PRIME: A Few Primitives Can Boost Robustness to Common Corruptions

Apostolos Modas∗†
apostolos.modas@epfl.ch

Rahul Rade∗‡
rarade@ethz.ch

Guillermo Ortiz-Jiménez†
guillermo.ortizjimenez@epfl.ch

Seyed-Mohsen Moosavi-Dezfooli‡
seyed.moosavi@inf.ethz.ch

Pascal Frossard†
pascal.frossard@epfl.ch

Abstract

Despite their impressive performance on image classification tasks, deep networks have a hard time generalizing to many common corruptions of their data. To fix this vulnerability, prior works have mostly focused on increasing the complexity of their training pipelines, combining multiple methods, in the name of diversity. However, in this work, we take a step back and follow a principled approach to achieve robustness to common corruptions. We propose PRIME, a general data augmentation scheme that consists of simple families of max-entropy image transformations. We show that PRIME outperforms the prior art for corruption robustness, while its simplicity and plug-and-play nature enables it to be combined with other methods to further boost their robustness. Furthermore, we analyze PRIME to shed light on the importance of the mixing strategy on synthesizing corrupted images, and to reveal the robustness-accuracy trade-offs arising in the context of common corruptions. Finally, we show that the computational efficiency of our method allows it to be easily used in both on-line and off-line data augmentation schemes1.

1. Introduction

Deep image classifiers do not work well in the presence of various types of distribution shifts [14, 18, 41]. Most notably, their performance can severely drop when the input images are affected by common distortions that are not present in the training set, such as digital artefacts, low contrast, or blurs [21, 29]. Yet, building classifiers that are robust to common corruptions is far from trivial. A naive solution is to include data with all sorts of corruptions during training, but the sheer scale of all possible types of naturally occurring perturbations that might affect an image is simply too large. Hence, the computational cost would render such a solution virtually impossible. Besides, the problem is per se ill-defined since there exists no formal description of all possible common corruptions.

To overcome this issue, the research community has recently favoured increasing the “diversity” of the training data via data augmentation schemes [11, 20, 22]. Intuitively, the hope is that showing very diverse augmentations of an image to a network would increase the chance that the latter becomes invariant to some common corruptions. Still, guaranteeing a good coverage over the whole space of such corruptions is hard. Hence, the current literature relies mostly on increasing the complexity of their training pipelines by combining independently developed data augmentation strategies, to hopefully increase diversity of augmentations and achieve state-of-the-art results on different common corruptions benchmarks [6, 7, 20, 24, 37, 43]. This incremental research strategy, though, has come at a cost.

Today, it is hard to pinpoint which elements of these methods meaningfully contribute to the overall robustness, and which ones have only a marginal benefit. Meanwhile, the increased complexity of some of the recent methods has made them impractical to use in many large-scale tasks. While in many occasions, these methods have been tailored to particular datasets and might not be general enough. Yet, the problem of building robust classifiers remains far from completely solved, and the gap between robustness

*The first two authors contributed equally to this work.
†Ecole Polytechnique Fédérale de Lausanne (EPFL).
‡ETH Zürich.

1The code of PRIME can be found at https://github.com/amodas/PRIME-augmentations
and standard accuracy is still large.

In this work, we take a step back, and aim to identify a systematic way for designing simple, yet effective data augmentation schemes to improve robustness to common corruptions. To do so, we formulate a new mathematical model for semantically-preserving natural corruptions, and build on basic concepts to characterize the notions of transformation strength and diversity using a few transformation primitives. Relying on this model, we propose PRIME, a new data augmentation scheme that samples transformations from a max-entropy distribution to effectively increase the coverage over the space of possible distortions (see Fig. 1). The performance of PRIME, alone, already tops the current baselines on different common corruption datasets, whilst it can also be combined with other methods to further boost their performance. The simplicity and flexibility of PRIME also make it an effective tool for understanding common corruption robustness, which can eventually lead to deep classifiers with better out-of-distribution generalization properties.

Altogether, the main contributions of our work include:

- We define a mathematical model based on a few primitives to characterize the notion of common corruptions.
- We introduce PRIME, a simple and efficient method to boost robustness to common corruptions.
- We experimentally show that PRIME, despite its simplicity, achieves high robustness on multiple common corruption benchmarks, while it can be easily integrated with other techniques to further boost performance. To the best of our knowledge, (currently) PRIME outperforms the strongest competing methods applied on the same convolutional architectures [1,10].
- Lastly, we use PRIME to shed light on multiple properties concerning the robustness to common corruptions, such as the role of mixing, the robustness-accuracy trade-off, and the importance of online augmentations.

Overall, we see our foundational work as an important step in the race for robustness against common corruptions. In this sense, we believe that PRIME has the potential to become the new baseline for learning robust classifiers, while it can also serve as one of the building blocks for effective data augmentation strategies.

**2. General model of visual corruptions**

“Common corruptions” is an umbrella term coined to describe the set of all possible distortions that can happen to natural images during their acquisition, storage, and processing lifetime, and these can be very diverse. Nevertheless, while the space of possible perturbations is big, the term “common corruptions” is generally used to refer to image transformations that, while degrading the quality of the images, still preserve their semantic information.

In this work, motivated by the “semantically-preserving” nature of common corruptions, we define a new model of natural distortions. Specifically, we leverage the long tradition of image processing in developing techniques to manipulate images while retaining their semantics, and construct a principled framework to mathematically characterize the space of natural-looking corruptions.

Let \( x : [0, 1]^2 \to [0, 1]^3 \) be a continuous image\(^2\) mapping pixel coordinates \( r = (r_1, r_2) \) to RGB values. We define our model of common corruptions as the action of the following additive subgroup of the near-ring of transformations [5] on \( x \) given by

\[
\mathcal{T}_x = \left\{ \sum_{i=1}^n \mu_i \ g_i \circ \cdots \circ g_{m_i}(x) : g_{i} \in \{\omega, \tau, \gamma\}, \mu_i \in \mathbb{R} \right\},
\]

(1)

where \( \omega, \tau \) and \( \gamma \) are random primitive transformations which distort \( x \) along different domains. Namely, the spectral (\( \omega \)), spatial (\( \tau \)), and color (\( \gamma \)) domain. As we will see, defining each of these primitives in a principled and coherent fashion will be enough to construct a set of perturbations which covers most types of visual corruptions.

All our fundamental primitives are designed based on two guiding principles: maximum entropy and smoothness. First, the principle of maximum entropy [9] suggests using distributions of functions which are as unbiased as possible given their constraints. This formalizes the concept of sampling efficiency, and it guarantees that with each new augmentation we provide the network with the maximum additional information possible about the domain of the transformation. In Appendix A, we provide a formal derivation of each maximum entropy distribution. Then, the principle of smoothness formalizes the notion of physical plausibility, as most naturally occurring processes are smooth.

In what follows, we describe each of our basic primitives in more detail. Note that, however, in general, all the distributions of transformations that we define are governed by two parameters: \( K \) to control smoothness, and \( \sigma^2 \) to control strength. Specifically, all our transformations fall back to identity mappings when \( \sigma^2 = 0 \), independently of \( K \).

**Spectral domain** We parameterize the distribution of random spectral transformations using random filters \( \omega(r) \), such that the output of the transformation follows

\[
\omega(x)(r) = (x \ast (\delta + \omega'))(r),
\]

(2)

\(^2\)We define our model of common corruptions in the continuous domain for simplicity. However, as is common in image processing, in practice we will work with discrete images on a regular grid.
where, $*$ is the convolution operator, $\delta(r)$ represents a Dirac delta, i.e., identity filter, and $\omega'(r)$ is implemented in the discrete grid as a finite impulse response (FIR) filter of size $K_\omega \times K_\omega$ with i.i.d random entries distributed according to $\mathcal{N}(0, \sigma_\omega^2)$. Here, $\sigma_\omega^2$ governs the transformation strength, while larger $K_\omega$ yields filters of higher spectral resolution. The bias $\delta(r)$ makes the output close to the original image.

**Spatial domain** We use the following model to define our distribution of random spatial transformations, which apply random perturbations over the coordinates of an image, i.e.,

$$\tau(x)(r) = x(r + \tau'(r))$$

This model has been recently proposed by Petrini et al. [34] to define a distribution of random smooth diffeomorphisms in order to study the stability of neural networks to small spatial transformations. To guarantee smoothness but preserve maximum entropy, Petrini et al. propose to parameterize the vector field $\tau'$ as

$$\tau'(r) = \sum_{i^2+j^2\leq K_\tau^2} \beta_{i,j} \sin(\pi i r_1) \sin(\pi j r_2),$$

where $\beta_{i,j} \sim \mathcal{N}(0, \sigma_\tau^2/(i^2+j^2))$. This choice of values guarantees that the resulting mapping is smooth according to the cut frequency $K_\tau$, while $\sigma_\tau^2$ determines its strength.

**Color domain** We follow a similar approach to define the distribution of random color transformations. That is, we build random mappings $\gamma$ between color spaces such that

$$\gamma(x)(r) = x(r) + \sum_{n=0}^{K_\gamma} \beta_n \odot \sin(\pi n x(r)),$$

where $\beta_n \sim \mathcal{N}(0, \sigma_\gamma^2 I_3)$, with $\odot$ denoting elementwise multiplication. Again, $K_\gamma$ controls the smoothness of the transformations and $\sigma_\gamma^2$ their strength. Note however that, compared to Eq. (4), the coefficients in Eq. (5) are not weighted by the inverse of the frequency, and have constant variance. In practice, we observe that reducing the variance of the coefficients for higher frequencies creates color mappings that are too smooth and almost imperceptible, so we decided to drop this dependency in our model.

Finally, note that, since most benchmarks of visual corruptions disallow the use of additive perturbations during training [21], our model does not include an additive perturbation category. Nevertheless, for the sake of completeness, and depending on the application, one may also want to define an extra set of max-entropy transformations given by $\eta(x)(r) = x(r) + \eta'(r)$, where $\eta'(r) \sim \mathcal{N}(0, \sigma_\eta^2)$, and which could be readily integrated in Eq. (1).

Overall, our model is flexible and covers a large part of the space of natural-looking distortions. Also, it allows to easily control the strength and style of the transformations with just a few parameters. For these reasons, in Sec. 3, we use this model to design an efficient augmentation scheme to build classifiers robust to common corruptions.

### Algorithm 1: PRIME

| Input: Image $x$, primitives $\mathcal{G} = \{\text{Id}, \omega, \tau, \gamma\}$ |
|-----------------------------------------------|
| Output: Augmented image $\tilde{x}$ |
| 1 $\tilde{x}_0 \leftarrow x$ |
| 2 for $i \in \{1, \ldots, n\}$ do |
| 3 $\tilde{x}_i \leftarrow x$ |
| 4 for $j \in \{1, \ldots, m\}$ do |
| 5 $g \sim \mathcal{U}(\mathcal{G})$ $\triangleright$ Strength $\sigma \sim \mathcal{U}(\sigma_{\text{min}}, \sigma_{\text{max}})$ |
| 6 $\tilde{x}_i \leftarrow g(\tilde{x}_i)$ |
| 7 end |
| 8 end |
| 9 $\mu \sim \text{Dir}(1)$ $\triangleright$ Random Dirichlet convex coefficients |
| 10 $\tilde{x} \leftarrow \sum_{i=0}^{n} \mu_i \tilde{x}_i$ |

### Figure 2. Example images generated with the basic transformations of our common corruptions model. Despite the perceptibility of the introduced distortion, the image semantics are preserved.

### 3. PRIME: A new augmentation scheme

We now introduce PRIME, a simple yet efficient augmentation scheme that uses our PRImitives of Maximum Entropy to confer robustness against common corruptions. The pseudo-code of PRIME is given in Algorithm 1, which draws a random sample from Eq. (1) using a convex combination of a composition of basic primitives. Below we describe the main implementation details of our algorithm.

#### Parameter selection

It is important to ensure that the semantic information of the image is preserved after it goes through PRIME. As measuring semantic preservation quantitatively is not simple, we subjectively select the parameters of each primitive based on visual inspection, ensuring maximum permissible distortion while retaining the semantic content of the images. However, to avoid relying on a specific strength for each transformation, PRIME generates
augmentations of different strengths by sampling $\sigma$ from a uniform distribution, with different minimum and maximum values for each primitive. Figure 2 shows a few visual examples illustrating each kind of transformation, while additional visual examples along with the details of all the parameters can be found in Appendix B.

Note that for the color primitive, we observed that using fairly large values for $K_\gamma$ (in the order of 500) is important for covering a large space of visual distortions. Unfortunately, implementing such a transformation can be memory inefficient. To avoid this issue, PRIME uses a slight modification of Eq. (5), and combines a fixed number of consecutive frequencies randomly chosen in the range $[0, K_\gamma]$. Finally, as some of our transformations can push the images outside of their color range, we always clip the output of each transformation so that it lies on $[0, 1]^3$.

### Mixing transformations

The concept of mixing has been a recurring theme in the augmentation literature [22, 43, 46, 47] and PRIME follows this same trend. In particular, Algorithm 1 uses a convex combination of $n$ basic augmentations consisting of the composition of $m$ of our primitive transformations. This process is formally equivalent to the algorithmic mixing from AugMix [22]. In general, the convex mixing procedure serves two purposes: (i) it broadens the set of possible training augmentations, (ii) it ensures that the augmented images stay close to the original image. We later provide empirical results which underline the efficacy of mixing in Sec. 5.2. Overall, the exact mixing parameters are provided in Appendix B.

### 4. Performance analysis

In this section, we compare the classification performance of our method on multiple datasets, with that of two current approaches: AugMix and DeepAugment (DA). We illustrate that PRIME significantly advances the corruption robustness over that of AugMix and also surpasses DeepAugment on all the benchmarks. We also show that our method yields additional benefits when employed in concert with domain adaptation [39].

### 4.1. Training setup

We consider four datasets: CIFAR-10 (C-10), CIFAR-100 (C-100) [25], ImageNet-100 (IN-100) and ImageNet (IN) [12]. IN-100 is a 100-class subset of IN obtained by selecting every 10th class in WordNet ID order. We train ResNet-18 [19] on C-10, C-100 and IN-100; and ResNet-50 on IN. Following AugMix, and for a complete comparison, we also integrate the Jensen-Shannon divergence (JSD)-based consistency loss with PRIME which compels the network to learn similar representations for differently augmented versions of the same input image. Note that all the models are trained for 100 epochs. Detailed training setup appears in Appendix C. We evaluate our trained models on the common corrupted versions (C-10-C, C-100-C, IN-100-C, IN-C) of the aforementioned datasets. The common corruptions [21] constitute 15 natural image distortions each applied with 5 different severity levels. At a higher level, these corruptions can be grouped into four categories, viz. noise, blur, weather and digital.

### 4.2. Robustness to common corruptions

In order to assess the effectiveness of PRIME, we evaluate its performance against C-10, C-100, IN-100 and IN common corruptions. The results are summarized in Tab. 1. Amongst individual methods, PRIME yields superior results compared to those obtained by AugMix and DeepAugment alone, and advances the baseline performance on the corrupted counterparts of all the four datasets. As listed, AugMix already has a good performance on C-10-C and C-100-C. Yet, PRIME pushes the corruption accuracy by 0.4% and 3.4% respectively over AugMix, albeit with a slight degradation in clean accuracy. On IN-100-C, a more complicated dataset, we observe significant improvements wherein PRIME outperforms AugMix by 10.9%. In fact, this increase in performance hints that our primitive transformations are actually able to cover a larger space of image corruptions.

| Dataset | Method  | Clean Acc | Common corruption Acc | mCE |
|---------|---------|-----------|-----------------------|-----|
| C-10    | Standard | 95.0      | 74.0                  | 24.0|
|         | AugMix  | 95.2      | 88.6                  | 11.4|
|         | PRIME   | 93.1      | 89.0                  | 11.0|
| C-100   | Standard | 76.7      | 51.9                  | 48.1|
|         | AugMix  | 78.2      | 64.9                  | 35.1|
|         | PRIME   | 77.6      | 68.3                  | 31.7|
| IN-100  | Standard | 88.0      | 49.7                  | 100.0|
|         | AugMix  | 88.7      | 60.7                  | 79.1|
|         | DA      | 86.3      | 67.7                  | 68.1|
|         | PRIME   | 85.9      | 71.6                  | 61.0|
|         | DA+AugMix | 86.5      | 73.1                  | 57.3|
|         | DA+PRIME | 84.9      | 74.9                  | 54.6|
| IN      | Standard | 76.1      | 38.1                  | 76.7|
|         | AugMix  | 77.5      | 48.3                  | 65.3|
|         | DA      | 76.7      | 52.6                  | 60.4|
|         | PRIME   | 77.0      | 55.0                  | 57.5|
|         | DA+AugMix | 75.8      | 58.1                  | 53.6|
|         | DA+PRIME | 75.5      | 59.9                  | 51.3|

Table 1. Clean and corruption accuracy, and mean corruption error (mCE) for different methods with ResNet-18 on C-10, C-100, IN-100 and ResNet-50 on IN. mCE is the mean corruption error on common corruptions un-normalized for C-10 and C-100; normalized relative to standard model on IN-100 and IN. † indicates that JSD consistency loss is not used. * Models taken from [10].
corruptions, compared to the restricted set of AugMix. Interestingly, the random transformations in PRIME also lead to a 3.9% boost in corrections accuracy over DeepAugment despite the fact that DeepAugment leverages additional knowledge to augment the training data via its use of pre-trained architectures. Moreover, PRIME provides cumulative gains when combined with DeepAugment, reducing the mean corruption error (mCE) of prior art (DA+AugMix) by 2.7% on IN-100-C. Lastly, we also evaluate the performance of PRIME on full IN-C. However, we do not use JSD with PRIME in order to reduce computational complexity. Yet, even without the JSD loss, PRIME outperforms, in terms of corruption accuracy, both AugMix (with JSD) and DeepAugment by 6.7% and 2.4% respectively, while the mCE is reduced by 7.8% and 2.9%. And last, when PRIME is combined with DeepAugment, it also surpasses the performance of DA+AugMix (with JSD), reaching a corruption accuracy of almost 60% and an mCE of 51.3%.

Additionally, given the nuances amongst individual corruption types in common corruptions, we perform a fine-grained analysis with PRIME on IN-100-C to ensure that our method leads to general improvements against all corruption types. As evident from the comparison in Tab. 2, PRIME alone, even without the JSD term, improves robustness over current techniques to almost every corruption type except blur. Further, incorporating the JSD term with PRIME attains the best results on all the corruption categories. Relative to the previous best by DeepAugment on IN-100-C, PRIME improves by 4.3% on noises, 2% on blurs, 3.4% on weather changes and 5.8% against digital distortions. This validates that PRIME helps against all common corruption types in IN-100-C, underlining the generality of our model of common corruptions.

### 4.3. Unsupervised domain adaptation

Recently, robustness to common corruptions has also been of significant interest in the field of unsupervised domain adaptation [3, 39]. The main difference is that, in domain adaptation, one exploits the limited access to test-time corrupted samples to adjust certain network parameters. Hence, it would be interesting to investigate the utility of PRIME under the setting of domain adaption.

To that end, we combine our method with the adaptation trick by Schneider et al. [39]. Specifically, we adjust the batch normalization (BN) statistics of our models using a few corrupted samples. Suppose $z_s \in \{\mu_s, \sigma_s\}$ are the BN mean and variance estimated from the training data, and $z_t \in \{\mu_t, \sigma_t\}$ are the corresponding statistics computed from $n$ unlabelled, corrupted test samples, then we re-estimate the BN statistics as follows.

$$
\hat{z} = \frac{N}{N + n} z_s + \frac{n}{N + n} z_t
$$

(6)

We consider three adaptation scenarios: single sample ($n = 1, N = 16$), partial ($n = 8, N = 16$) and full ($n = 400, N = 0$) adaptation. Here, we do not perform parameter tuning for $N$. As shown in Tab. 3, simply correcting BN statistics using as little as 8 corrupted samples pushes the corruption accuracy of PRIME from 71.6% to 75.3%. In general, PRIME yields cumulative gains in combination with adaptation and has the best IN-100-C accuracy.

### 5. Robustness insights using PRIME

In this section, we exploit the simplicity and the controllable nature of PRIME to investigate different aspects behind robustness to common corruptions. We first analyze how each transformation domain contributes to the overall robustness of the network. Then, we empirically locate and justify the benefits of mixing the transformations of each domain. Moreover, we demonstrate the existence of a robustness-accuracy trade-off in the context of common corruptions. Finally, we comment on the low-complexity benefits of PRIME in different data augmentation settings.

| Method | IN-100-C (↑) | Noise (↑) | Blur (↑) | Weather (↑) | Digital (↑) | IN-100 (↑) |
|--------|-------------|-----------|----------|-------------|-------------|------------|
| AugMix | 55.2        | 38.9      | 56.8     | 57.0        | 64.2        | 88.0       |
| AugMix | 60.7        | 44.8      | 63.1     | 60.7        | 70.3        | **88.7**   |
| DA     | 67.7        | 75.9      | 62.5     | 63.6        | 70.9        | 86.3       |
| PRIME† | 68.8        | 78.8      | 58.3     | 66.0        | 74.8        | 87.1       |
| PRIME  | **71.6**    | **80.2**  | **64.5** | **67.0**    | **76.7**    | **85.9**   |

Table 2. Classification accuracy (↑) of various methods on different corruption types contained in IN-100-C. † indicates that JSD consistency loss is not used. Network used: ResNet-18.

| Method | w/o adapt | single adapt | partial adapt | full adapt | single adapt |
|--------|-----------|--------------|---------------|------------|--------------|
| Standard | 49.7 | 53.8 | 62.0 | 63.9 | 88.1 |
| AugMix | 60.7 | 65.5 | 71.3 | 73.0 | **88.3** |
| DA | 67.7 | 70.2 | 72.7 | 74.6 | 86.3 |
| PRIME | **71.6** | **73.5** | **75.3** | **76.6** | **85.7** |

Table 3. Performance of different methods in concert with domain adaptation on IN-100. Partial adaptation uses 8 samples; full adaptation uses 400 corrupted samples. Network used: ResNet-18.
against blur, weather and digital corruptions. Spatial operations also improve on blurs, but on elastic transforms as well (digital). On the contrary, color transformations excel on noises and certain high frequency digital distortions, e.g. pixelate and JPEG artefacts, and have a minor effect on weather changes. Besides, incrementally combining these transformations lead to cumulative gains e.g. spatial+color help on both noises and blurs. Yet, for obtaining the best results, the combination of all transformations is required. This means that each transformation increases the coverage over the space of possible distortions and the increase in robustness comes from their cumulative contribution.

### 5.2. The role of mixing

In most data augmentation methods, besides the importance of the transformations themselves, mixing has been claimed as an essential module for increasing diversity in the training process \cite{henderson2021learning, mandt2017stochastic, henderson2021generalization, yu2020constantly}. In our attempt to provide insights on the role of mixing in the context of common corruptions, we found out that it is capable of constructing augmented images that look perceptually similar to their corrupted counterparts. In fact, the improvements on specific corruption types observed in Tab. 4 can be largely attributed to mixing. As exemplified in Fig. 3a and 3b, careful combinations of spectral transformations with the clean image introduce brightness and contrast-like artefacts that look similar to the corresponding corruptions in IN-C. Also, combining spatial transformations creates blur-like artefacts that look identical to zoom blur in IN-C (Fig. 3d). Finally, notice in Fig. 3c how mixing color transformations helps fabricate corruptions of the “noise” category. This means that the max-entropy color model of PRIME enables robustness to different types of noise without explicitly adding any during training. This might explain the significant improvement over the “noise” category in Tab. 2.

The role of mixing helps on both noises and blurs. Yet, for obtaining the best results, the combination of all transformations is required. This means that each transformation increases the coverage over the space of possible distortions and the increase in robustness comes from their cumulative contribution.

| Transform | IN-100C | Noise | Blur | Weather | Digital | IN-100 |
|-----------|---------|-------|------|---------|---------|--------|
| None      | 49.7    | 27.3  | 48.6 | 54.8    | 62.6    | 88.0   |
| ω         | 64.1    | 60.7  | 55.4 | 66.6    | 72.9    | 87.3   |
| τ         | 53.8    | 30.1  | 56.2 | 57.6    | 65.4    | 87.0   |
| γ         | 59.9    | 67.4  | 52.6 | 54.4    | 67.1    | 86.9   |
| ω+τ       | 64.5    | 58.5  | 57.3 | 66.8    | 73.9    | 87.7   |
| ω+γ       | 67.5    | 77.2  | 55.7 | 65.3    | 74.2    | 87.1   |
| ω+τ+γ     | 63.3    | 74.7  | 57.4 | 56.2    | 67.8    | 86.2   |
| ω+τ+γ+γ   | 68.8    | 78.8  | 58.3 | 66.0    | 74.6    | 87.1   |

Table 4. Impact of the different primitives of max-entropy (ω: spectral, γ: color, τ: spatial) from PRIME on common corruption accuracy (↑). All the transformations are essential for the performance of PRIME. All values models are trained without JSD loss. Network used: ResNet-18.

Note that one of the main goals of data augmentation is to achieve maximum coverage of the space of possible distortions using a limited transformation budget, i.e., within a few training epochs. The principle of max-entropy guarantees this within each primitive, but the effect of mixing on the overall space is harder to quantify. In this regard, we can use the distance in the embedding space of a SimCLRv2 \cite{chen2020simple} model φ, as a proxy for visual similarity \cite{mohamed2014probabilistic, liu2021unsupervised}. We are interested in measuring how mixing the base transformations changes the likelihood that an augmentation scheme generates some sample during training that is visually similar to some of the common corruptions. To that end, we randomly select \( N = 1000 \) training images \( \{x_n\}_{n=1}^N \) from IN, along with their \( C = 75 \) (15 corruptions of 5 severity levels) associated common corruptions \( \{x_n^c\}_{c=1}^C \), and generate for each of the clean images another \( T = 100 \) transformed samples \( \{\hat{x}_n^t\}_{t=1}^T \) using each augmentation scheme. Moreover, for each corruption \( x_n^c \) we find its closest neighbor \( \hat{x}_n^t \) from the set of generated samples using the cosine distance in the embedding space\(^3\). Our overall measure of fitness is

\[
\frac{1}{NC} \sum_{n=1}^N \sum_{c=1}^C \min_{t} \left\{ 1 - \frac{\phi(\hat{x}_n^c)^\top \phi(\hat{x}_n^t)}{||\phi(\hat{x}_n^c)||_2 \cdot ||\phi(\hat{x}_n^t)||_2} \right\}.
\]

Table 5 shows the values of this measure applied to Aug-Mix and PRIME, with and without mixing\(^4\). For reference, we also report the values of the clean (no transform) images \( \{x_n\}_{n=1}^N \). Clearly, mixing helps reducing the distance between the common corruptions and the augmented samples from both methods. We also observe that PRIME, even with only 100 augmentations per image – in the order of number of training epochs – can generate samples which

\(^3\)Examples of nearest neighbors can be found in Appendix E

\(^4\)More percentile scores can be found in Appendix F
are twice as close to the common corruptions as AugMix. In fact, the feature similarity between training augmentations and test corruptions was also studied in [29], with an attempt to justify the good performance of AugMix on C-10. Yet, we see that the fundamental transformations of AugMix are not enough to span a broad space warranting high perceptual similarity to IN-C. The significant difference in terms of perceptual similarity in Tab. 5 between AugMix and PRIME may explain the superior performance of PRIME on IN-100-C and IN-C (cf. Tab. 1).

5.3. Robustness vs. accuracy trade-off

An important phenomenon observed in the literature of adversarial robustness is the so-called robustness-accuracy trade-off [16, 35, 42], where technically adversarial training [27] with smaller perturbations (typically smaller $\varepsilon$) results in models with higher standard but lower adversarial accuracy, and vice versa. In this sense, we want to understand if the strength of the image transformations introduced through data augmentation can also cause such phenomenon in the context of robustness to common corruptions. As described in Sec. 2, each of the transformations of PRIME has a strength parameter $\sigma$, which can be seen as the analogue of $\varepsilon$ in adversarial robustness. Hence, we can easily reduce or increase the strength of the transformations by setting $\hat{\sigma} = \alpha \sigma$, where $\alpha \in \mathbb{R}^+$. Then, by training a network for different values of $\alpha$ we can monitor its accuracy on the clean and the corrupted datasets.

We train a ResNet-18 on C-10 and IN-100 using the setup of Sec. 4.1. For reducing complexity, we do not use the JSD loss and we train for 30 epochs. This could cause some performance drop compared to the results of Tab. 1, but we expect the overall trends in terms of accuracy and robustness to be preserved. Regarding the scaling of the parameters’ strength, for C-10 we set $\alpha \in [10^{-3}, 10^2]$ and sample 100 values spaced evenly on a log-scale, while for IN-100 we set $\alpha \in [10^{-2}, 10^2]$ and we sample 20 values.

The results are presented in Fig. 4. For both C-10 and IN-100, it seems that there is a sweet spot for the scale around $\alpha = 0.2$ and $\alpha = 1$ respectively, where the accuracy on common corruptions reaches its maximum. For $\alpha$ smaller than these values, we observe a clear trade-off between validation and robust accuracy. While the robustness to common corruptions increases, the validation accuracy decays. However, for $\alpha$ greater than the sweet-spot values, we observe that the trade-off ceases to exist since both the validation and robust accuracy present similar behaviour (slight decay). In fact, these observations indicate that robust and validation accuracies are not always positively correlated, and that one might have to slightly sacrifice validation accuracy in order to achieve robustness.

5.4. Sample complexity

Finally, we investigate the necessity of performing augmentation during training (on-line augmentation), compared to statically augmenting the dataset before training (off-line augmentation). On the one hand, on-line augmentation is useful when the dataset is huge and storing augmented versions requires a lot of memory. Besides, there are cases where offline augmentation is not feasible as it relies on pre-trained or generative models which are unavailable in certain scenarios, e.g. DeepAugment [20] or AdA [6] cannot be applied on C-100. On the other hand, off-line augmentation may be necessary to avoid the computational cost of generating augmentations during training.

To this end, for each of the C-10 and IN-100 training sets, we augment them with $k = 1, 2, \ldots, 10$ i.i.d. PRIME transformed versions. Afterwards, for different values of $k$, we train a ResNet-18 on the corresponding augmented dataset and report the accuracy on the validation set and the common corruptions. For the training setup, we follow the settings of Sec. 4.1, but without JSD loss. Also, since we increase the size of the training set by $(k+1)$, we also divide the number of training epochs by the same factor, in order to keep the overall number of gradient updates the same.

The performance on common corruptions is presented in Fig. 5. The first thing to notice is that, even for $k = 1$, the obtained robustness to common corruptions is already quite good. In fact, for IN-100 the accuracy (65%) is already better than the best achievable result of on-line AugMix (60.7% with JSD loss cf. Tab. 2). Regarding C-10 we observe that for $k = 4$ the actual difference with respect

| Method          | Min. cosine distance ($\times 10^{-3}$) |
|-----------------|----------------------------------------|
|                 | Avg. (↓) | Median (↓) |
| None (clean)    | 25.38    | 6.44       |
| AugMix (w/o mix)| 20.57    | 3.56       |
| PRIME (w/o mix) | 10.61    | 1.88       |
| AugMix          | 17.48    | 2.61       |
| PRIME           | 7.71     | 1.61       |

Table 5. Minimum cosine distances in the ResNet-50 SimCLRv2 embedding space between 100 augmented samples from 1000 ImageNet images, and their corresponding common corruptions.
to the on-line augmentation is almost negligible (88.8% vs. 89.3%), especially considering the overhead of transforming the data at every epoch. Technically, this means that augmenting C-10 with 4 PRIME counterparts off-line is enough for achieving good robustness to common corruptions.

Finally, we also see in Fig. 5 that for IN-100 the corruption accuracy presents a very slow improvement after \( k = 4 \). Comparing the accuracy at this point (67.2%) to the one obtained with on-line augmentation without JSD (68.8% cf. Tab. 2) we observe a gap of 1.6%. Hence, given the high complexity of on-line augmentation on such large scale datasets, simply augmenting the training with 4 extra PRIME samples presents a good compromise for achieving competitive robustness. Nevertheless, the increase of 1.6% introduced by on-line augmentation is rather significant, hinting that generating transformed samples during training might be necessary for maximizing performance. In this regard, the lower computational complexity of PRIME allows it to easily achieve this +1.6% through on-line augmentation, since it only requires \( 1.27\times \) additional training time compared to standard training. This can be a significant advantage with respect to other methods, like DeepAugment, that cannot be even applied on-line.

6. Related work

Common corruptions Towards evaluating the robustness of deep neural networks (DNNs) to natural distribution shifts, authors in [21] proposed common corruptions benchmarks (CIFAR-10-C and ImageNet-C) constituting 15 realistic image distortions. Later studies [20], while considering the example of blurring, demonstrated that performance improvements on these common corruptions do generalize to real-world images; consequently supporting the use of common corruptions benchmarks. Recent work [29] showed that current augmentation techniques undergo a performance degradation when evaluated on corruptions that are perceptually dissimilar from those in ImageNet-C. In addition to common corruptions, current literature studies other benchmarks e.g. adversarially filtered data [23], artistic renditions [20] and in-domain datasets [36].

Improving corruption robustness Data augmentation has been the central pillar for improving the generalization performance of DNNs [11, 13, 26, 46, 47]. A notable augmentation strategy for endowing corruption robustness is AugMix [22] which employs a careful combination of stochastic augmentation operations and mixing. Although AugMix attains significant gains on CIFAR-10-C, it does not perform well against sophisticated benchmarks like ImageNet-C. DeepAugment (DA) [20] addresses this issue and diversifies the space of augmentations by introducing distorted images computed by perturbing the weights of image-to-image networks. DA, combined with AugMix, achieves the current state-of-the-art on ImageNet-C. Other schemes include: (i) worst-case noise training [37], (ii) inducing shape bias through stylized images [17], (iii) adversarial counterparts of DeepAugment [6] and AugMix [43], (iv) pre-training and/or adversarial training [24,45], (v) constraining the total variation of convolutional layers [38] and (vi) learning the image information in the phase rather than amplitude [7]. Besides, Vision Transformers [15] have been shown to be more robust to common corruptions than standard CNNs [4,31]. Finally, unsupervised domain adaptation [3,39] using a few corrupted samples has also been shown to provide a considerable boost in corruption robustness. Nonetheless, domain adaptation is orthogonal to this work as it requires knowledge of a target distribution.

7. Concluding remarks

We took a systematic approach to understand the notion of common corruptions and formulated a universal model that captures a wide variety of semantic preserving, common image transformations. Crucially relying on this model, we proposed a novel data augmentation scheme called PRIME, which instantiates our model of corruptions, to confer robustness against common corruptions. From a practical perspective, our method is principled yet efficient and can be conveniently incorporated into existing training procedures. Moreover, it yields a strong baseline on existing corruption benchmarks outperforming current singleton methods. Additionally, we provided useful insights stemming from PRIME that strengthen our understanding of different aspects in the realm of common corruptions. Our findings underlined the importance of diversity across augmentation groups, the efficacy of mixing and also touched on the robustness-accuracy and online-offline augmentation trade-offs. We believe that our insights and PRIME pave the way for building robust models in real-life scenarios. PRIME, for instance, provides a ready-to-use recipe for data-scarce domains such as medical imaging.
Acknowledgments

We thank Alessandro Favero for the fruitful discussions and feedback. This work has been partially supported by the CHIST-ERA program under Swiss NSF Grant 20CH21_180444, and partially by Google via a Postdoctoral Fellowship and a GCP Research Credit Award.

References

[1] Domain generalization on ImageNet-C. https://paperswithcode.com/sota/domain-generalization-on-imagenet-c. Accessed: 2021-12-22. 2

[2] Paul Beale. Statistical Mechanics. Elsevier, 1996. 11

[3] Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. Revisiting batch normalization for improving corruption robustness. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021. 5, 8

[4] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Anders Veit. Understanding robustness of Transformers for image classification. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 8

[5] Franz Binder, Erhard Aichinger, Jürgen Ecker, Christof Nöhauer, and Peter Mayr. Algorithms for near-rings of nonlinear transformations. Association for Computing Machinery, 2000. 2

[6] Dan A. Calian, Florian Stimberg, Olivia Wiles, Sylvestre-Alvise Rebuffi, Andras Gyorgy, Timothy Mann, and Sven Gowal. Defending against image corruptions through adversarial augmentations. arXiv preprint arXiv:2104.01086, 2021. 1, 7, 8

[7] Guangyao Chen, Peixi Peng, Li Ma, Jia Li, Lin Du, and Yonghong Tian. Amplitude-phase recombination: Rethinking robustness of convolutional neural networks in frequency domain. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 1, 8

[8] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. In Advances in Neural Information Processing Systems, 2020. 6

[9] Thomas M. Cover and Joy A. Thomas. Elements of Information Theory. Wiley-Interscience, 2006. 2, 11

[10] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2021. 2, 4, 16, 17

[11] Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 1, 8

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009. 4

[13] Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552, 2017. 8

[14] Samuel Dodge and Lina Karam. Understanding how image quality affects deep neural networks. In 2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX), 2016. 1

[15] 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 International Conference on Learning Representations, 2021. 8

[16] Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Analysis of classifiers’ robustness to adversarial perturbations. Machine Learning, 107(3):481–508, 2018. 7

[17] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In International Conference on Learning Representations, 2019. 8, 15

[18] Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In Advances in Neural Information Processing Systems, 2018. 1

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016. 4, 13

[20] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kada-vath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In IEEE Conference on Computer Vision and Pattern Recognition, 2021. 1, 7, 8, 13, 15

[21] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In International Conference on Learning Representations, 2019. 1, 3, 4, 8

[22] Dan Hendrycks*, Norman Mu*, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple method to improve robustness and uncertainty under data shift. In International Conference on Learning Representations, 2020. 1, 4, 6, 8

[23] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021. 8

[24] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the effectiveness of adversarial training against common corruptions. arXiv preprint arXiv:2103.02325, 2021. 1, 8

[25] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009. 4
[26] Raphael Gontijo Lopes, Dong Yin, Ben Poole, Justin Gilmer, and Ekin D. Cubuk. Improving robustness without sacrificing accuracy with patch gaussian augmentation. arXiv preprint arXiv:1906.02611, 2019. 8

[27] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In International Conference on Learning Representations, Apr. 2018. 7

[28] Mohammad Saeed Masiha, Amin Gohari, Mohammad Hossein Yassaei, and Mohammad Reza Aref. Learning under distribution mismatch and model misspecification. In IEEE International Symposium on Information Theory (ISIT), 2021. 11

[29] Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. arXiv preprint arXiv:2102.11273, 2021. 1, 7, 8, 15

[30] Mazda Moayeri and Soheil Feizi. Sample efficient detection and classification of adversarial attacks via self-supervised embeddings. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 6

[31] Katelyn Morrison, Benjamin Gilby, Colton Lipchak, Adam Mattioli, and Adriana Kovashka. Exploring corruption robustness: Inductive biases in vision transformers and mlp-mixers. arXiv preprint arXiv:2106.13122, 2021. 8

[32] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. Dokl. Akad. Nauk SSSR, 1983. 13

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems, 2019. 13

[34] Leonardo Petrini, Alessandro Favero, Mario Geiger, and Matthieu Wyart. Relative stability toward diffeomorphisms indicates performance in deep nets. In Advances in Neural Information Processing Systems, 2021. 3, 11, 12

[35] Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. In Proceedings of the 37th International Conference on Machine Learning, July 2020. 7

[36] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In Proceedings of the 36th International Conference on Machine Learning, 2019. 8

[37] Evgenia Rusak, Lukas Schott, Roland S. Zimmermann, Julian Bitterwolf, Oliver Bringham, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In Computer Vision – ECCV 2020, 2020. 1, 8

[38] Tommoy Saikia, Cordelia Schmid, and Thomas Brox. Improving robustness against common corruptions with frequency biased models. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 8

[39] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. Improving robustness against common corruptions by covariate shift adaptation. In Advances in Neural Information Processing Systems, 2020. 4, 5, 8

[40] Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of residual networks using large learning rates. arXiv preprint arXiv:1708.07120, 2018. 13

[41] Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. In Advances in Neural Information Processing Systems, 2020. 1

[42] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In International Conference on Learning Representations, May 2019. 7

[43] Haotao Wang, Chaowei Xiao, Jean Kossaifi, Zhiding Yu, Anima Anandkumar, and Zhangyang Wang. Augmax: Adversarial composition of random augmentations for robust training. In Advances in Neural Information Processing Systems, 2021. 1, 4, 6, 8

[44] Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, Advances in Neural Information Processing Systems, 2017. 11

[45] Mingyang Yi, Lu Hou, Jiacheng Sun, Lifeng Shang, Xin Jiang, Qun Liu, and Zhiming Ma. Improved OOD generalization via adversarial training and pretraining. In Proceedings of the 86th International Conference on Machine Learning, 2021. 8

[46] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In 2019 IEEE/CVF International Conference on Computer Vision, 2019. 4, 6, 8

[47] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In International Conference on Learning Representations, 2018. 4, 6, 8

[48] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. 6
A. Maximum entropy transformations

To guarantee as much diversity as possible in our model of common corruptions, we follow the principle of maximum entropy to define our distributions of transformations [9]. Note that using a set of augmentations that guarantees maximum entropy comes naturally when trying to optimize the sample complexity derived from certain information theoretic generalization bounds, both in the clean [44] and corrupted setting [28]. Specifically, the principle of maximum entropy postulates favoring those distributions that are as unbiased as possible given the set of constraints that defines a family of distributions. In our case, these constraints are given in the form of an expected strength, i.e., $\sigma^2$, desired smoothness, i.e., $K$, and/or some boundary conditions, e.g., the displacement field must be zero at the borders of an image.

Let us make this formal. In particular, let $\mathcal{I}$ denote the space of all images $x : \mathbb{R}^2 \rightarrow \mathbb{R}^3$, and let $f : \mathcal{I} \rightarrow \mathcal{I}$ denote a random image transformation distributed according to the law $\mu$. Further, let us define a set of constraints $\mathcal{C} \subseteq \mathcal{F}$, which restrict the domain of applicability of $f$, i.e., $f \in \mathcal{C}$, and where $\mathcal{F}$ denotes the space of functions $\mathcal{I} \rightarrow \mathcal{I}$. The principle of maximum entropy postulates using the distribution $\mu$ which has maximum entropy given the constraints:

$$\begin{align*}
\text{maximize} \quad & H(\mu) = \int_{\mathcal{F}} d\mu(f) \log(\mu(f)) \quad (8) \\
\text{subject to} \quad & f \in \mathcal{C} \quad \forall f \sim \mu,
\end{align*}$$

where $H(\mu)$ represents the entropy of the distribution $\mu$ [9]. In its general form, solving Eq. (8) for any set of constraints $\mathcal{C}$ is intractable. However, leveraging results from statistical physics, we will see that for our domains of interest, Eq. (8) has a simple solution. In what follows we derive those distributions for each of our family of transformations.

A.1. Spectral domain

As we introduced in Sec. 2, we propose to parameterize our family of spectral transformations using an FIR filter of size $K_\omega \times K_\omega$. That is, we are interested in finding a maximum entropy distribution over the space of spectral transformations with a finite spatial support.

Nevertheless, on top of this smoothness constraint we are also interested in controlling the strength of the transformations. We define the strength of a distribution of random spectral transformations applied to an image $x$, as the expected $L_2$ norm of the difference between the clean and transformed images, i.e.,

$$\mathbb{E}_\mu \|x - \omega(x)\|_2^2 = \mathbb{E}_{\omega'} \|\omega' \ast x\|_2^2, \quad (9)$$

which using Young’s convolution inequality is bounded as

$$\mathbb{E}_\omega \|\omega' \ast x\|_2^2 \leq \|x\|_2^2 \mathbb{E}_\omega \|\omega'\|_2^2. \quad (10)$$

Indeed, we can see that the strength of a distribution of random smooth spectral transformations is governed by the expected norm of its filter. In the discrete domain, this can be simply computed as

$$\mathbb{E}_{\omega'} \|\omega'\|_2^2 = \sum_{j=1}^{K_\omega} \sum_{j=1}^{K_\omega} E_{\omega'} \omega_{i,j}^2. \quad (11)$$

Considering this, we should then look for a maximum entropy distribution whose samples satisfy

$$C = \{\omega' \in \mathbb{R}^{K_\omega \times K_\omega} \land \mathbb{E}_{\omega'} \|\omega'\|_2^2 = K^2 \sigma^2 \mid \omega \sim \mu\}. \quad (12)$$

Now, note that this set is defined by an equality constraint involving a sum of $K^2$ quadratic random variables. In this sense, we know that the Equipartition Theorem [2] applies and can be used to identify the distribution of maximum entropy. That is, the solution of Eq. (8) in the case that $C$ is given by Eq. (12), is equal to the distribution of FIR filters whose coefficients are iid with law $\mathcal{N}(0, \sigma^2_\omega)$.

A.2. Spatial domain

The distribution of diffeomorphisms of maximum entropy with a fixed norm was derived by Petrini et al. in [34]. The derivation is similar to the spectral domain, but with the additional constraint that the diffeomorphisms produce a null displacement at the borders of the image.

A.3. Color domain

We can follow a very similar route to derive the distribution of maximum entropy among all color transformations, where, specifically, we constraint the transformations to yield $\gamma(0) = 0$ and $\gamma(1) = 1$ on every channel independently. Doing so, the derivation of the maximum entropy distribution can follow the same steps as in [34].

B. PRIME implementation details

In this section, we provide additional details regarding the implementation of PRIME described in Sec. 3. Since the parameters of the transformations are empirically selected, we first provide more visual examples for different values of smoothness $K$ and strength $\sigma$. Then, we give the exact values of the parameters we use in our experiments supported by additional visual examples and we also describe the parameters we use for the mixing procedure.

B.1. Additional transformed examples

Here we provide additional visual examples for each of the primitives of PRIME illustrating the effect of the following two factors: (i) smoothness controlled by parameter $K$, and (ii) strength of the transformation $\sigma$ on the resulting transformed images created by the primitives. Figs. 6,
7 and 8 demonstrate the resulting spectrum of images created by applying spectral, spatial and color transformations while varying the parameters $K$ and $\sigma$. Notice how increasing the strength $\sigma$ of each transformation drifts the augmented image farther away from its clean counterpart, yet produces plausible images when appropriately controlled.

B.2. Transformation parameters

We now provide the parameters of each transform that we selected and used in our experiments. In general, the values might vary for inputs of different dimensionality and resolution (i.e., CIFAR-10/100 vs ImageNet images).

Figure 6. Example images (IN-100) generated with spectral transformations from our common corruptions model. In each row, we enlarge the transformation strength $\sigma_\omega$ from left to right. From top to bottom, we increase the spectral resolution of the filter $K_\omega$.

Figure 7. Example images (IN-100) generated with spatial transformations from our common corruptions model. In each row, we enlarge the transformation strength $\sigma_\tau$ from left to right. From top to bottom, we increase the cut frequency $K_\tau$.

Figure 8. Example images (IN-100) generated with color transformations from our common corruptions model. In each row, we enlarge the transformation strength $\sigma_\gamma$ from left to right. From top to bottom, we increase $K_\gamma$.

Regarding the spectral transform of Eq. (2) we found out that, for the FIR filter $\omega'$, a size of $K_\omega = 3$ results into semantically preserving images for CIFAR-10/100 and ImageNet. For the latter, one can stretch the filter size to $5 \times 5$ or even $7 \times 7$, but then slight changes on the strength, $\sigma_\omega$, might destroy the image semantics. Eventually, given $K_\omega = 3$, we observed that $\sigma_\omega = 4$ is good enough for CIFAR-10/100 and ImageNet. Concerning the spatial transform of Eq. (4), for the cut-off parameter $K_\tau$ we followed the value regimes proposed by Petrini et al. [34] and set $K_\tau = 100$ for CIFAR-10/100; $K_\tau = 500$ for ImageNet. Furthermore, for a given $K_\tau$, Petrini et al. also compute the appropriate bounds for the transformation strength, $\sigma_\tau^2 \leq \sigma_\tau^2 \leq \sigma_\tau^2$, such that the resulting diffeomorphism remains bijective and the pixel displacement does not destroy the image. In fact, in their original implementation, Petrini et al. directly sample $\sigma_\tau \sim U(\sigma_\tau^\text{min}, \sigma_\tau^\text{max})$ instead of explicitly setting the strength. In our implementation, we also follow the same approach. Regarding the color transform of Eq. (5) we found out that for CIFAR-10/100 a cut-off value of $K_\gamma = 100$ and a strength of $\sigma_\gamma = 0.01$ result into semantically preserving images for CIFAR-10/100; while for ImageNet, the corresponding values are $K_\gamma = 500$ and $\sigma_\gamma = 0.05$. As for the bandwidth (consecutive frequencies) $\Delta$ we observed that a value of $\Delta = 20$ was memory sufficient for ImageNet, but for CIFAR-10/100, due to its lower dimensionality, we can use all the frequencies to be used, e.g. $\Delta = K_\gamma$.

Finally, as mentioned in Sec. 3, we randomly sample the strength of the transformations $\sigma$ from a uniform distribution of given minimum and maximum values. Regarding

5The official implementation of Petrini et al. diffeomorphisms can be found at https://github.com/pcsl-epfl/diffeomorphism.
Figure 9. Example images (IN) generated with the transformations of our common corruptions model. Despite the perceptibility of the introduced distortion, the image semantics are preserved.

**B.3. Parameters for mixing procedure**

Regarding the mixing parameters of our experiments, we fix the total number of generated transformed images (width) to be $n = 3$. As for the composition of the transformations (depth), we follow a stochastic approach such that, on every iteration $i \in \{1, \ldots, n\}$, only $\hat{m} \in [1, m]$ compositions are performed, with $m = 3$. In fact, in Algorithm 1 we do not explicitly select randomly a new $\hat{m}$ for every $i$ but we provide the identity operator $\text{Id}$ instead. This guarantees that, in some cases, no transformation is performed.

**C. Detailed experimental setup**

We now provide all the experimental details for the performance evaluation of Sec. 4. All models are implemented in PyTorch [33] and are trained for 100 epochs using a cyclic learning rate schedule [40] with cosine annealing and a maximum learning rate of 0.2 unless stated otherwise. For IN, we fine-tune a regularly pretrained network$^6$ with a maximum learning rate of 0.01 following Hendrycks et al. [20]. We use SGD optimizer with momentum factor 0.9 and Nesterov momentum [32]. On C-10 & C-100, we set the batch size to 128 and use a weight decay of 0.0005. On IN-100 and IN, the batch size is 256 and weight decay is 0.0001. We employ ResNet-18 [19] on C-10, C-100 and IN-100; and use ResNet-50 for IN. The augmentation hyperparameters for AugMix and DeepAugment are the same as in their original implementations.

**D. Additional mixing examples**

Continuing Sec. 5.2, we present additional examples in Fig. 10 to demonstrate the significance of mixing in PRIME. We observe that the mixing procedure is capable of constructing augmented images that look perceptually similar to common corruptions. To illustrate this, we provide several examples in Fig. 10 for PRIME (upper half) and AugMix (lower half) on CIFAR-10 and ImageNet-100. As shown in Figs. 10a and 10b, mixing spectral transformations with the clean images tends to create weather-like artefacts resembling frost and fog respectively. Carefully combining clean and spatially transformed images produces blurs (Fig. 10c) and even elastic transform (Fig. 10e). Moreover, blending color augmentation with clean image produces shot noise as evident in Fig. 10d; Whereas spectral+color transformed image looks similar to snow corruption (Fig. 10f). All these observations explain the good performance of PRIME on the respective corruptions.

---

$^6$We use the pretrained model available at [https://pytorch.org/docs/stable/torchvision/models.html](https://pytorch.org/docs/stable/torchvision/models.html)
Figure 10. The mixing procedure creates distorted images that look visually similar to the test-time corruptions. In each example (CIFAR-10/ImageNet-100), we show the clean image, the PRIME/AugMix augmented image and the corresponding common corruption that resembles the image produced by mixing. We also report the mixing combination used for recreating the corruption. ◦ stands for composition and + represents convex combination (mixing). (Top 3 rows): PRIME, and (Last 2 rows): AugMix.

Apart from the mixing in PRIME, the mixing in AugMix also plays a crucial role in its performance. In fact, a combination of translate and shear operations with the clean image create blur-like modifications that resemble defocus blur (Fig. 10g) and motion blur (Fig. 10). This answers why AugMix excels at blur corruptions and is even better than DeepAugment against blurs (cf. Tab. 2). In addition, on CIFAR-10, notice that mixing solarize and clean produces impulse noise-like modifications (Fig. 10j), which justifies the improvements on noise attained by AugMix (refer Tab. 7).

E. SimCLR nearest neighbours

Regarding the minimum distances in the SimCLRv2 embedding space of Tab. 5, we also provide in Fig. 11 some visual examples of the nearest neighbours of each method. In general, we observe that indeed smaller distance in the embedding space typically corresponds to closer visual similarity in the input space, with PRIME generating images that resemble more the corresponding common corruptions, compared to AugMix. Nevertheless, we also notice that for “Blurs” AugMix generates images that are more visually similar to the corruptions than PRIME, an observation that is on par with the lower performance of PRIME (without JSD) on blur corruptions (cf. Tab. 2) compared to AugMix.

F. Cosine distance statistics

Recall that in Tab. 5 we provide the average and the median of the minimum cosine distances computed in the SimCLRv2 embedding space. We now provide in Tab. 6 the values for different percentiles of these distances. We observe that the behaviour is consistent across different percentiles: PRIME (with or without mixing) is always producing feature representations that are more similar to the common...
corruptions, compared to any version of AugMix. Note also that for smaller percentiles (5%, 10%, 25%) it seems that PRIME without mixing reaches even lower values than PRIME. However, the difference with respect to PRIME can be considered as insignificant since it is in the order of $10^{-5}$ (note that all values in the table are in the order of $10^{-3}$); while a larger population of images (>1000) would potentially smooth out this difference.

G. Performance per corruption

Beyond the average corruption accuracy that we report in Tab. 1, we also provide here the performance of each method on the individual corruptions. The results on CIFAR-10/100 and ImageNet/ImageNet-100 are shown on Tab. 7 and Tab. 8 respectively. Compared to AugMix on CIFAR-10/100, the improvements from PRIME are mostly observed against Gaussian noise (+9.7%/21.5%), shot noise (+5.5%/15.6%), glass blur (+7.1%/12.3%) and JPEG compression (+2.0%/4.4%). These results show that PRIME can really push the performance against certain corruptions in CIFAR-10/100-C despite the fact that AugMix is already good on these datasets. However, AugMix turns out to be slightly better than PRIME against impulse noise, defocus blur and motion blur modifications; all of which have been shown to be resembled by AugMix created images (see Fig. 10). With ImageNet-100, PRIME enhances the diversity of augmented images, and leads to general improvements against all corruptions except certain blurs. On ImageNet, we observe that, in comparison to DeepAugment, the supremacy of PRIME is reflected on almost every corruption type, except some blurs and pixelate corruptions where DeepAugment is slightly better. When PRIME is used in conjunction with DeepAugment, compared to AugMix combined with DeepAugment, our method seems to lack behind only on blurs, while on the rest of the corruptions achieves higher robustness.

H. Performance per severity level

We also want to investigate the robustness of each method on different severity levels of the corruptions. The results for CIFAR-10/100 and ImageNet/ImageNet-100 are presented in Tab. 9 and Tab. 10 respectively. With CIFAR-10/100, PRIME predominantly helps against corruptions with maximal severity and yields +3.4% and +7.1% gains on CIFAR-10 and CIFAR-100 respectively. Besides on ImageNet-100, PRIME again excels at corruptions with moderate to higher severity. This observations also holds when PRIME is employed in concert with DeepAugment. With ImageNet too this trend continues, and we observe that, compared to DeepAugment, PRIME improves significantly on corruptions of larger severity (+3.4% and +5.5% on severity levels 4 and 5 respectively). Also, this behaviour is consistent even when PRIME is combined with DeepAugment and is compared to DeepAugment+AugMix, where we see that again on levels 4 and 5 there is a significant improvement of +2.1% and +3.7% respectively.

I. Performance on other corruptions

Finally, to examine the universality of PRIME, we evaluate the performance of our ImageNet-100 trained models against two other corrupted datasets: (i) ImageNet-100-C (IN-100-C) [29], and (ii) stylized ImageNet-100 (SIN-100) [17]. While IN-100-C is composed of corruptions that are perceptually dissimilar to those in IN-100-C, stylized IN-100 only retains global shape information and discard local texture cues from IN-100 test images, via style transfer. Thus, it would be interesting to test the performance of PRIME against these datasets since it would serve as a indicator for general corruption robustness of PRIME. More information about the corruption types contained in IN-100-C is available in the original paper [29].

Tab. 11 enumerates the classification accuracy of different standalone approaches against IN-100-C on average, individual corruptions in IN-100-C and SIN-100. We can see that PRIME surpasses AugMix and DeepAugment by 4% and 1.2% respectively on IN-100-C. PRIME particularly helps against certain distortions such as blue noise sample (BSmpl), inverse sparkles and plasma noise. PRIME also works well against style-transferred images in SIN-100 and improves accuracy by 5.1% over AugMix and 3.2% over DeepAugment. Besides, the diversity of our method means that we can actually get a better performance by increasing the number of training epochs. With 1.5x training epochs, we observe about 1% accuracy refinement on each benchmark.

We also perform a similar analysis with ImageNet trained models and evaluate their robustness on three other distribution shift benchmarks: (i) IN-C [29], (ii) SIN [17] as described previously and (iii) ImageNet-R (IN-R) [20]. ImageNet-R contains naturally occurring artistic renditions (e.g., paintings, embroidery, etc.) of objects from the ImageNet dataset. The classification accuracy achieved by different methods on these datasets is listed in Tab. 12.
IN-C, PRIME outperforms AugMix and DeepAugment by 3.1% and 1.3% respectively. Besides, PRIME also obtains competitive results on IN-R and SIN datasets. Altogether, our empirical results indicate that the performance gains obtained by PRIME indeed translate to other corrupted datasets.

| Dataset | Method  | Clean CC | CC Avg. | Noise | Shot | Impulse | Blur | Defocus | Glass | Motion | Zoom | Weather | Frost | Fog | Bright | Contrast | Elastic | Pixel | Digital | JPEG |
|---------|---------|----------|---------|-------|-------|---------|------|---------|-------|--------|------|---------|-------|-----|--------|----------|--------|-------|---------|------|
| C-10    | Standard | 95.0     | 74.0    | 45.1  | 58.7  | 54.9    | 83.2 | 53.3    | 76.9  | 79.1   | 83.1 | 79.3   | 89.0  | 93.6 | 76.5   | 83.9     | 75.1   | 77.9   |
|         | AugMix   | 95.2     | 88.6    | 79.3  | 84.8  | 85.8    | 94.1 | 78.9    | 92.4  | 93.4   | 89.7 | 89.0   | 91.9  | 94.3 | 90.5   | 90.5     | 87.6   | 87.5   |
|         | PRIME    | 93.1     | 89.0    | 89.0  | 90.3  | 84.2    | 91.2 | 85.7    | 89.5  | 90.9   | 88.3 | 89.9   | 86.8  | 92.4 | 89.8   | 89.1     | 89.0   | 89.5   |
| C-100   | Standard | 76.1     | 51.9    | 25.3  | 33.7  | 26.6    | 60.8 | 47.1    | 55.5  | 57.6   | 60.8 | 56.2   | 62.5  | 72.2 | 53.2   | 63.4     | 50.1   | 52.7   |
|         | AugMix   | 78.2     | 64.9    | 46.7  | 55.1  | 60.6    | 76.2 | 47.3    | 72.6  | 74.3   | 67.4 | 64.4   | 69.9  | 75.5 | 67.4   | 69.6     | 64.9   | 61.8   |
|         | PRIME    | 77.6     | 68.3    | 68.2  | 70.7  | 59.8    | 72.3 | 59.6    | 69.2  | 71.2   | 68.3 | 69.0   | 67.4  | 75.2 | 71.2   | 69.3     | 66.8   | 66.2   |

Table 7. Per-corruption accuracy of different methods on C-10 and C-100 (ResNet-18).

| Dataset | Method  | Clean CC | CC Avg. | Noise | Shot | Impulse | Blur | Defocus | Glass | Motion | Zoom | Weather | Frost | Fog | Bright | Contrast | Elastic | Pixel | Digital | JPEG |
|---------|---------|----------|---------|-------|-------|---------|------|---------|-------|--------|------|---------|-------|-----|--------|----------|--------|-------|---------|------|
| IN-100  | Standard | 88.0     | 49.7    | 80.9  | 29.0  | 22.0    | 45.6 | 44.6    | 50.4  | 53.9   | 43.8 | 46.2   | 50.5  | 78.6 | 42.9   | 68.8     | 68.9   | 10.6   |
|         | AugMix   | 88.7     | 60.7    | 45.2  | 45.8  | 43.4    | 58.7 | 53.3    | 69.5  | 71.0   | 49.1 | 52.7   | 60.2  | 80.7 | 59.6   | 73.3     | 73.6   | 74.7   |
|         | PRIME    | 85.9     | 71.6    | 80.6  | 80.0  | 80.1    | 57.2 | 66.3    | 66.2  | 68.2   | 61.5 | 68.2   | 57.2  | 81.2 | 68.3   | 73.7     | 82.9   | 81.9   |
| IN-100  | DA       | 86.3     | 67.7    | 76.3  | 75.6  | 75.7    | 64.2 | 61.7    | 61.3  | 62.7   | 54.4 | 62.8   | 55.7  | 81.6 | 49.7   | 69.9     | 83.3   | 80.6   |
|         | DA+AugMix| 86.5     | 71.1    | 75.5  | 75.8  | 74.9    | 74.1 | 68.3    | 76.0  | 72.1   | 59.9 | 68.6   | 61.4  | 82.1 | 72.4   | 71.1     | 83.8   | 81.1   |
| IN      | Standard | 76.1     | 39.2    | 29.3  | 27.0  | 23.8    | 38.8 | 26.8    | 38.7  | 36.2   | 32.5 | 38.1   | 45.4  | 68.0 | 39.0   | 45.3     | 53.4   |
|         | AugMix*  | 77.5     | 48.3    | 40.6  | 41.1  | 37.7    | 47.7 | 34.9    | 53.5  | 49.0   | 39.9 | 43.8   | 47.1  | 69.5 | 51.1   | 52.0     | 57.0   | 60.3   |
|         | DA†      | 76.7     | 52.6    | 56.6  | 54.9  | 56.3    | 51.7 | 40.1    | 48.7  | 39.5   | 44.2 | 50.3   | 52.1  | 71.1 | 48.3   | 50.9     | 65.5   | 59.3   |
| IN      | PRIME†   | 77.0     | 55.0    | 61.9  | 60.6  | 60.9    | 47.6 | 39.0    | 48.4  | 46.0   | 47.4 | 50.8   | 54.1  | 71.7 | 58.2   | 56.3     | 59.5   | 62.2   |
|         | DA+AugMix| 75.8     | 58.1    | 59.4  | 59.6  | 59.1    | 59.0 | 46.8    | 61.1  | 51.5   | 49.4 | 53.3   | 55.9  | 70.8 | 58.7   | 54.3     | 68.3   | 61.5   |
|         | DA+PRIME†| 75.5     | 59.9    | 67.4  | 67.2  | 66.8    | 56.2 | 47.5    | 54.3  | 47.3   | 52.8 | 56.4   | 56.3  | 71.7 | 62.3   | 57.3     | 70.3   | 65.1   |

Table 8. Per-corruption accuracy of different methods on IN-100 (ResNet-18) and IN (ResNet-50). † indicates that JSD consistency loss is not used. * Models taken from RobustBench [10].
| Dataset | Method | Clean | CC Avg. | 1 | 2 | 3 | 4 | 5 |
|---------|--------|-------|---------|---|---|---|---|---|
| C-10    | Standard | 95.0  | 74.0    | 87.4 | 81.7 | 75.7 | 68.3 | 56.7 |
|         | AugMix  | 95.2  | 88.6    | 93.1 | 91.8 | 89.9 | 86.7 | 81.7 |
|         | PRIME   | 93.1  | 89.0    | 91.7 | 90.8 | 89.8 | 87.9 | 85.1 |
| C-100   | Standard | 76.7  | 51.9    | 66.7 | 59.4 | 52.8 | 45.0 | 35.4 |
|         | AugMix  | 78.2  | 64.9    | 73.3 | 70.0 | 66.6 | 61.3 | 53.4 |
|         | PRIME   | 77.6  | 68.3    | 74.0 | 71.8 | 69.3 | 65.7 | 60.5 |

Table 9. Average accuracy for each corruption severity level of different methods on C-10 and C-100 (ResNet-18).

| Dataset | Method | Clean | CC Avg. | 1 | 2 | 3 | 4 | 5 |
|---------|--------|-------|---------|---|---|---|---|---|
| IN-100  | Standard | 88.0  | 49.7    | 73.5 | 61.0 | 49.8 | 37.2 | 27.0 |
|         | AugMix  | 88.7  | 60.7    | 80.4 | 71.8 | 63.8 | 50.3 | 37.2 |
|         | DA      | 86.3  | 67.7    | 81.2 | 75.4 | 69.9 | 61.2 | 50.8 |
|         | PRIME   | 85.9  | 71.6    | 81.7 | 77.5 | 73.4 | 66.9 | 58.4 |
|         | DA+AugMix | 86.5  | 73.1    | 82.7 | 78.0 | 75.5 | 69.6 | 59.9 |
|         | DA+PRIME | 84.9  | 74.9    | 82.0 | 78.7 | 76.4 | 71.8 | 65.5 |
| IN      | Standard | 76.1  | 39.2    | 60.6 | 49.8 | 39.8 | 27.7 | 18.0 |
|         | AugMix  | 77.5  | 48.3    | 66.7 | 58.3 | 51.1 | 39.1 | 26.5 |
|         | DA      | 76.7  | 52.6    | 69.0 | 61.7 | 55.4 | 44.9 | 32.1 |
|         | PRIME   | 77.0  | 55.0    | 68.9 | 63.1 | 56.9 | 48.3 | 37.6 |

Table 10. Average accuracy for each corruption severity level of different methods on IN-100 (ResNet-18) and IN (ResNet-50). † indicates that JSD consistency loss is not used. ∗ Models taken from RobustBench [10].

| Method | Clean | IN-100-C Avg. | IN-100-□ Avg. | BSmpl | Brown | Caustic | Ckbd | CSmooth | CSine | ISpark | Perlin | Plasma | SFreq | Spark | SIN-100 |
|--------|-------|----------------|----------------|-------|-------|---------|------|---------|-------|--------|--------|--------|-------|------|---------|
| Standard | 88.0  | 49.7           | 55.1           | 47.6  | 71.3  | 30.1   | 66.4 | 29.3    | 45.7  | 72.1   | 34.6   | 34.9   | 78.4  | 18.8 |
| AugMix  | 88.7  | 60.7           | 61.0           | 63.0  | 73.2  | 75.3   | 69.4 | 39.9    | 44.9  | 77.4   | 42.8   | 44.7   | 79.8  | 28.0 |
| DA      | 86.3  | 67.7           | 63.8           | 77.1  | 76.6  | 72.6   | 60.9 | 42.9    | 44.3  | 78.0   | 43.4   | 64.5   | 77.8  | 29.9 |
| PRIME   | 85.9  | 71.6           | 65.0           | 74.9  | 74.3  | 73.2   | 59.2 | 53.4    | 47.5  | 76.8   | 48.6   | 66.9   | 75.5  | 33.1 |
| +1.5x epochs | 86.1  | 72.5           | 65.9           | 77.1  | 75.6  | 74.1   | 59.4 | 54.0    | 46.3  | 77.6   | 50.4   | 67.7   | 76.4  | 34.1 |

Table 11. Classification accuracy of different methods on IN-100-C, IN-100-□ and Stylized IN-100 (SIN-100) with ResNet-18.

| Method | Clean | IN-C Avg. | IN-□ Avg. | BSmpl | Brown | Caustic | Ckbd | CSmooth | CSine | ISpark | Perlin | Plasma | SFreq | Spark | IN-R |
|--------|-------|-----------|-----------|-------|-------|---------|------|---------|-------|--------|--------|--------|-------|------|------|
| Standard | 76.1  | 39.2      | 40.0      | 36.2  | 57.8  | 54.1   | 46.1 | 14.4    | 20.9  | 61.6   | 24.3   | 19.0   | 65.2  | 36.2 | 7.4  |
| AugMix  | 77.5  | 48.3      | 46.5      | 59.5  | 56.5  | 59.1   | 51.7 | 25.6    | 21.6  | 65.3   | 23.1   | 36.2   | 66.4  | 41.0 | 11.2 |
| DA      | 76.7  | 52.6      | 48.3      | 60.1  | 61.1  | 57.7   | 46.8 | 25.4    | 24.4  | 68.4   | 26.5   | 45.6   | 66.8  | 42.2 | 14.2 |
| PRIME   | 77.0  | 55.0      | 49.6      | 59.5  | 61.4  | 60.1   | 48.1 | 26.9    | 28.3  | 66.5   | 36.4   | 41.9   | 66.5  | 42.2 | 14.0 |

Table 12. Classification accuracy of different methods on IN-C, IN-□, ImageNet-R (IN-R) and Stylized IN (SIN) with ResNet-50. † indicates that JSD consistency loss is not used. ∗ Models taken from RobustBench [10].