.8 (8.7%)	15.2%
SimSiam	cars196	85.7	85.2 (0.6%)	75.7 (11.6%)	64.4 (24.9%)	4.2 (95.1%)	5.9 (93.1%)	79.3 (7.5%)	38.8%
SimSiam	aircraft70	89.7	89.0 (0.8%)	86.9 (3.2%)	82.3 (8.3%)	23.5 (73.8%)	28.9 (67.7%)	86.4 (3.7%)	26.3%
MoCo-b	cifar10	96.8	96.7 (0.2%)	94.5 (2.4%)	89.6 (7.5%)	81.5 (15.9%)	77.4 (20.1%)	89.4 (7.7%)	9.0%
MoCo-b	cifar100	84.8	84.1 (0.9%)	78.5 (7.5%)	72.2 (14.8%)	59.0 (30.5%)	53.9 (36.4%)	70.8 (16.5%)	17.8%
MoCo-b	stl10	96.3	96.3 (0.0%)	91.8 (4.6%)	91.6 (4.9%)	64.3 (33.2%)	61.0 (36.7%)	90.3 (6.2%)	14.3%
MoCo-b	cars196	85.7	84.6 (1.3%)	75.7 (11.7%)	62.8 (26.7%)	3.6 (95.8%)	5.0 (94.2%)	78.5 (8.3%)	39.7%
MoCo-b	aircraft70	90.3	89.3 (1.1%)	88.0 (2.5%)	82.5 (8.6%)	22.6 (75.0%)	27.2 (69.9%)	86.9 (3.8%)	26.8%
SimCLR2	cifar10	96.3	95.8 (0.5%)	93.3 (3.1%)	87.1 (9.6%)	76.8 (20.3%)	72.2 (25.0%)	86.2 (10.5%)	11.5%
SimCLR2	cifar100	84.8	84.2 (0.7%)	78.6 (7.3%)	69.5 (18.1%)	56.7 (33.2%)	51.4 (39.4%)	67.3 (20.7%)	19.9%
SimCLR2	stl10	95.5	95.2 (0.3%)	89.7 (6.0%)	86.5 (9.4%)	54.6 (42.8%)	55.8 (41.6%)	88.0 (7.8%)	18.0%
SimCLR2	cars196	77.9	75.3 (3.4%)	64.9 (16.8%)	47.0 (39.6%)	3.0 (96.2%)	4.5 (94.3%)	68.1 (12.6%)	43.8%
SimCLR2	aircraft70	84.8	83.8 (1.1%)	82.9 (2.2%)	72.5 (14.5%)	20.1 (76.3%)	23.8 (72.0%)	79.4 (6.3%)	28.7%
BarlowTwins	cifar10	96.8	96.7 (0.1%)	94.4 (2.5%)	87.9 (9.2%)	76.4 (21.0%)	70.1 (27.6%)	84.9 (12.3%)	12.1%
BarlowTwins	cifar100	83.9	83.6 (0.4%)	76.9 (8.4%)	64.2 (23.5%)	46.1 (45.1%)	39.0 (53.5%)	56.4 (32.8%)	27.2%
BarlowTwins	stl10	97.3	96.8 (0.6%)	92.2 (5.2%)	91.2 (6.3%)	52.9 (45.7%)	52.0 (46.6%)	87.1 (10.5%)	19.1%
BarlowTwins	cars196	73.5	69.0 (6.3%)	53.2 (27.7%)	38.0 (48.3%)	2.7 (96.4%)	3.4 (95.3%)	57.1 (22.4%)	49.4%
BarlowTwins	aircraft70	81.1	77.9 (4.0%)	76.5 (5.7%)	63.8 (21.3%)	11.3 (86.0%)	15.5 (80.9%)	67.7 (16.5%)	35.7%
DeepCluster2-b	cifar10	96.5	96.5 (0.0%)	94.6 (2.0%)	89.9 (6.9%)	80.8 (16.3%)	75.7 (21.6%)	87.7 (9.2%)	9.3%
DeepCluster2-b	cifar100	84.7	83.6 (1.3%)	78.3 (7.5%)	71.6 (15.4%)	57.6 (32.0%)	49.2 (41.9%)	66.8 (21.1%)	19.9%
DeepCluster2-b	stl10	96.8	96.3 (0.4%)	93.7 (3.2%)	93.1 (3.8%)	62.3 (35.6%)	57.6 (40.4%)	88.9 (8.1%)	15.3%
DeepCluster2-b	cars196	81.6	79.4 (2.6%)	68.6 (16.0%)	56.3 (31.0%)	3.4 (95.8%)	4.9 (94.0%)	66.5 (18.5%)	43.0%
DeepCluster2-b	aircraft70	87.9	87.2 (0.8%)	85.4 (2.9%)	77.8 (11.5%)	15.4 (82.5%)	20.0 (77.2%)	79.1 (10.1%)	30.8%
SwAV-b	cifar10	96.3	96.4 (-0.2%)	94.0 (2.3%)	89.8 (6.7%)	81.6 (15.2%)	75.9 (21.2%)	87.8 (8.7%)	9.0%
SwAV-b	cifar100	83.7	83.1 (0.7%)	77.4 (7.5%)	70.8 (15.4%)	58.1 (30.5%)	49.7 (40.6%)	66.3 (20.8%)	19.3%
SwAV-b	stl10	96.3	96.6 (-0.3%)	92.8 (3.7%)	92.7 (3.8%)	63.2 (34.3%)	58.9 (38.9%)	88.6 (8.0%)	14.7%
SwAV-b	cars196	82.2	80.1 (2.5%)	70.5 (14.2%)	60.4 (26.5%)	3.8 (95.4%)	5.4 (93.4%)	67.7 (17.6%)	41.6%
SwAV-b	aircraft70	89.2	88.2 (1.2%)	87.2 (2.3%)	80.0 (10.4%)	18.2 (79.6%)	22.9 (74.3%)	81.3 (8.9%)	29.5%

D.7. Pre-train on corrupted CIFAR-10, but test on uncorrupted images

In the main paper, we show the results when both the pre-training and evaluation datasets are corrupted in the same consistent way. In the following Table D.7, we report the accuracy numbers obtained from KNN evaluation on the original uncorrupted images. Since these models are pre-trained on the pixel or patch-level corrupted dataset, the results reflect the transfer capability of the pre-trained representation from corrupted data to original data. We find that the trend is similar to evaluating on corrupted data that Sup appears more robust.

Table D.7: Uncorrupted evaluation results of robustness to pre-training pixel-level gamma distortion and patch-level corruption (global and local shuffling) with CIFAR-10 and ResNet-18.

Method	Orig	$\gamma = 0.2$	G4x4	G8x8	L4x4	L8x8	Avg Δ
Supervised	92.23	82.72	63.03	36.94	61.56	62.51	-
Δ	-	10.31%	31.66%	59.95%	33.25%	32.22%	33.48%
MoCo-v2 KNN Eval	82.55	72.01	46.66	32.93	48.91	53.78	-
Δ	-	15.40%	43.48%	60.11%	40.75%	34.85%	38.39%

E. Additional Visualization

E.1. Visualizing corrupted images

Please check Figure E.4 for more visual examples of the pixel-level gamma distortion and patch-level shuffling corruptions we used.

E.2. Visualizing Grad-CAM attention maps

Figure E.5 visualizes the Grad-CAM (Selvaraju et al., 2017) attentions maps of ResNet-18 models pre-trained and linearly fine-tuned on either uncorrupted or 4x4 global patch shuffled images. We discover some difference in terms of the *equivariant* property: Sup models are largely equivariant to 4x4 global patch shuffling – the attention is focused on the object parts even after patch shuffling, whereas the MoCo model pre-trained on 4x4 global shuffled images are not – it is rather focused on distracting parts. The quality of attention map correlates with the top-1 validation accuracy, where Sup on 4x4 achieves 65% and MoCo achieves 35%. Intuitively, a model can be more robust to such global patch shuffling if it possesses such equivariant property. This shows the robustness of SL from another aspect, because it can robustly learn the same feature even under the shuffling disturbance.

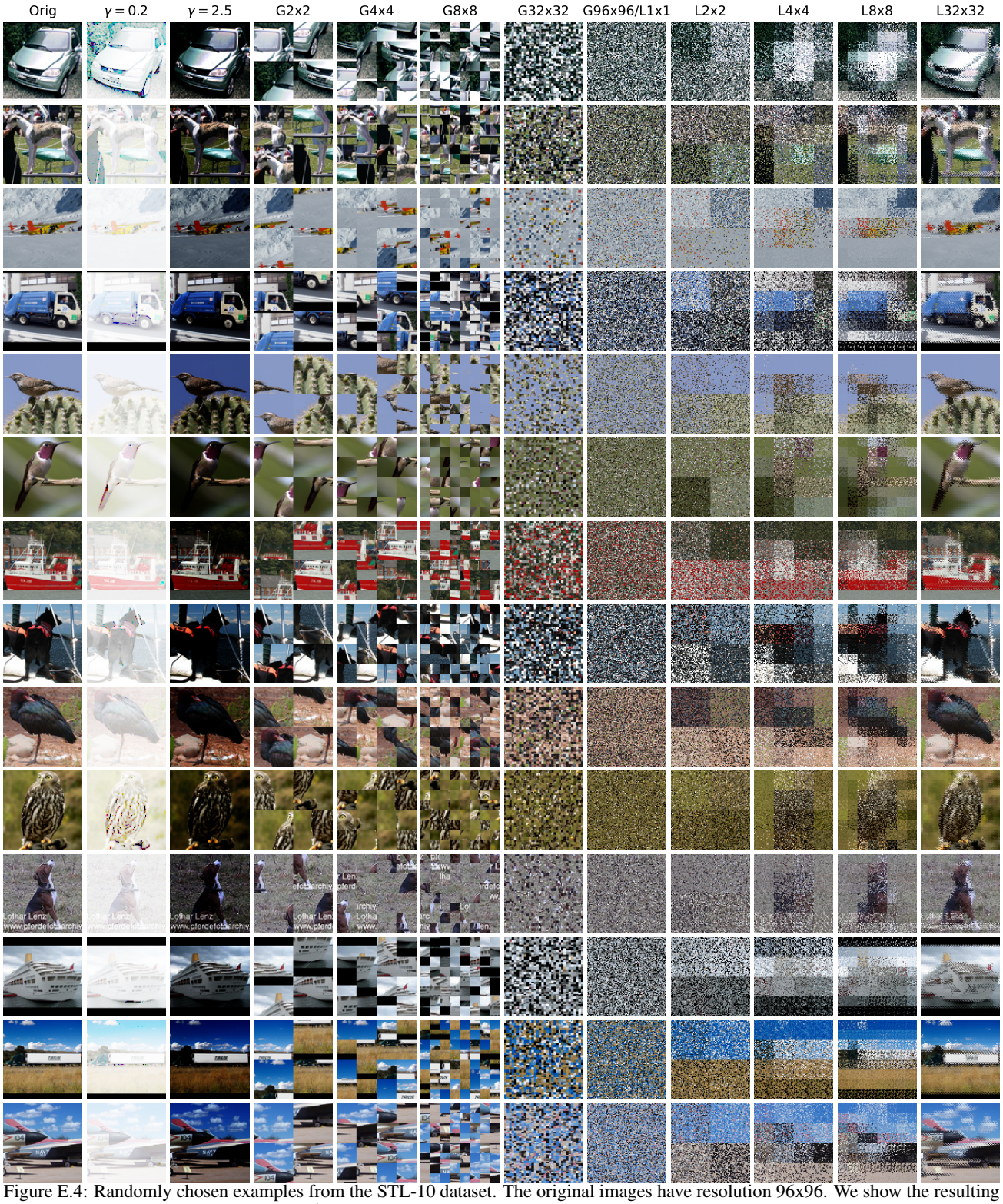

Figure E.4: Randomly chosen examples from the STL-10 dataset. The original images have resolution 96x96. We show the resulting images of gamma distortion ($\gamma = 0.2, 2.5$), global shuffling (G2x2, weaker – G96x96, stronger), and local shuffling (L1x1, stronger – L32x32, weaker). G1x1 and L96x96 revert to the original, while G96x96 and L1x1 are the most random ones (and have similar effect). Gamma distortion reduces information in pixel intensity. Global shuffling destroys global but preserves local structure, Local shuffling is the opposite.

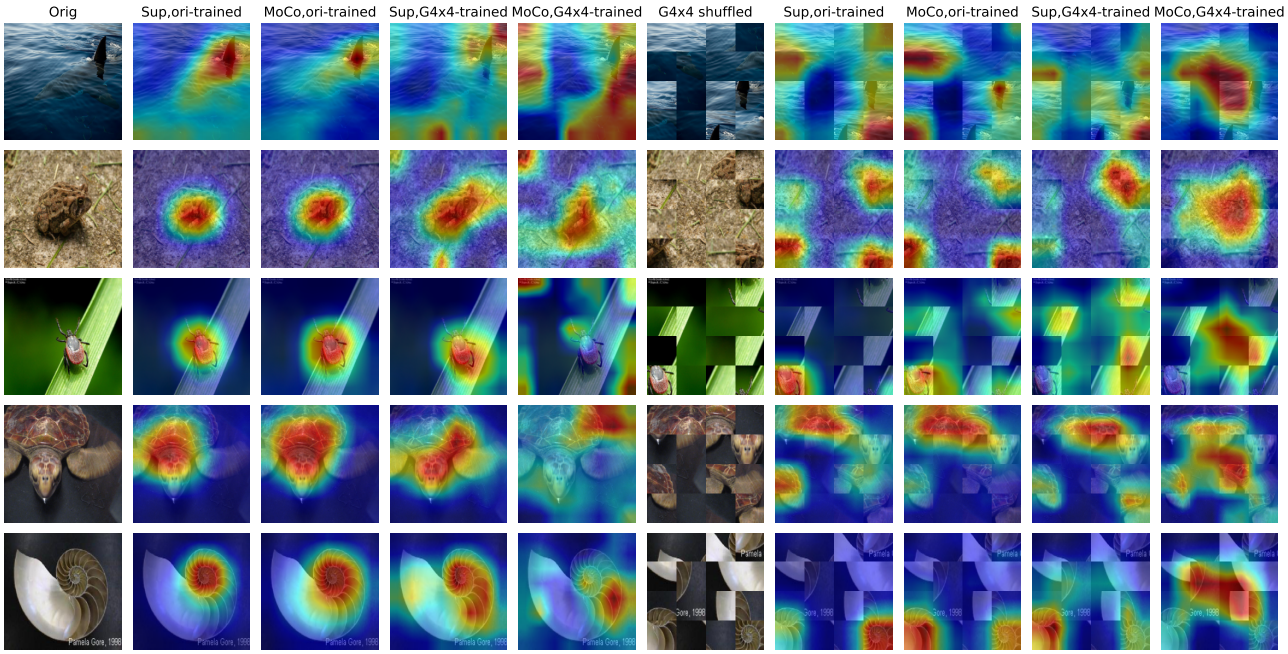

Figure E.5: Randomly chosen images from ImageNet-100. We consider 4x4 global patch shuffling and visualize the Grad-CAM attention maps of 4 models: Sup trained on original images, MoCo trained on original images, Sup trained on shuffled images, and MoCo trained on shuffled images. The attention map of the MoCo on shuffled images model is less equivariant to the patch shuffling.