DC4L: Distribution Shift Recovery via Data-Driven Control for Deep Learning Models

Proceedings of Machine Learning Research vol 242:1–32, 2024

DC4L: Distribution Shift Recovery via Data-Driven Control for Deep Learning Models

Vivian Lin VILIN@SEAS.UPENN.EDU
Kuk Jin Jang JANGKJ@SEAS.UPENN.EDU
Souradeep Dutta DUTTASO@SEAS.UPENN.EDU
Michele Caprio CAPRIO@SEAS.UPENN.EDU
Oleg Sokolsky SOKOLSKY@SEAS.UPENN.EDU
Insup Lee LEE@SEAS.UPENN.EDU
University of Pennsylvania

Abstract

Deep neural networks have repeatedly been shown to be non-robust to the uncertainties of the real world, even to naturally occurring ones. A vast majority of current approaches have focused on data-augmentation methods to expand the range of perturbations that the classifier is exposed to while training. A relatively unexplored avenue that is equally promising involves sanitizing an image as a preprocessing step, depending on the nature of perturbation. In this paper, we propose to use control for learned models to recover from distribution shifts online. Specifically, our method applies a sequence of semantic-preserving transformations to bring the shifted data closer in distribution to the training set, as measured by the Wasserstein distance. Our approach is to 1) formulate the problem of distribution shift recovery as a Markov decision process, which we solve using reinforcement learning, 2) identify a minimum condition on the data for our method to be applied, which we check online using a binary classifier, and 3) employ dimensionality reduction through orthonormal projection to aid in our estimates of the Wasserstein distance. We provide theoretical evidence that orthonormal projection preserves characteristics of the data at the distributional level. We apply our distribution shift recovery approach to the ImageNet-C benchmark for distribution shifts, demonstrating an improvement in average accuracy of up to 14.21% across a variety of state-of-the-art ImageNet classifiers. We further show that our method generalizes to composites of shifts from the ImageNet-C benchmark, achieving improvements in average accuracy of up to 9.81%. Finally, we test our method on CIFAR-100-C and report improvements of up to 8.25%.

Keywords: distribution shift, Markov decision process, reinforcement learning

1. Introduction

Deep learning models are excellent at learning patterns in large high dimensional datasets. However, the brittleness of deep neural networks (DNNs) to distribution shifts is a challenging problem. Hendrycks and Gimpel (2018) showed that, even in naturally occurring distribution shift scenarios, a classifier's performance can deteriorate substantially. Arguably, one of the most widespread uses of deep learning techniques currently is in image recognition. In this paper, we propose to use decision and control for learned models (DC4L) to improve robustness to distribution shifts, with demonstration on image classification tasks. The idea is to actively *sanitize* a set of images depending on the type of the distribution shift. Intuitively, the training distribution of a classifier is viewed as its *comfort zone*, where the behavior is more predictable and trustworthy. At run time, when exposed to perturbations that push the images outside of this comfort zone, a feedback policy

takes control actions which can bring the images back to a more familiar space. The control actions are so chosen that the semantic meaning of the images are preserved. This ensures correctness. While approaches for DNN robustness have until now focused on data-augmentation techniques (Hendrycks et al., 2019; Erichson et al., 2022; Verma et al., 2019; Yun et al., 2019; Kim et al., 2020b; Hendrycks et al., 2020), we show that by using ideas from the data-driven control paradigm, it is possible to provide an additional level of performance boost beyond what can be offered by SOTA augmentation methods.

Our technique exploits the following observation: when distribution shift arises in the external environment due to natural causes, it persists for a certain duration of time. For instance, when a corruption in image quality occurs due to snow, this corruption does not disappear in the next image frame. This gives the system some time to *adapt* and *recover* from this shift by computing some semantic preserving transformations to the data. Our technique, Supervisory system for Shift Adaptation and Recovery (`SuperStAR`), applies a sequence of semantic-preserving transforms to the input data, correcting the input to align with the original training set of the classifier. We show that formulating the sequence selection problem as a Markov decision process (MDP) lends a natural solution: reinforcement learning (RL).

In summary, our contributions towards addressing the problem of robustness to semantic preserving shifts are as follows. 1) We translate the problem of distribution shift recovery for neural networks to a Markov decision process, which we solve using reinforcement learning. 2) We identify a minimum condition of operability for our method, which we check online using a binary classifier. 3) We develop a method to efficiently compute the degree of distribution shift by projecting to a lower dimensional space. This uses results from Cai and Lim (2022) in conjunction with the Wasserstein distance. 4) We demonstrate an application to ImageNet-C and achieve significant accuracy improvements (up to 14.21% averaged across all shift severity levels) on top of standard training and data-augmentation schemes. We further show that our method generalizes beyond the ImageNet-C benchmark, yielding up to 9.81% on composite Imagenet-C shifts and up to 8.25% on CIFAR-100-C in accuracy improvements.

2. Preliminaries

We begin by assuming that the images are sampled from a measurable space $(\mathcal{X}, \mathcal{A}_\mathcal{X})$. Let $\Delta(\mathcal{X}, \mathcal{A}_\mathcal{X})$ denote the set of all probability measures on $(\mathcal{X}, \mathcal{A}_\mathcal{X})$. We pick a distribution $D \in \Delta(\mathcal{X}, \mathcal{A}_\mathcal{X})$ from which the current set of images are sampled. Assume that the labels belong to a measurable space $(\mathcal{Y}, \mathcal{A}_\mathcal{Y})$, and a classifier C is an $\mathcal{A}_\mathcal{X} \backslash \mathcal{A}_\mathcal{Y}$ measurable map, $\mathcal{C} : \mathcal{X} \to \mathcal{Y}$. An *oracle classifier* $\mathcal{C}^*$ produces the ground truth labels. Next, we define a semantic preserving transform $\mathbb{T}$.

Definition 1 (Semantic Preserving Transform) *A function $\mathbb{T} : \mathcal{X} \to \mathcal{X}$ is semantic preserving iff $\mathcal{C}^*(x) = \mathcal{C}^*(\mathbb{T}(x))$, for all $x \in \mathcal{X}$.*

We denote by $\mathbb{S}$ the set of all such semantic preserving transforms. In the standard empirical risk minimization (ERM) paradigm, we approximate $\mathcal{C}^*$ with some classifier $\mathcal{C}$. When measuring robustness to common corruptions, typically a corrupting transform $\mathbb{T}_c$ belongs to $\mathbb{S}$. This includes transforms like the addition of Gaussian noise, speckle noise, and alike. The error of a classifier $\mathcal{C}$ under the distribution D, is defined as

$$err(D) := \mathbb{E}_{x \sim D} \left[\mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x)) \right],$$

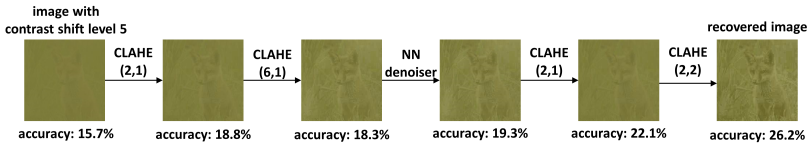

Figure 2: Example transformations applied to an image with contrast shift level 5. CLAHE(x,y) denotes histogram equalization with strength determined by x and y (details in Appendix E). The policy applies a non-trivial composition of transformations that would be difficult to find through manual manipulation. The policy chooses few redundant actions and improves the accuracy of an AugMix-trained ResNet-50 on a random batch of 1000 images.

where $\mathbb{1}$ is the standard indicator function, which evaluates to 1 iff $\mathcal{C}^*(x) \neq \mathcal{C}(x)$.[1] For a robust classifier we expect $err(D)$ to be minimal for multiple choices of the distribution D. For instance in the case of common corruptions introduced in Hendrycks and Dietterich (2019), the goal is to optimize the choice of classifier $\mathcal{C}$ such that $err(D)$ is minimized, even under shift. This is typically achieved using data-augmentation schemes such as Augmix, NoisyMix, and DeepAugment.

3. Functionality and Problem Statement

At a high level, the functioning of `SuperStAR` is akin to a supervisor for the classifier shown in Figure 1. `SuperStAR` detects distribution shifts and computes a recovery strategy to be applied before sending an image to a classifier. Determining the appropriate recovery strategy presents an interesting challenge.

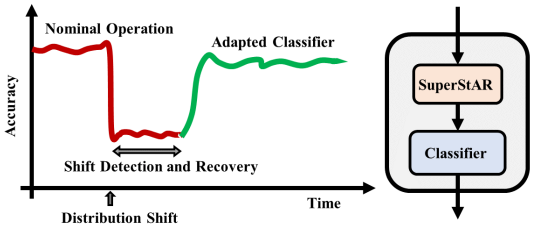

Figure 1: Overview of `SuperStAR`. At deployment, assume that a distribution shift causes a drop in accuracy. This is detected through changes in the Wasserstein distance between a validation set and the corrupted set. `SuperStAR` computes a composition of transforms $\mathcal{I}_k$ to adapt to the shift and recover accuracy. This composition of `SuperStAR` with the classifier helps it detect and adapt, boosting robustness of classification.

Consider a random variable $x_c = \mathbb{T}(z)$, where $z \sim D$ and $\mathbb{T}$ is a semantic preserving transform. Note that $\mathbb{T}$ is generally not invertible. However, one may possibly choose an element $\mathbb{T}'$ from the set of semantic preserving transforms $\mathbb{S}$ that partially recovers the accuracy drop due to $\mathbb{T}$. It might even be effective to select a sequence of such transforms: an ordered set $\mathcal{T} := \{\mathbb{T}_k, \mathbb{T}_{k-1}, \dots \mathbb{T}_1\}$ with $k \geq 1$, such that $\mathbb{S} \ni \mathcal{I}_k(x) := \mathbb{T}_k \circ \mathbb{T}_{k-1} \circ \cdots \circ \mathbb{T}_1(x)$.[2] Ideally, we wish to find an algorithm which optimizes the following:

$$\mathcal{I}_k^* = \text{argmin}_{\mathcal{I}_k} R(\mathcal{I}_k) \text{ , where } R(\mathcal{I}_k) = err(D_{\mathcal{I}_k \circ \mathbb{T}}) - err(D), \tag{1}$$

with $D_{\mathcal{I}_k \circ \mathbb{T}}$ denoting the distribution of $\mathcal{I}_k \circ \mathbb{T}(z)$, $z \sim D$. The transform $\mathcal{I}_k$ is what effectively *reverses* an image corruption due to $\mathbb{T}$.

At deployment, when `SuperStAR` detects a shift in distribution compared to a clean validation set, it should propose a composition of transforms $\mathcal{I}_k$ to minimize the cost outlined in Equation 1. From the vantage point of classifier $\mathcal{C}$, images transformed using $\mathcal{I}_k$ *appear* to be closer to the *home* distribution D. An example result of this is shown in Figure 2.

4. Distribution Shift Recovery is a Markov Decision Process

Our task is to compute a composition of transforms $\mathcal{I}_k$ to apply to the corrupted set $\mathbf{V}_c$, sampled i.i.d from $D_{\mathbb{T}}$, to realize the optimization cost outlined in Equation 1. To this end, we note the following theorem.

1. For notational convenience, we hereafter denote both a random variable and its realization by a lowercase Latin letter.
2. The order $\prec_{\mathcal{T}}$ on $\mathcal{T}$ is given by $\mathbb{T}_i \prec_{\mathcal{T}} \mathbb{T}_j \iff i < j, i, j \in \mathbb{N}$. $\mathbb{T}_0$ is assumed to be the identity function.

Theorem 2 $R(\mathcal{I}_k) \leq \alpha \cdot d_{TV}(D, D_{\mathcal{I}_k \circ \mathbb{T}})$*, for some finite* $\alpha \in \mathbb{R}$ *and semantic preserving transform* $\mathcal{I}_k$*.*

Proof Let us denote by d the pdf of $D_{\mathcal{I}_k \circ \mathbb{T}}$ and by d' the pdf of D. We begin by expanding the definition of $R(\mathcal{I}_k)$.

$$\begin{aligned}
R(\mathcal{I}_k) &= err(D_{\mathcal{I}_k \circ \mathbb{T}}) - err(D) \\
&= \mathbb{E}_{x_1 \sim D_{\mathcal{I}_k \circ \mathbb{T}}, x_2 \sim D} \left[\mathbb{1}(\mathcal{C}^*(x_1) \neq \mathcal{C}(x_1)) - \mathbb{1}(\mathcal{C}^*(x_2) \neq \mathcal{C}(x_2)) \right] \\
&= \textstyle\int_{\mathcal{X}} \mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x))[d(x) - d'(x)] \mathrm{d}x \\
&\leq \left| \textstyle\int_{\mathcal{X}} \left(\sup_{x \in \mathcal{X}} \mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x)) \right) [d(x) - d'(x)] \mathrm{d}x \right| \\
&= \left| \left(\sup_{x \in \mathcal{X}} \mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x)) \right) \textstyle\int_{\mathcal{X}} [d(x) - d'(x)] \mathrm{d}x \right| \\
&\leq \left| \sup_{x \in \mathcal{X}} \mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x)) \right| \cdot \sup_{A \in \mathcal{A}_{\mathcal{X}}} \left| \textstyle\int_A [d(x) - d'(x)] \mathrm{d}x \right| \\
&= \alpha \cdot d_{TV}(D, D_{\mathcal{I}_k \circ \mathbb{T}}),
\end{aligned}$$

where $\alpha = \left| \sup_{x \in \mathcal{X}} \mathbb{1}(\mathcal{C}^*(x) \neq \mathcal{C}(x)) \right|$. The first step assumes $\mathcal{X}$ is a continuous space. The rest follows from expressing the definition of computing expectation in terms of the classifier error. ■

Thus, for a transform $\mathbb{T}$ (possibly a corruption), it is possible for the classifier to recover performance if the apparent distribution under $\mathcal{I}_k \circ \mathbb{T}$ is close enough to the original distribution D. Then the optimal $\mathcal{I}_k$ is the sequence of semantic preserving transforms that minimize the distance from D. Equivalently, this can be viewed as a sequence of actions maximizing a reward. More formally, the task of computing $\mathcal{I}_k$ can be formulated as a reactive policy for an MDP, which can be learned using standard reinforcement learning techniques. In this section, we present this MDP formulation in the context of an image classification task, but it may be applied in any learning setting.

Definition 3 (MDP) *A* Markov Decision Process (MDP) *is a* 6*-tuple* $\mathcal{E} = (\mathcal{S}, \mathbb{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathrm{I}_0)$*, where* $\mathcal{S} \subseteq \mathbb{R}^n$ *is the set of states,* $\mathbb{A} \subseteq \mathbb{R}^m$ *is the set of actions,* $\mathcal{P}(s'|s, a)$ *specifies the probability of transitioning from state* s *to* s' *on action* a*,* $\mathcal{R}(s, a)$ *is the reward returned when taking action* a *from state* s*,* $\gamma \in [0, 1)$ *is the discount factor, and* I_0 *is the initial state distribution.*

State Representation. The environment has access to a set $\mathbf{V}_c \subset \mathcal{X}$, which is a set of possibly corrupted images. A state of the MDP is a compressed representation of this set $\mathbf{V}_c$, capturing the type of corruptions in an image. Let us assume that this projection is captured by some function $\mathbb{F}_R : \mathcal{A}_{\mathcal{X}} \to \mathfrak{R}$, where $\mathfrak{R}$ is the space of representations for a set of images. We want a representation $r = \mathbb{F}_R(\mathbf{V}_c)$ to be rich enough that a policy can decipher the appropriate choice of action in $\mathbb{A}$, but also compact enough that it is possible to learn a policy within a few episodes. Typically, a smaller state space size leads to faster convergence for reinforcement learning algorithms.

In this paper, for a set of images $\mathbf{V}_c$, we select a 3-dimensional state representation that measures the average brightness, standard deviation, and entropy of the images in $\mathbf{V}_c$. We convert each image x to grayscale, then obtain the discrete wavelet transform and compute its average brightness $B(x)$, standard deviation $S(x)$, and entropy $E(x)$. The state representation is an average of all these values across the images,

$$\mathbb{F}_R(\mathbf{V}_c) = \tfrac{1}{|\mathbf{V}_c|} \textstyle\sum_{x \in \mathbf{V}_c} \left[B(x), \quad S(x), \quad E(x) \right]^\top. \tag{2}$$

Actions and Transitions. The set of actions $\mathbb{A} \subseteq \mathbb{S}$ is a set of semantic preserving transforms from which the learner chooses to maximize some reward. For an example, see the action set

selected for the ImageNet-C benchmark in Appendix E. Hence, capturing Equation 1 as a reward leads the agent to pick actions that mitigate the current corruption to some extent. Transitions model the effect of applying a transform from $\mathbb{S}$ to a possibly corrupted set $\mathbf{V}_c$. With slight abuse of notation, we use $\mathbb{T}(\mathbf{V}_c)$ to denote set $\{v' : v' = \mathbb{T}(v), v \in \mathbf{V}_c\}$.

Computing Reward. As shown in Figure 3, computing the reward for a set of images $\mathbf{V}_c$ corresponds to measuring the distance from a clean validation set $\mathbf{V}$. Ideally, the distance between the distributions from which $\mathbf{V}_c$ and $\mathbf{V}$ are drawn would be measured, but this is difficult without knowledge of the source distributions. Instead, we use an empirical estimate of the Wasserstein distance (Bonneel et al., 2011) to compute the distributional distance between sets $\mathbf{V}_c$ and $\mathbf{V}$. It should be noted that a variety of distance functions can be employed here. We explore alternatives in Appendix A.

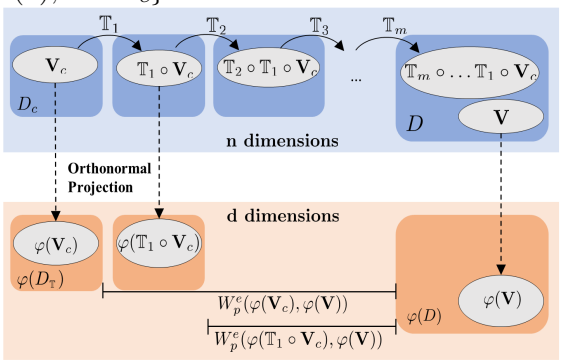

Figure 3: Operation of SuperStAR. Starting from $\mathbf{V}_c$, the algorithm selects a sequence of transforms $\mathcal{T}$ which move $\mathbf{V}_c$ closer to the original distribution $\mathbf{V}$. During the sequence, the orthonormal projections $\varphi(\mathbf{V})$ and $\varphi(\mathbf{V}_c)$ are used to compute the Wasserstein distance $W_p(\varphi(\mathbf{V}_c), \varphi(\mathbf{V}))$. See Section 6 for details.

In practice the policy might not be able to reduce the W_p^e to a level such that the classifier completely recovers the loss in accuracy. One reason for this is the possible non-existence of the inverse of the corruption transform. Another possible issue is that a transformation may overly alter the image such that the classifier performs poorly. Although we can combat against this by ensuring that actions make incremental changes to the image, this is hard to control. We therefore add a regularizer to the Wasserstein distance that penalizes excessive changes to the image. This is achieved using a visual similarity between pairs of images known as $ssim$ (Wang et al., 2004).

Given $\lambda > 0$ and $0 \leq \omega < 1$, the reward function is given by

$$R(s_t, a_t) = -W_p^e(\mathbf{V}, \mathbb{F}_R^{-1}(s_t)) + \lambda L_S(\mathbb{F}_R^{-1}(s_0), \mathbb{F}_R^{-1}(s_{t+1})), \tag{3}$$

where,

$$L_S(\mathbf{X}, \mathbf{Y}) = \begin{cases} \log(1 - ssim(\mathbf{X}, \mathbf{Y})), & ssim(\mathbf{X}, \mathbf{Y}) < \omega \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Note that the regularization hyperparameter λ influences the level of aggression in correction. For example, selecting a large λ favors actions with minimal influence on the data.

Initial State. At test time, the initial state of the MDP is produced by a random environment corruption from the set $\mathbb{S}$ that the system is subjected to. In reality the designer does not have access to any of these corrupting transforms. Hence, at training time we train a policy network to reverse a set of *surrogate* corruptions from $\mathbb{S}$, with the hope that some of these transfer at inference time to an unseen set of corruptions. An example surrogate corruption set and selection methodology for the ImageNet-C benchmark is given in Appendix E. We pick uniformly randomly from a finite set of $\mathsf{S}_c \subset \mathbb{S}$ of these surrogate corruptions to sample the initial state I_0.

Standard RL techniques can be applied to learn a policy $\pi : \mathcal{S} \rightarrow \mathbb{A}$ for the MDP $\mathcal{E}$, which is a strategy to recover the distribution shift. The value of a state for policy π is the expected return $\mathbf{R}_\pi(s_0)$, starting from state s_0, while executing the policy π at every step, $a_t = \pi(s_t)$. The optimal policy π^* maximizes the reward starting from the initial state distribution – i.e., $\pi^* = \arg\max V^{\mathrm{I}_0}(\pi)$, where $V^{\mathrm{I}_0}(\pi) = \mathbb{E}_{s_0 \in \mathrm{I}_0}[\mathbf{R}_\pi(s_0)]$.

5. Applying a Learned Policy for Detection and Correction

As described in Section 4, a DNN policy π is trained at design time such that, given a corrupted set $\mathbf{V}_c$ of i.i.d observations from corrupted distribution $D_{\mathbb{T}}$, the policy generates a finite composition $\mathcal{I}_k$ of $k < \infty$ transforms from $\mathbb{S}$. This solves the optimization problem in Equation 1. At design time, we assume the algorithm has access to a validation set $\mathbf{V} = \{v_1, \ldots, v_n\}$, where v_i is drawn i.i.d from the training distribution D.

At inference / run-time `SuperStAR` uses the corrupted set $\mathbf{V}_c = \{v_1^c, v_2^c, \ldots, v_n^c\}$ drawn i.i.d from $D_{\mathbb{T}}$, $\mathbb{T} \in \mathbb{S}$, and the policy π to select a correcting sequence $\mathcal{T} := \{\mathbb{T}_k, \mathbb{T}_{k-1}, \ldots \mathbb{T}_1\}$. We fix a maximum horizon k to keep the algorithm tractable at both learning and deployment. The procedure for selecting sequence $\mathcal{T}$ is presented in Algorithm 1.

Algorithm 1 Transformation Selection

Require: Validation set $\mathbf{V}$, corrupted set $\mathbf{V}_c$, policy π, horizon k, thresholds $\alpha, \beta \in (0, 1]$

1: $w_0 \leftarrow \tilde{W}(\mathbf{V}, \mathbf{V}_c)$
2: $\mathbf{V}_0 \leftarrow \mathbf{V}_c$
3: $T \leftarrow \{\ \}$
4: **for** $i = 1, \ldots, k$ **do**
5: **if** $\mathbf{V}_i$ is inoperable **then**
6: return T
7: **end if**
8: $\mathbb{T}_i \leftarrow \pi(\mathbf{V}_{i-1})$
9: $\mathbf{V}_i \leftarrow \mathbb{T}_i(\mathbf{V}_{i-1})$
10: $w_i \leftarrow \tilde{W}(\mathbf{V}, \mathbf{V}_i)$
11: **if** $w_i \leq \alpha\, w_0 \vee w_i \geq \beta\, w_{i-1}$ **then**
12: return T
13: **end if**
14: $T \leftarrow \{\mathbb{T}_i\} \cup T$
15: **end for**
16: Return T

In Line 1, Algorithm 1 uses the estimator $\tilde{W}$ to estimate the initial Wasserstein distance. Next, it runs policy π for k steps (Lines 5−16), where it iteratively applies the transform picked by the policy to update the corrupted set. The algorithm uses a lower dimensional state representation of $\mathbf{V}$ for evaluating the policy and for the estimator $\tilde{W}$ (see Section 4). The algorithm collects and returns this set of transforms in T.

There are two stopping criteria to prevent the selection of damaging transforms. The first, in line 5, checks for a minimum condition of operability on the set (see Section 5.1). The second, in Line 11, guards against overly transforming images to diminishing returns. It also hedges against the chance that the distribution shift is not semantic preserving, which is always possible in reality. Hence, we pause the policy when the Wasserstein distance decreases beyond a threshold.

5.1. Minimum Condition for Operability

The optimal policy π^* selects transformations that maximize the reward function, consisting of some distance function (e.g., the Wasserstein distance) plus a regularizing factor. The efficacy of such a policy thus depends on the distance function being a reliable estimate of a classifier's performance on the given data. We define an operable corrupted set $\mathbf{V}_c$ as follows.

Definition 4 (Operable Set) *For some distance function d, classifier f, validation set $\mathbf{V}$, and set of transformations $\{\mathbb{T}_i\}_{i=1}^t$, a set $\mathbf{V}_c = \{v_1^c, c_2^c, \ldots, v_n^c\}$ is operable iff $\{d(\mathbf{V}, \mathbb{T}_i(\mathbf{V}_c))\}_{i=1}^t$ is negatively linearly correlated with $\{\frac{1}{n}\sum_{j=1}^n \mathbb{1}[f(\mathbb{T}_i(v_j^c)) = \mathcal{C}^*(\mathbb{T}_i(v_j^c))]\}_{i=1}^t$.*

Since the accuracy of the classifier we wish to adapt is unknown on inference-time data, operability is in practice difficult to evaluate. However, we can instead predict operability from the state representation $\mathbb{F}_R(\mathbf{V}_c)$ using a simple binary classifier, which is trained on surrogate corrupted data generated at design time. Label generation can be performed by evaluating the distance function and accuracies on these surrogate shifts, where $\mathbb{T}_i(\mathbf{V}_c)$ are variations in the severity of shift

approximating the effect of applying transformations. To evaluate accuracy, the selected classifier f can be unique from the classifier we wish to adapt. This is motivated by the model collinearity phenomenon (Mania and Sra, 2020), in which the relative accuracy on different distributions of data is often the same across multiple classifiers.

6. Dimensionality Reduction

The empirical estimate of the Wasserstein distance converges in sample size to the true Wasserstein distance slowly in large dimensions (Ramdas et al., 2017). Ideally, by reducing the dimensionality of the sample data, fewer samples are needed to achieve an accurate estimate of the Wasserstein distance. We present rigorous theoretical justification that orthonormal projection is a reduction technique that preserves characteristics of the sample data at a distributional level.

Let $m, n \in \mathbb{N}$ and $p \in [1, \infty]$. Call $M(\mathbb{R}^n)$ and $M(\mathbb{R}^m)$ the spaces of probability measures on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. Denote by $M^p(\mathbb{R}^n)$ and $M^p(\mathbb{R}^m)$ the spaces of probability measures having finite p-th moment on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively (here $p = \infty$ is interpreted in the limiting sense of essential supremum). For convenience, we consider only probability measures with densities, so that we do not have to check which measure is absolutely continuous to which other measure (Cai and Lim, 2022, Section III).

Suppose $m \leq n$ and consider the Stiefel manifold on $m \times n$ matrices with orthonormal rows.

$$O(m, n) := \{V \in \mathbb{R}^{m \times n} : VV^\top = I_d\}.$$

For any $V \in O(m, n)$ and $b \in \mathbb{R}^m$, let

$$\varphi_{V,b} : \mathbb{R}^n \to \mathbb{R}^m, \quad x \mapsto \varphi_{V,b}(x) := Vx + b,$$

and for any $\mu \in M(\mathbb{R}^n)$, let $\varphi_{V,b}(\mu) := \mu \circ \varphi_{V,b}^{-1}$ be the pushforward measure. This can be seen as a projection of μ onto the smaller dimensional space $\mathbb{R}^m$, and we call it a *Cai-Lim projection*; it is not unique: it depends on the choice of V and b. Recall then the definition of the p-Wasserstein distance between $\mu, \nu \in M^p(\mathbb{R}^n)$: $W_p(\mu, \nu) := \left[\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathbb{R}^{2n}} \|x - y\|_2^p \, \mathrm{d}\gamma(x, y)\right]^{\frac{1}{p}}$, where $\| \cdot \|_2$ denotes the Euclidean norm and $\Gamma(\mu, \nu) := \{\gamma \in M(\mathbb{R}^{2n}) : \pi_1^n(\gamma) = \nu, \pi_2^n(\gamma) = \mu\}$ is the set of couplings between μ and ν, where π_1^n is the projection onto the first n coordinates and π_2^n is the projection onto the last n coordinates. The following is an important result.

Lemma 5 ***(Cai and Lim, 2022, Lemma II.1)*** *Let $m, n \in \mathbb{N}$ and $p \in [1, \infty]$, and assume $m \leq n$. For any $\mu, \nu \in M^p(\mathbb{R}^n)$, any $V \in O(m, n)$, and any $b \in \mathbb{R}^m$, we have that*

$$W_p\left(\varphi_{V,b}(\mu), \varphi_{V,b}(\nu)\right) \leq W_p\left(\mu, \nu\right).$$

This can be interpreted as "losing some information" when performing a Cai-Lim projection: in smaller dimensional spaces, distributions μ and ν seem to be closer than they actually are. This is an inevitable byproduct of any projection operation. Lemma 5 implies the following corollary.

Corollary 6 *Let $m, n \in \mathbb{N}$ and $p \in [1, \infty]$, and assume $m \leq n$. Consider $\mu, \nu, \rho, \zeta \in M^p(\mathbb{R}^n)$, and pick any $V \in O(m, n)$ and any $b \in \mathbb{R}^m$. Suppose $W_p(\mu, \nu) \geq W_p(\rho, \zeta)$. Then, there exists $\varepsilon > 0$ such that if $W_p\left(\mu, \nu\right) - W_p\left(\varphi_{V,b}(\mu), \varphi_{V,b}(\nu)\right) \leq \varepsilon$, then*

$$W_p\left(\varphi_{V,b}(\mu), \varphi_{V,b}(\nu)\right) \geq W_p\left(\varphi_{V,b}(\rho), \varphi_{V,b}(\zeta)\right).$$

Proof Set $\varepsilon = W_p(\rho, \zeta) - W_p(\varphi_{V,b}(\rho), \varphi_{V,b}(\zeta))$. The result follows immediately by Lemma 5. ■

Corollary 6 states the following. If we "do not lose too much information" when performing a Cai-Lim projection of the two farthest apart distributions, then the inequality $W_p(\mu, \nu) \geq W_p(\rho, \zeta)$ between the original distribution is preserved between their projections. A more intuitive discussion, as well as empirical justification, can be found in Appendices B and C.

7. Related Work

Distribution shifts can make deep learning models act dangerously in the real world (Narasimhamurthy et al., 2019). Some offline techniques aim to improve classification robustness to distribution shift at training time. These include data augmentation, which expands the experiences that a model would be exposed to during training (Hendrycks et al., 2019; Erichson et al., 2022; Verma et al., 2019; Yun et al., 2019; Kim et al., 2020b; Hendrycks et al., 2020). Another offline approach is domain adaptation, in which models are adapted to perform well on unlabeled data in some new distribution (Ben-David et al., 2010; Bousmalis et al., 2017; Hoffman et al., 2018). Offline approaches lack flexibility to shifts unforeseen at train time, but they can be combined with online methods like `SuperStAR` to further improve performance.

Alternate techniques aim to handle distribution shift online. For example, test-time adaptation methods perform additional training at inference time in response to incoming data (Gandelsman et al., 2022; Wang et al., 2022; Mummadi et al., 2021). Test time augmentation approaches instead apply transformations to the input, then ensemble predictions if multiple transformations are applied (Simonyan and Zisserman, 2014; Krizhevsky et al., 2012; He et al., 2016; Guo et al., 2017; Mummadi et al., 2021; Kim et al., 2020a; Lyzhov et al., 2020). In contrast, `SuperStAR` selects transformations online without querying the model. Furthermore, our work uniquely identifies MDPs as an equivalent formulation of the transformation selection problem, enabling the application of standard RL techniques. We further explore related work in Appendix D.

8. Application: ImageNet-C

The ImageNet-C dataset (Hendrycks and Dietterich, 2019) is constructed from ImageNet samples corrupted by 19 semantic preserving transformations. We deploy the learned policy π in `SuperStAR` to correct ImageNet-C corruptions, and we evaluate the accuracies of Resnet-50 classifiers with and without correction. We evaluate a baseline classifier trained without data augmentation and classifiers trained with data augmentation through AugMix, NoisyMix, DeepAugment, DeepAugment with Augmix, and Puzzlemix. We also evaluate the ability to generalize outside of the ImageNet-C benchmark. Experiment details can be found in Appendix E.[3]

ImageNet-C. Table 1 summarizes the classifier accuracy improvements from applying `SuperStAR` to ImageNet-C. For brevity, we exclude from Table 1 shifts for which `SuperStAR` incurs no change in accuracy (see Appendix F for full table). Corrections via our `SuperStAR` algorithm lead to accuracy improvements for a majority of shifts, with maximum improvement of 14.21% (averaged across all five severity levels). In general accuracy improvements are greater for higher severities (due to space constraints, per-severity accuracy improvements are shown in Appendix F). Furthermore, when combined with data augmentation, `SuperStAR` often leads to higher accuracies than

3. Our code can be found at https://github.com/vwlin/SuperStAR.

Table 1: Average accuracies (%) on each ImageNet-C shift with and without SuperStAR for ResNet-50 classifiers. Accuracy improvement is denoted by Δ = R (recovered) − S (shifted). Values are over 5 severity levels with 3 trials each. "gaussian" refers to Gaussian noise.

	No Data Aug			AugMix			NoisyMix			DeepAugment			DeepAug+AugMix			PuzzleMix		
shift	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ
none	74.52	74.52	0.00	75.94	75.94	0.00	76.22	76.22	0.00	75.86	75.86	0.00	75.26	75.26	0.00	75.63	75.63	0.00
gaussian	31.11	43.44	12.33	41.90	50.87	8.98	52.71	55.51	2.80	59.07	59.48	0.41	55.39	61.43	6.05	41.48	46.94	5.46
shot	28.61	42.81	14.21	41.78	50.93	9.15	51.81	55.38	3.57	58.21	58.46	1.24	55.76	62.37	6.61	37.39	45.56	7.77
impulse	26.57	39.36	12.79	38.78	47.49	8.71	50.73	53.37	2.64	58.61	58.38	-0.23	55.16	60.67	5.50	35.28	42.82	7.54
snow	30.51	29.28	-1.13	37.89	36.85	-1.04	43.20	41.52	-1.68	41.71	39.91	-1.80	47.68	46.36	-1.32	39.48	37.74	-1.75
frost	35.16	35.07	-0.09	41.39	41.24	-0.15	50.05	49.46	-0.59	46.87	46.08	-0.78	51.21	50.26	-0.95	46.96	45.75	-1.21
brightness	65.17	65.72	0.55	67.35	68.46	1.11	68.82	69.87	1.05	69.04	69.73	0.69	69.42	70.18	0.76	69.59	69.67	0.08
contrast	35.56	37.69	2.14	48.96	49.85	0.89	50.37	52.74	2.37	44.89	48.23	3.33	56.01	57.40	1.39	50.56	52.87	2.30
speckle	36.09	49.26	13.18	50.61	56.94	6.33	57.67	60.88	3.22	62.21	63.81	1.59	60.93	65.66	4.74	42.24	51.92	9.68
spatter	46.65	46.44	-0.20	53.25	52.93	-0.32	57.63	57.34	-0.29	53.74	53.58	-0.16	57.75	57.61	-0.14	53.27	52.95	-0.32
saturate	59.00	59.17	0.17	61.42	61.89	0.47	63.48	64.00	0.52	64.59	64.81	0.22	65.79	66.12	0.33	65.96	65.60	-0.37

SuperStAR or data augmentation alone. In the cases of no shift and the shifts not shown in Table 1, SuperStAR refrains from taking any action and does not affect accuracy. This is examined further in Appendix G.

Interestingly, for a hyperparameter selection that allows some increase in Wasserstein distance at each step ($\beta = 1.12$), we find that SuperStAR selects a non-trivial 5-action sequence of transformations for contrast shift severity level 5, which incrementally increases the accuracy of the AugMix classifier. Figure 2 shows a sample image with the applied transformations and resulting accuracies. The transformations incur a noticeable change in the image.

Generalization beyond ImageNet-C. We also evaluate the ability of SuperStAR to generalize outside of the ImageNet-C benchmark. 1) We construct composite ImageNet-C shifts from pairs of the ImageNet-C corruptions for which SuperStAR improves accuracy. To each shift, we reapply our operability classifier and policy network without retraining. Table 2 shows the average classifier accuracy improvements from SuperStAR. For nearly all shifts, our method improves classifier accuracy, with a maximum improvement in average accuracy of 9.81% (see Appendix H for severity level breakdowns). 2) We also apply our method to CIFAR-100-C (Hendrycks and Dietterich, 2019), an analagous benchmark to ImageNet-C. We use the pretrained policy network and retrain only the operability classifier on the surrogate corruptions regenerated for CIFAR-100. We evaluate on a variety of Wide ResNets trained with data augmentation (AugMix, NoisyMix, and PuzzleMix) and without (baseline).[4] Appendix I contains further details. Table 3 shows the accuracy improvements from applying SuperStAR to CIFAR-100-C. Without any retraining of the policy network, SuperStAR improves average accuracy by up to 8.25% (see Appendix J for full table and severity level breakdowns). In some cases, such as impulse noise, our method decreases accuracy for the classifiers trained with data augmentation, while increasing that of the baseline classifier. We attribute this to the operability classifier mislabeling such shifts as operable.

Overall, SuperStAR demonstrates an ability to dynamically respond to distribution shift online, selecting context-appropriate actions (or inaction) and significantly improving classification accuracy on ImageNet-C for a variety of corruptions. In some cases, SuperStAR identifies complex sequences of corrective transformations. We attribute this strong performance largely to our selection of surrogate shifts (see Appendix K for more discussion). SuperStAR also generalizes to distribution shifts outside of the ImageNet-C benchmark, without retraining the policy network.

4. DeepAugment and DeepAugment with Augmix are not evaluated on CIFAR-100 in the original publications.

Table 2: Average accuracies (%) on each composite ImageNet-C shift with and without SuperStAR for ResNet-50 classifiers. Composites are combinations of Gaussian noise (GN), shot noise (ShN), impulse noise (IN), speckle noise (SpN) with brightness (B), contrast (C), saturate (S). Accuracy improvement is denoted Δ = R (recovered) − S (shifted). Values are over 5 severity levels with 3 trials each.

	No Data Aug			AugMix			NoisyMix			DeepAugment			DeepAug+AugMix			PuzzleMix		
shift	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ
GN + B	23.75	30.65	6.90	29.03	37.20	8.17	42.87	44.37	1.50	51.29	49.80	-1.49	43.03	52.17	9.11	33.25	35.15	1.90
GN + C	22.39	22.39	0.00	29.40	29.40	0.00	35.96	35.96	0.00	34.00	34.00	0.00	41.31	41.31	0.00	32.46	32.46	0.00
GN + S	19.35	26.37	7.02	26.65	33.43	6.79	37.13	38.50	1.37	42.83	44.50	1.68	42.41	50.82	8.41	28.96	30.98	2.02
IN + B	22.03	30.08	8.04	29.62	37.91	8.29	43.66	45.89	2.24	52.47	52.90	0.43	44.14	53.95	9.81	32.13	35.65	3.52
IN + C	18.41	18.41	0.00	25.89	25.89	0.00	33.05	33.05	0.00	31.61	31.61	0.00	39.72	39.72	0.00	26.27	26.27	0.00
IN + S	17.59	24.61	7.02	25.24	30.37	5.12	34.23	36.07	1.84	40.96	42.34	1.37	43.58	49.05	5.47	25.74	29.37	3.62
ShN + B	22.32	30.08	7.76	28.38	35.82	7.44	40.26	41.64	1.38	49.44	47.65	-1.79	41.38	50.61	9.23	31.84	33.63	1.79
ShN + C	21.13	21.03	-0.10	28.36	27.94	-0.42	35.22	34.98	-0.24	34.26	33.96	-0.30	41.05	40.76	-0.29	30.13	30.26	**0.13**
ShN + S	18.49	26.03	7.54	27.33	33.74	6.40	37.58	39.11	1.52	42.47	44.84	2.37	44.04	51.26	7.22	26.33	30.03	3.70
SpN + B	27.20	32.43	5.23	32.64	38.55	5.91	43.60	45.64	2.05	52.77	52.37	-0.41	45.82	52.99	7.17	35.80	37.81	2.01
SpN + C	24.00	24.18	0.18	32.70	32.58	-0.12	39.21	39.57	0.36	36.30	36.98	0.68	44.19	44.49	0.30	33.40	33.74	0.34
SpN + C	23.80	31.36	7.56	35.87	40.22	4.35	45.04	45.98	0.93	47.64	50.87	3.23	50.25	55.98	5.72	30.55	35.94	5.39

Table 3: Average accuracies (%) on each CIFAR-100-C shift with and without SuperStAR for Wide ResNet classifiers. Accuracy improvement is denoted by Δ = R (recovered) − S (shifted). Values are over 5 severity levels with 3 trials each.

	No Data Aug			AugMix			NoisyMix			PuzzleMix		
shift	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ
none	81.13	81.13	0.00	76.28	76.28	0.00	81.29	81.29	0.00	84.01	84.01	0.00
gaussian noise	21.12	26.81	5.70	47.89	51.07	3.18	65.91	66.34	0.43	20.87	28.18	7.31
shot noise	29.96	36.34	6.38	55.69	58.24	2.55	70.39	70.72	0.33	31.12	39.37	8.25
impulse noise	19.21	26.09	6.88	59.68	59.09	-0.59	79.72	76.08	-3.64	37.18	37.01	-0.17
glass blur	20.68	26.61	5.93	54.08	56.09	2.00	58.82	60.48	1.66	31.07	37.62	6.55
speckle noise	31.58	35.76	4.18	58.11	59.33	1.23	71.67	71.57	-0.09	33.96	38.94	4.98
spatter	61.23	62.23	1.00	72.28	71.25	-1.02	78.11	77.44	-0.67	79.73	78.90	-0.83
saturate	68.82	68.88	0.06	64.52	64.77	0.25	69.83	70.21	0.39	72.93	72.99	0.05

Finally, we note that although promising, SuperStAR is limited in its speed of response and its reliance on appropriately selected actions and surrogate corruptions. We further discuss these limitations and more in Appendix L.

9. Conclusion

In this work we presented SuperStAR, which uses control for learned models to detect and recover from distribution shift. SuperStAR uses the Wasserstein distance (with a theoretically-sound approach for dimensionality reduction using orthonormal projections) to detect distribution shifts and select recovery actions from a library of image correction techniques. We formulate this action selection problem as a Markov decision process, and we train the policy for computing actions using reinforcement learning. To hedge against harmful actions, we employ a binary classifier to check a minimum condition for our method to operate on corrupted data. We applied our approach to various classifiers on the ImageNet-C dataset, and we obtained significant accuracy improvements when compared to the classifiers alone. Finally, we showed that SuperStAR generalizes to composite ImageNet-C shifts and CIFAR-100-C with no retraining of the policy. Expansion of the action library and additional tuning can lead to further improvements on these benchmarks.

Acknowledgments

This research was supported in part by ARO W911NF-20-1-0080. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Christina Baek, Yiding Jiang, Aditi Raghunathan, and J Zico Kolter. Agreement-on-the-line: Predicting the performance of neural networks under distribution shift. *Advances in Neural Information Processing Systems*, 35:19274–19289, 2022.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79:151–175, 2010.

Nicolas Bonneel, Michiel Van De Panne, Sylvain Paris, and Wolfgang Heidrich. Displacement interpolation using lagrangian mass transport. In *Proceedings of the 2011 SIGGRAPH Asia conference*, pages 1–12, 2011.

Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

Yuhang Cai and Lek-Heng Lim. Distances between probability distributions of different dimensions. *IEEE Transactions on Information Theory*, 2022.

S Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE transactions on image processing*, 9(9):1532–1546, 2000.

David L Donoho and Jain M Johnstone. Ideal spatial adaptation by wavelet shrinkage. *biometrika*, 81(3):425–455, 1994.

N. Benjamin Erichson, Soon Hoe Lim, Francisco Utrera, Winnie Xu, Ziang Cao, and Michael W. Mahoney. Noisymix: Boosting robustness by combining data augmentations, stability training, and noise injections. *CoRR*, abs/2202.01263, 2022. URL https://arxiv.org/abs/2202.01263.

Yossi Gandelsman, Yu Sun, Xinlei Chen, and Alexei Efros. Test-time training with masked autoencoders. *Advances in Neural Information Processing Systems*, 35:29374–29385, 2022.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Dan Hendrycks and Kevin Gimpel. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, October 2018. URL http://arxiv.org/abs/1610.02136. arXiv:1610.02136 [cs].

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, 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. *CoRR*, abs/2006.16241, 2020. URL https://arxiv.org/abs/2006.16241.

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International conference on machine learning*, pages 1989–1998. Pmlr, 2018.

Leonid Vital'evič Kantorovič. On a problem of Monge. *Zap. Nauchn. Sem. S.-Peterburg. Otdel. Mat. Inst. Steklov. (POMI)*, 312(11):15–16, 2004.

Ramneet Kaur, Kaustubh Sridhar, Sangdon Park, Susmit Jha, Anirban Roy, Oleg Sokolsky, and Insup Lee. CODiT: Conformal Out-of-Distribution Detection in Time-Series Data, July 2022. URL http://arxiv.org/abs/2207.11769. arXiv:2207.11769 [cs].

Ildoo Kim, Younghoon Kim, and Sungwoong Kim. Learning loss for test-time augmentation. *Advances in Neural Information Processing Systems*, 33:4163–4174, 2020a.

Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. *CoRR*, abs/2009.06962, 2020b. URL https://arxiv.org/abs/2009.06962.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, May 2017. URL http://arxiv.org/abs/1609.04802. arXiv:1609.04802 [cs, stat] version: 5.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples, February 2018. URL http://arxiv.org/abs/1711.09325. arXiv:1711.09325 [cs, stat].

Alexander Lyzhov, Yuliya Molchanova, Arsenii Ashukha, Dmitry Molchanov, and Dmitry Vetrov. Greedy policy search: A simple baseline for learnable test-time augmentation. In *Conference on Uncertainty in Artificial Intelligence*, pages 1308–1317. PMLR, 2020.

Horia Mania and Suvrit Sra. Why do classifier accuracies show linear trends under distribution shift? *arXiv preprint arXiv:2012.15483*, 2020.

John P Miller, Rohan Taori, Aditi Raghunathan, Shiori Sagawa, Pang Wei Koh, Vaishaal Shankar, Percy Liang, Yair Carmon, and Ludwig Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*, pages 7721–7735. PMLR, 2021.

Chaithanya Kumar Mummadi, Robin Hutmacher, Kilian Rambach, Evgeny Levinkov, Thomas Brox, and Jan Hendrik Metzen. Test-time adaptation to distribution shift by confidence maximization and input transformation. *arXiv preprint arXiv:2106.14999*, 2021.

Monal Narasimhamurthy, Taisa Kushner, Souradeep Dutta, and Sriram Sankaranarayanan. Verifying conformance of neural network models: Invited paper. In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, 2019. doi: 10.1109/ICCAD45719.2019.8942151.

Stephan Rabanser, Stephan Günnemann, and Zachary C. Lipton. Failing Loudly: An Empirical Study of Methods for Detecting Dataset Shift, October 2019. URL http://arxiv.org/abs/1810.11953. arXiv:1810.11953 [cs, stat].

Aaditya Ramdas, Nicolás García Trillos, and Marco Cuturi. On wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019.

Cedric Villani. *Optimal Transport: Old and New*. Springer Berlin, Heidelberg, 2009.

Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7201–7211, 2022.

Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004. doi: 10.1109/TIP.2003.819861.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *CoRR*, abs/1905.04899, 2019. URL http://arxiv.org/abs/1905.04899.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Learning Enriched Features for Real Image Restoration and Enhancement. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, volume 12370, pages 492–511. Springer International Publishing, Cham, 2020. ISBN 978-3-030-58594-5 978-3-030-58595-2. doi: 10.1007/978-3-030-58595-2_30. URL https://link.springer.com/10.1007/978-3-030-58595-2_30. Series Title: Lecture Notes in Computer Science.

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Multi-Stage Progressive Image Restoration, March 2021. URL http://arxiv.org/abs/2102.02808. arXiv:2102.02808 [cs].

Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Learning Enriched Features for Fast Image Restoration and Enhancement, April 2022. URL http://arxiv.org/abs/2205.01649. arXiv:2205.01649 [cs, eess].

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, July 2017. ISSN 1941-0042. doi: 10.1109/TIP.2017.2662206. Conference Name: IEEE Transactions on Image Processing.

Kaihao Zhang, Wenqi Ren, Wenhan Luo, Wei-Sheng Lai, Bjorn Stenger, Ming-Hsuan Yang, and Hongdong Li. Deep Image Deblurring: A Survey, May 2022. URL http://arxiv.org/abs/2201.10700. arXiv:2201.10700 [cs].

Appendix A. Alternative Distance Functions

Although we employ the Wasserstein distance to estimate the distance between a clean validation set of images and a corrupted set, a variety of alternatives can be used instead. For example, another natural choice for measuring the distance between two distributions is an f-divergence, such as the popular Total Variation (TV) distance, Kullback-Leibler (KL) divergence, and Jensen-Shannon (JS) divergence. In these cases, orthonormal projection can again be applied for dimensionality reduction, as equivalents to Proposition 6 hold for f-divergences (Cai and Lim, 2022). We note, however, that the TV distance, KL divergence, and JS divergence saturate when comparing distributions with disjoint supports, leading to potentially uninformative rewards when training the policy network. More reading on the TV distance, KL divergence, and JS divergence compared to the Wasserstein distance can be found in Arjovsky et al. (2017).

The previously mentioned methods for estimating the distance between two sets of images rely solely on the data itself, agnostic of the model performing inference on said data. While such distance functions lend flexibility across models and modalities, they fail to capitalize on information the model itself can provide. An alternative "distance function" that exploits the model is Agreement-on-the-Line (ALine-D), an approach for estimating the accuracy of a classifier on out-of-distribution data (Baek et al., 2022). ALine-D builds on the empirical observation that many models exhibit strong linear correlation between in-distribution (ID) accuracy and out-of-distribution (OOD) accuracy (Miller et al., 2021). Combined with a similar phenomenon for the agreement among an ensemble of models on ID and OOD data, ALine-D estimates the accuracy of any ensemble member on OOD data, given only accuracies on ID data and agreements on ID/OOD data. This estimate can in turn be used to devise a distance function for use in our reward function, although incurring an additional computational burden from the evaluation of the ensemble. Ultimately, despite the benefits of ALine-D, we prioritize generalizability and instead select the Wasserstein distance.

Appendix B. An Intuitive View of Orthonormal Projection for the Wasserstein Distance

As discussed in the main text, we show that orthonormal projection is a reduction technique that preserves characteristics of the sample data at a distributional level. An intuitive motivation for this arises from a metric we refer to as the *Cai-Lim distance* (Cai and Lim, 2022).

We assume the same preliminaries as in Section 6 of the main text. Now denote by

$$\Phi^{-}(\mu, d) := \{\beta \in M(\mathbb{R}^m) : \varphi_{V,b}(\mu) = \beta, \\ \text{for some } V \in O(d, n), b \in \mathbb{R}^m\}$$

the set of Cai-Lim projections of μ onto $\mathbb{R}^m$. We call *Cai-Lim distance* between $\mu \in M^p(\mathbb{R}^n)$ and $\nu \in M^p(\mathbb{R}^m)$, $m \leq n$, the smallest p-Wasserstein distance between ν and a Cai-Lim projection of μ onto $\mathbb{R}^m$, for some $p \in [1, \infty]$. That is,

$$W_p^{CL}(\mu, \nu) := \inf_{\beta \in \Phi^{-}(\mu, d)} W_p(\beta, \nu). \tag{5}$$

We note that *Cai-Lim distance* is closely related to Monge and Kantorovich's formulations of the optimal transport problem (subsection B.1 below). A generalization of the Wasserstein distance, the Cai-Lim distance allows for measurements of distance between distributions of different dimensions. It follows directly from (5) that when an orthonormal projection is applied, $\nu = \varphi_{V,0}(\mu)$, the Cai-Lim distance is trivially zero, $W_p^{CL}(\mu, \nu) = 0$. This suggests that the orthonormal projection preserves information about the distance between two distributions.

B.1. Cai-Lim Distance and Optimal Transport

Monge's formulation of the optimal transport problem can be formulated as follows. Let $\mathcal{X}, \mathcal{Y}$ be two separable metric Radon spaces. Let $c : \mathcal{X} \times \mathcal{Y} \to [0, \infty]$ be a Borel-measurable function. Given probability measures μ on $\mathcal{X}$ and ν on $\mathcal{Y}$, Monge's formulation of the optimal transportation problem is to find a transport map $T : \mathcal{X} \to \mathcal{Y}$ that realizes the infimum

$$\inf_{T_\star(\mu) = \nu} \int_{\mathcal{X}} c(x, T(x)) \, \mathrm{d}\mu(x), \tag{6}$$

where $T_\star(\mu) \equiv \mu \circ T^{-1}$ is the pushforward of μ by T.

Monge's formulation of the optimal transportation problem can be ill-posed, because sometimes there is no T satisfying $T_\star(\mu) = \nu$. Equation (6) can be improved by adopting Kantorovich's formulation of the optimal transportation problem, which is to find a probability measure $\gamma \in \Gamma(\mu, \nu)$ that attains the infimum

$$\inf_{\gamma \in \Gamma(\mu, \nu)} \int_{\mathcal{X} \times \mathcal{Y}} c(x, y) \, \mathrm{d}\gamma(x, y), \tag{7}$$

where $\Gamma(\mu, \nu)$ is the set of joint probability measures on $\mathcal{X} \times \mathcal{Y}$ whose marginals are μ on $\mathcal{X}$ and ν on $\mathcal{Y}$. A minimizer for this problem always exists when the cost function c is lower semi-continuous and $\Gamma(\mu, \nu)$ is a tight collection of measures.

Our Cai-Lim distance formulation, Equation (5), can be written as

$$\inf_{\beta\in\Phi^{-}(\mu,d)}\left[\left(\inf_{\gamma\in\Gamma(\beta,\nu)}\int_{\mathbb{R}^{2d}}\|x-y\|_2^p\,\mathrm{d}\gamma(x,y)\right)^{1/p}\right]. \tag{8}$$

Therefore, (8) is a combination of (6) and (7). To see this, notice that $\mathcal{X}=\mathcal{Y}=\mathbb{R}^d$, and $(x,y)\mapsto c(x,y)=\|x-y\|_2^p$, for some $p\in[1,\infty]$. Hence, the inner part of (8) corresponds to Kantorovich's formulation of the optimal transportation problem. The outer $\inf$, instead, has a Mongenian flavor to it. With this, we mean that the probability measure β must be the one minimizing the p-Wasserstein distance between ν and all the elements in the set $\Phi^{-}(\mu,d)$ of Cai-Lim projections of μ onto $\mathbb{R}^d$, $d\leq n$. Because

$$\Phi^{-}(\mu,d):=\{\beta\in M(\mathbb{R}^m):\varphi_{V,b}(\mu)=\beta,\ \text{for some } V\in O(d,n), b\in\mathbb{R}^m\},$$

and $\varphi_{V,b}(\mu)\equiv\varphi_{V,b}^{-1}{}_{\star}(\mu):=\mu\circ\varphi_{V,b}^{-1}$ is the pushforward of μ by $\varphi_{V,b}^{-1}$, this reminds us (heuristically) of Monge's formulation. The fact that (8) has a solution is guaranteed by Cai and Lim (2022, Section II).

Some further references can be found at Kantorovič (2004); Villani (2009).

Appendix C. Empirical Evaluation of Orthonormal Projection

Figures 4, 5, and 6 compare the empirical Wasserstein distance using orthonomal projection to Gaussian random projection and sparse random projection when MNIST is perturbed with additive Gaussian noise, additive impulse noise, and Gaussian blur, respectively. We find that orthonormal projection better preserves distributional information than the alternatives.

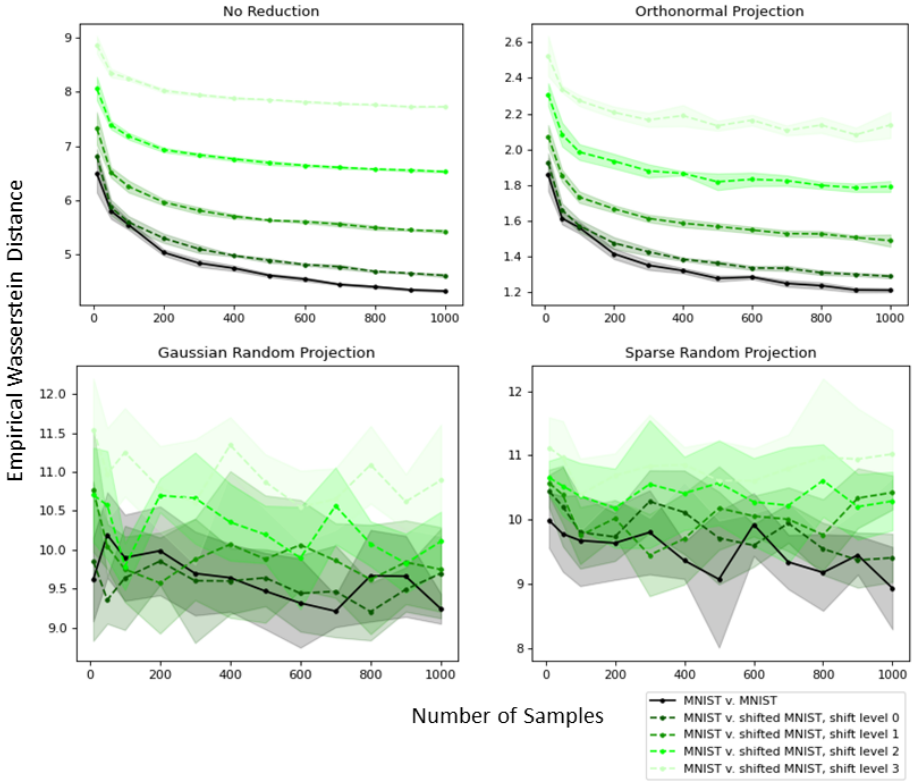

Figure 4: Empirical Wasserstein distance between MNIST and MNIST with varied levels of additive Gaussian noise, measured over a range of sample sizes. Curves are taken over 5 trials. MNIST samples are downsampled to 24×24, flattened, and projected to 50 dimensions. Orthonormal projection better preserves distributional information than Gaussian random projection and sparse random projection.

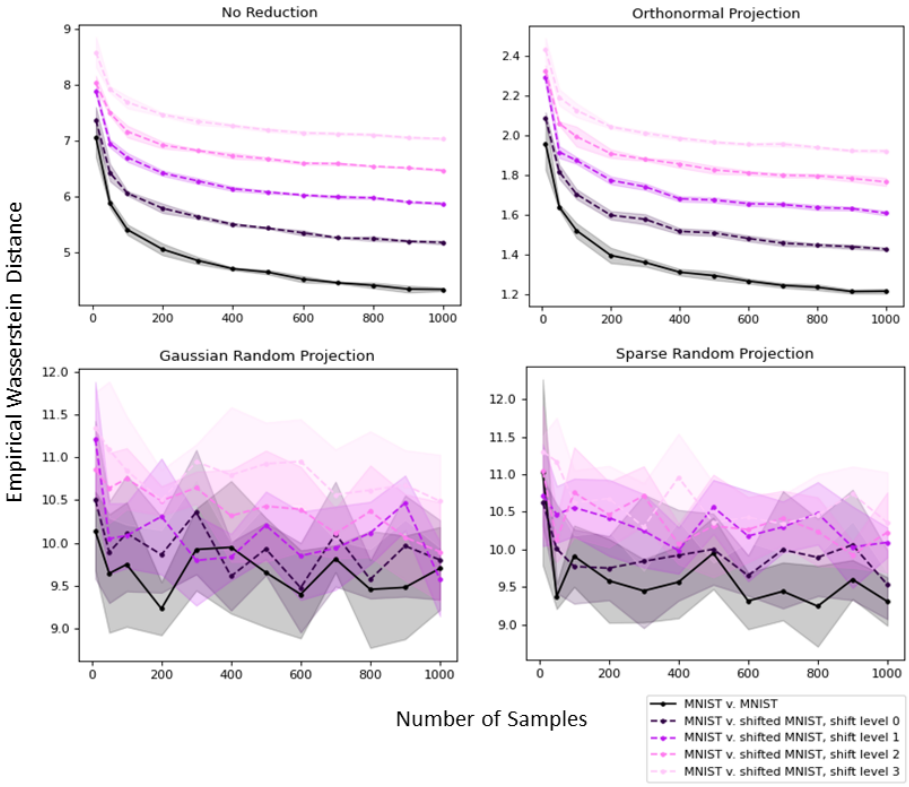

Figure 5: Empirical Wasserstein distance between MNIST and MNIST with varied levels of additive impulse noise, measured over a range of sample sizes. Curves are taken over 5 trials. MNIST samples are downsampled to 24×24, flattened, and projected to 50 dimensions. Orthonormal projection better preserves distributional information than Gaussian random projection and sparse random projection.

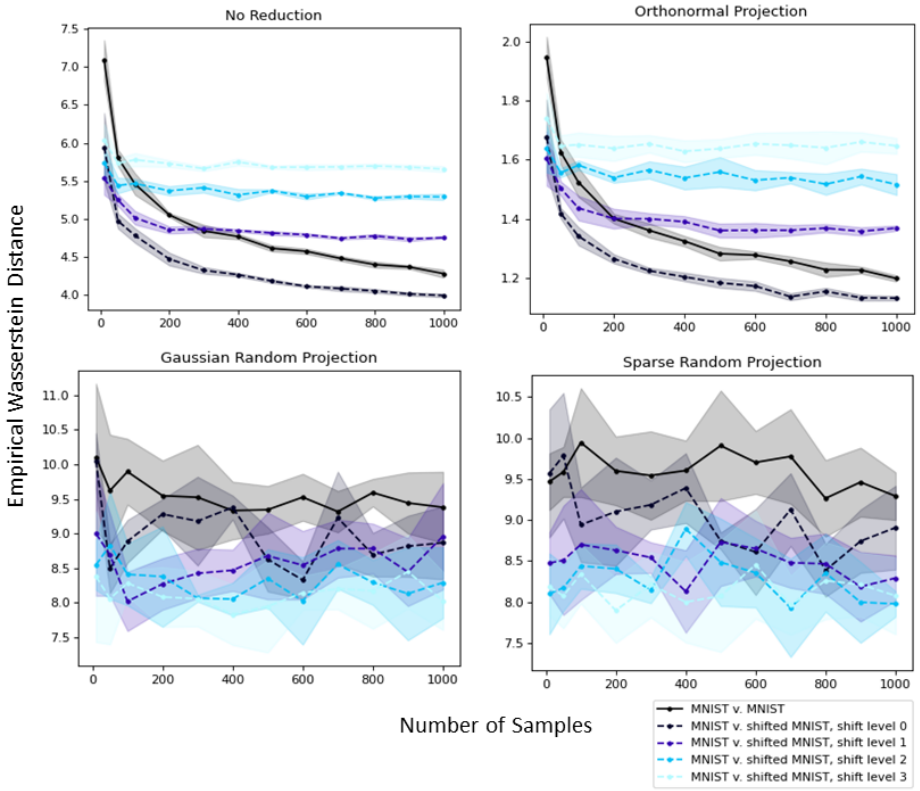

Figure 6: Empirical Wasserstein distance between MNIST and MNIST with varied levels of Gaussian blur, measured over a range of sample sizes. Curves are taken over 5 trials. MNIST samples are downsampled to 24×24, flattened, and projected to 50 dimensions. Orthonormal projection better preserves distributional information than Gaussian random projection and sparse random projection.

Appendix D. Related Work

Offline approaches for classification robustness. Most common among offline approaches are data augmentation techniques, which expand the experiences that a model would be exposed to during training. AugMix (Hendrycks et al., 2019) applies a set of transformations (e.g., rotation, translation, shear, etc.) to distort the original image. NoisyMix (Erichson et al., 2022) further injects noises into the augmented examples. ManifoldMix (Verma et al., 2019) also uses a data augmentation approach, but focuses on generating examples from a lower-dimensional latent manifold. Other notable work in the recent literature includes CutMix (Yun et al., 2019), PuzzleMix (Kim et al., 2020b), and DeepAugment (Hendrycks et al., 2020). Another offline approach is domain adaptation, in which models are trained on data from one distribution (i.e., source data) are adapted to perform well on unlabeled data in some other distribution (i.e., target data) (Ben-David et al., 2010). Among the many domain adaptation techniques, input-level translation is most similar to our work. These techniques attempt to move the source distribution closer to the target distribution by augmenting the source data in the input or feature space (Bousmalis et al., 2017; Hoffman et al., 2018). Finally, a related but distinct offline technique for classification robustness is adversarial training (Goodfellow et al., 2014), which aims to improve robustness to adversarial examples rather than distribution shift. In contrast to these offline approaches, our online technique allows for greater flexibility to distribution shifts unforeseen at train time. However, our method can be used in concert with offline approaches to further improve performance on corrupted samples.

Online approaches for classification robustness. A broad literature adjust for distribution shift at inference time. Test-time adaptation methods, for example, perform additional training at inference time to adapt models to incoming unlabeled data Gandelsman et al. (2022); Wang et al. (2022); Mummadi et al. (2021). A major challenge of such approaches is the generation of reliable pseudo-labels, especially for data under distribution shift.

Other online approaches, including ours, require no additional training of the model. Test time augmentation methods, for example, apply a set of transformations separately to the input, then ensemble the model's predictions on the different transformations of the data (Simonyan and Zisserman, 2014; Krizhevsky et al., 2012; He et al., 2016; Guo et al., 2017; Mummadi et al., 2021). In addition to the added complexity, a key disadvantage of this method is that the same transformations are applied to every input seen online. To overcome this, other techniques select the single best transformation among several candidates based on the model's performance on these candidates (Kim et al., 2020a; Lyzhov et al., 2020). Our method similarly selects transformations online and dynamically, but can do so without querying the model. Hence, when transformations are selected, they can be applied to any model to improve performance. Furthermore, our work uniquely identifies MDPs as an equivalent formulation of the transformation selection problem, allowing for the application of standard reinforcement learning techniques.

Distribution shift detection. Out-of-distribution shift detection has gained significant interest as more machine learning models are deployed into safety critical systems (Kaur et al., 2022; Hendrycks and Gimpel, 2018; Lee et al., 2018). In (Rabanser et al., 2019), various approaches to dimensionality reduction are explored and empirical results demonstrating the benefit to distribution shift detection are presented. Our work expands on these observations and utilizes a theoretically sound approach of using orthogonal projections for dimensionality reduction.

Deep learning for image restoration. Deep learning approaches for image restoration and recovery from corruption are well studied in the literature and exhaustive review is beyond the scope

of this paper. Most of these approaches focus on a specific corruption type or a set of corruptions. For example, Zhang et al. (2017) approaches Gaussian denoising by constructing denoising convolutional feed-forward networks. For a recent survey of deep blurring approaches and comprehensive approaches targeting multiple sets of noises we refer to Zhang et al. (2022); Ledig et al. (2017); Zamir et al. (2021, 2020, 2022). Such approaches utilize a generative model to restore information lost due to a low resolution. These methods are limited in recovery and can result in image artifacts which degrade classification performance. Here, we attempt to alleviate such limitations by using a composition of transformations which are selected automatically depending on the context. Additionally, the performance can be enhanced with additional image transformations in the action library (see Section 8).

Appendix E. Experimental Details

E.1. Surrogate Corruptions

Since in practice the corrupting transformations present at test time are unknown to the designer, we select surrogate corruptions for training the operability classifier and the policy network. To maximize generalization, we select six surrogate corruptions which subject the images to a broad set of transformations and influence each dimension of their state representations. Each serves to increase or decrease one of entropy, brightness, or standard deviation. Below, we describe each of the corruptions in greater detail.

Uniform Noise. We use additive uniform noise, $Uniform(a, b)$, as a representative of shifts that increase the entropy of the image. We select $b = -a \in \{0.14, 0.22, 0.32, 0.40, 0.90\}$ to generate five severity levels of uniform noise shift.

Median Blur. We use median blur as a representative of shifts that decrease the entropy of the image. We select a square kernel size $k \in \{2, 3, 4, 5, 6\}$ to generate five severity levels of median blur shift.

γ Correction (Type A). We adjust the γ value of the images, with $\gamma > 1$, to create a representative of shifts that increase the average brightness of the image. We select $\gamma \in \{1.4, 1.7, 2.0, 2.5, 3.0\}$ to generate five severity levels of this γ correction.

γ Correction (Type B). We adjust the γ value of the images, with $\gamma < 1$, to create a representative of shifts that decrease the average brightness of the image. We select $\gamma \in \{0.9, 0.8, 0.7, 0.6, 0.5\}$ to generate five severity levels of this γ correction.

Sigmoid Correction (Type A). We perform sigmoid correction to create a representative of shifts that increase the standard deviation of the image. We select cutoff 0.5 and gain $g \in \{7, 8, 9, 10, 11\}$ to generate five severity levels of this sigmoid correction.

Sigmoid Correction (Type B). We perform sigmoid correction to create a representative of shifts that decrease the standard deviation of the image. We select cutoff 0.5 and gain $g \in \{7, 6, 5, 4, 3\}$ to generate five severity levels of this sigmoid correction.

Figure 7 shows samples of images under these surrogate corruptions. We now describe the criteria used to select the parameters of each corruption listed above for five severity levels. Given a random sample $\mathbf{V}_c$ of the surrogate corrupted data and a random sample $\mathbf{V}$ of ImageNet data, we select the parameter for severity level 5 (strongest) such that

$$\left| \frac{\mathbb{F}_R(\mathbf{V}_c)_i - \mathbb{F}_R(\mathbf{V})_i}{\mathbb{F}_R(\mathbf{V})_i} \right| \in [0.3, 1.0], \tag{9}$$

where $\mathbb{F}_R(\mathbf{X})_i$ denotes the ith index of $\mathbb{F}_R(\mathbf{X})$. We use $i = 2$ for uniform noise and median blur, $i = 0$ for γ correction type A/B, and $i = 1$ for sigmoid correction type A/B. For the severity levels 1-4, we select parameters such that there is large variation in $\mathbb{F}_R(\mathbf{X})_i$ across the five severity levels.

Figure 7: Sample images from ImageNet subjected to surrogate corruptions.

E.2. Operability Classifier.

On the surrogate corruptions, we train a binary decision tree with depth of 8 selected by a hyperparameter sweep, and using Gini index as the split criterion. For label generation, we use a ResNet-50 classifier trained without data augmentation. The resulting classifier has an AUROC of 0.80 on a held out set of surrogate corrupted data. Of the six surrogate shifts, the operability classifier usually picks data under median blur data as inoperable and uniform noise data as operable. A full evaluation of the operability classifier is shown in Table 4.

Table 4: Percentage of images that the operability classifier labels as inoperable for surrogate and ImageNet-C shifts. Shifts gamma type A and gamma type B refer to γ correction with $\gamma > 1$ and $\gamma < 1$, respectively. Shifts sigmoid type A and sigmoid type B refer to sigmoid correction with parameters that increase and decrease the standard deviation of the image, respectively. In general, the classifier labels blur-type shifts as inoperable most often. Conversely, the classifier labels noise-type shifts as operable most often.

Surrogates	
shift	**% inoperable**
uniform noise	0.01
median blur	48.69
gamma type A	8.72
gamma type B	12.56
sigmoid type A	7.10
sigmoid type B	15.57

ImageNet-C	
shift	**% inoperable**
gaussian noise	0.004
shot noise	0.02
impulse noise	0.009
defocus blur	80.35
glass blur	70.22
motion blur	63.86
zoom blur	66.76
snow	0.18
frost	0.23
fog	30.18
brightness	6.99
contrast	1.77
elastic	23.09
pixelate	43.83
jpeg	41.63
speckle noise	0.03
gaussian blur	76.21
spatter	2.26
saturate	10.31

E.3. Policy Network.

Similar to the operability classifier, the policy network must be trained on surrogate corruptions selected at design time. However, when selecting surrogates for the policy network, we must additionally limit the size of the state space that must be explored to avoid convergence issues. Thus,

we select a subset of the six surrogate shifts previously described, consisting of two shifts with low inoperability rates, uniform noise and γ correction with $\gamma > 1$.

We manually select an action library $\mathbb{A}$ of eight correcting transforms, which aim to address both of these surrogate shifts. For each transform, we select weak parameters to guard against single actions with irreversibly destructive effect. Simultaneously, this allows `SuperStAR` to adapt accordingly to the noise severity. To target uniform noise, we include several denoising actions. These are a denoising convolutional neural network from the MATLAB Deep Learning Toolbox, a bilateral filter with filter size 2, wavelet denoising with BayesShrink thresholding (Chang et al., 2000), and wavelet denoising with VisuShrink thresholding (Donoho and Johnstone, 1994). To target γ correction, we include three Contrast Limited Adaptive Histogram Equalization (CLAHE) actions. Broadly, CLAHE applies histogram equalization over small tiles in the image, while limiting the allowable amount of contrast change. Our selected three CLAHE transformations, are CLAHE with (tile size, limit) of (2,1), (2,2), and (6,1). Finally, we include in $\mathbb{A}$ an action that does not enact any transformation, allowing for inaction in the presence of benign corruptions.

We train a deep neural network policy π with actions $\mathbb{A}$ and surrogate corruptions defined above. Sequences are limited to five actions, but a shorter sequence may occur based on the stopping criteria in Algorithm 1. We choose our stopping condition threshold to be $\alpha = 0.9$ and $\beta = 0.995$. For the estimation of Wasserstein distances, we convert images to grayscale and project them to 5000 dimensions using a randomly generated orthonormal matrix. We select the reward hyperparameters $\lambda = 20$ and $\omega = 0.994$. The neural network policy π is trained using the advantage actor critic algorithm. For both the actor and critic network, we employ a linear ReLU network with two hidden layers of 128 and 256 units. We train the actor and critic networks until convergence (about 1600 episodes) with learning rate 10^{-4}, learning frequency 1, discount factor 0.9, and exploration rate decaying exponentially from 0.9 to 0.1 according the rule $0.9^{0.07*(episode+1)}$.

The training curves are shown in Figure 8.

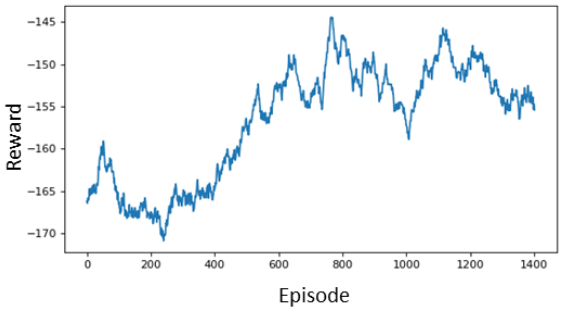

Figure 8: Reinforcement learning training curves, averaged with a moving window of 200 episodes. Values shown are the reward at episode termination.

Appendix F. Full Experimental Results for `SuperStAR` on ImageNet-C Shifts

Table 5: Average accuracies (%) on each ImageNet-C shift with and without `SuperStAR`for ResNet-50 classifiers. Accuracy improvement is denoted by Δ = R (recovered) − S (shifted). Values are over 5 severity levels with 3 trials each.

	No Data Aug			AugMix			NoisyMix			DeepAugment			DeepAug+AugMix			PuzzleMix		
shift	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ
none	74.52	74.52	0.00	75.94	75.94	0.00	76.22	76.22	0.00	75.86	75.86	0.00	75.26	75.26	0.00	75.63	75.63	0.00
gaussian noise	31.11	43.44	**12.33**	41.90	50.87	**8.98**	52.71	55.51	**2.80**	59.07	59.48	**0.41**	55.39	61.43	**6.05**	41.48	46.94	**5.46**
shot noise	28.61	42.81	**14.21**	41.78	50.93	**9.15**	51.81	55.38	**3.57**	58.21	58.46	**1.24**	55.76	62.37	**6.61**	37.39	45.56	**7.77**
impulse noise	26.57	39.36	**12.79**	38.78	47.49	**8.71**	50.73	53.37	**2.64**	58.61	58.38	-0.23	55.16	60.67	**5.50**	35.28	42.82	**7.54**
defocus blur	35.21	35.21	0.00	44.48	44.48	0.00	44.72	44.72	0.00	48.10	48.10	0.00	55.52	55.52	0.00	38.02	38.02	0.00
glass blur	25.55	25.55	0.00	32.97	32.97	0.00	35.71	35.71	0.00	38.39	38.39	0.00	44.45	44.45	0.00	25.71	25.71	0.00
motion blur	36.25	36.25	0.00	49.63	49.63	0.00	49.53	49.53	0.00	45.46	45.46	0.00	57.56	57.56	0.00	39.29	39.29	0.00
zoom blur	36.27	36.27	0.00	47.45	47.45	0.00	47.16	47.16	0.00	39.84	39.84	0.00	50.54	50.54	0.00	39.85	39.85	0.00
snow	30.51	29.28	-1.13	37.89	36.85	-1.04	43.20	41.52	-1.68	41.71	39.91	-1.80	47.68	46.36	-1.32	39.48	37.74	-1.75
frost	35.16	35.07	-0.09	41.39	41.24	-0.15	50.05	49.46	-0.59	46.87	46.08	-0.78	51.21	50.26	-0.95	46.96	45.75	-1.21
fog	42.79	42.79	0.00	44.97	44.97	0.00	51.34	51.34	0.00	49.88	49.88	0.00	54.46	54.46	0.00	55.62	55.62	0.00
brightness	65.17	65.72	**0.55**	67.35	68.46	**1.11**	68.82	69.87	**1.05**	69.04	69.73	**0.69**	69.42	70.18	**0.76**	69.59	69.67	**0.08**
contrast	35.56	37.69	**2.14**	48.96	49.85	**0.89**	50.37	52.74	**2.37**	44.89	48.23	**3.33**	56.01	57.40	**1.39**	50.56	52.87	**2.30**
elastic	43.24	43.24	0.00	50.18	50.18	0.00	51.02	51.02	0.00	50.35	50.35	0.00	53.26	53.26	0.00	43.31	43.31	0.00
pixelate	45.51	45.51	0.00	57.25	57.25	0.00	54.23	54.23	0.00	64.31	64.31	0.00	67.31	67.31	0.00	49.03	49.03	0.00
jpeg	52.47	52.47	0.00	58.49	58.49	0.00	61.85	61.85	0.00	56.99	56.99	0.00	61.31	61.31	0.00	56.83	56.83	0.00
speckle noise	36.09	49.26	**13.18**	50.61	56.94	**6.33**	57.67	60.88	**3.22**	62.21	63.81	**1.59**	60.93	65.66	**4.74**	42.24	51.92	**9.68**
gaussian blur	38.08	38.08	0.00	47.17	47.17	0.00	47.30	47.30	0.00	51.93	51.93	0.00	57.53	57.53	0.00	41.07	41.07	0.00
spatter	46.65	46.44	-0.20	53.25	52.93	-0.32	57.63	57.34	-0.29	53.74	53.58	-0.16	57.75	57.61	-0.14	53.27	52.95	-0.32
saturate	59.00	59.17	**0.17**	61.42	61.89	**0.47**	63.48	64.00	**0.52**	64.59	64.81	**0.22**	65.79	66.12	**0.33**	65.96	65.60	-0.37

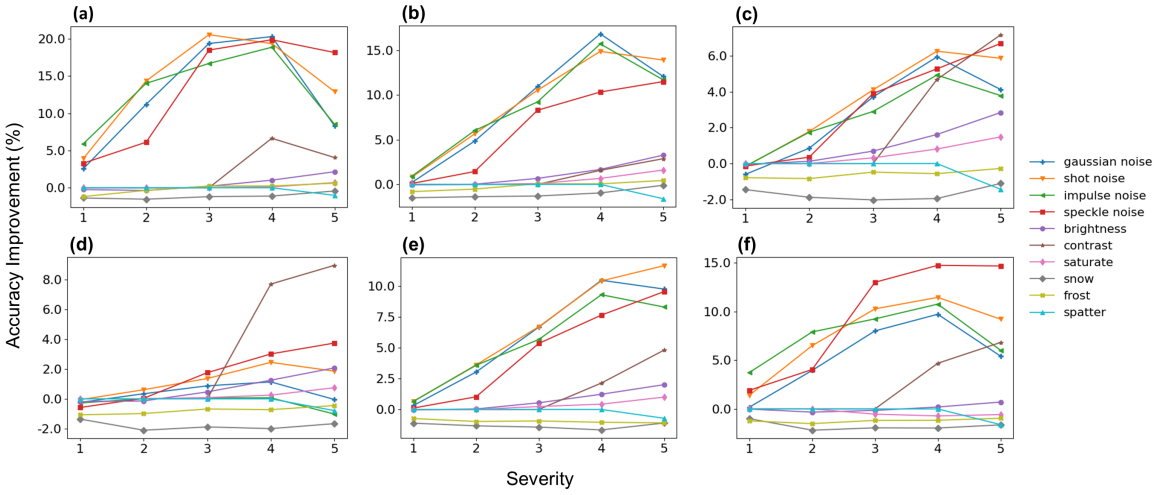

Figure 9: Accuracy improvements for increasing severity evaluated on classifiers trained with a) no data augmentation, b) AugMix, c) NoisyMix, d) DeepAugment, e) DeepAugment and AugMix, and f) PuzzleMix. In general, performance improvements are more pronounced for greater severities of corruption.

Appendix G. Inaction for Select Shifts

As noted in the main text, `SuperStAR` refrains from taking action when no shift is present and for a variety of ImageNet-C shifts. This inaction occurs due to a combination of both our operability classifier and our stopping conditions in Algorithm 1. For example, blur-type shifts are frequently labeled as inoperable: over 80% of defocus blur images are inoperable, and similarly for other blur-type shifts (see Appendix E). We also observe that inaction can at times be overly conservative. An example case is fog shift, where an appropriate action exists (e.g., CLAHE), yet no action is taken. We now explore this inaction on fog shift further.

Although `SuperStAR` chooses to take no action on fog shift, there exists an appropriate action for this shift. In fact, Contrast Limited Adaptive Histogram Equalization (CLAHE) can achieve a 4.15% accuracy improvement, averaged across all five severity levels, on the AugMix classifier. Further, if allowed to take the full 5-step sequence of actions, `SuperStAR` would initially select a CLAHE action to correct for fog shift. This is supported by Figure 10, which shows that samples from fog shift have a state representation similar to those from our surrogate γ correction (with $\gamma > 1$) shift and other ImageNet-C shifts for which CLAHE is a strong correction (i.e., brightness, contrast, and saturate), leading `SuperStAR` to select a similar action for fog shift. However, `SuperStAR` uses stopping conditions that can prematurely terminate its procedure. The nature of fog corruption is such that, after taking the first action, `SuperStAR` decides to terminate the process. We speculate that this can be improved by using larger batch sizes for the validation set, which would allow for a more accurate estimate of the Wasserstein distance. Simultaneously, this would come at a higher computational cost for training the RL policy.

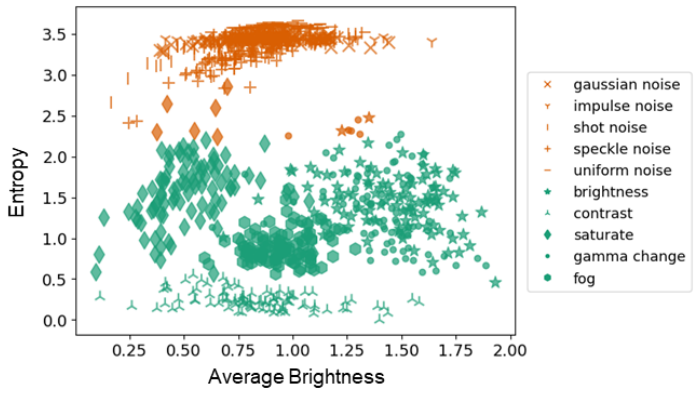

Figure 10: K-means clustering of sampled state representations for brightness, contrast, and saturate shifts from ImageNet-C, fog shift from ImageNet-C, and our surrogates. 100 samples were taken from each shift. $K = 2$ clusters were chosen by selecting K from the range $[1, 10]$ via the elbow method. The projection onto the entropy and average brightness dimensions of the state representation is shown. Samples from fog shift are clustered with those from the γ correction, brightness, contrast, and saturate shifts.

Appendix H. Full Experimental Results for `SuperStAR` on Composite ImageNet-C Shifts

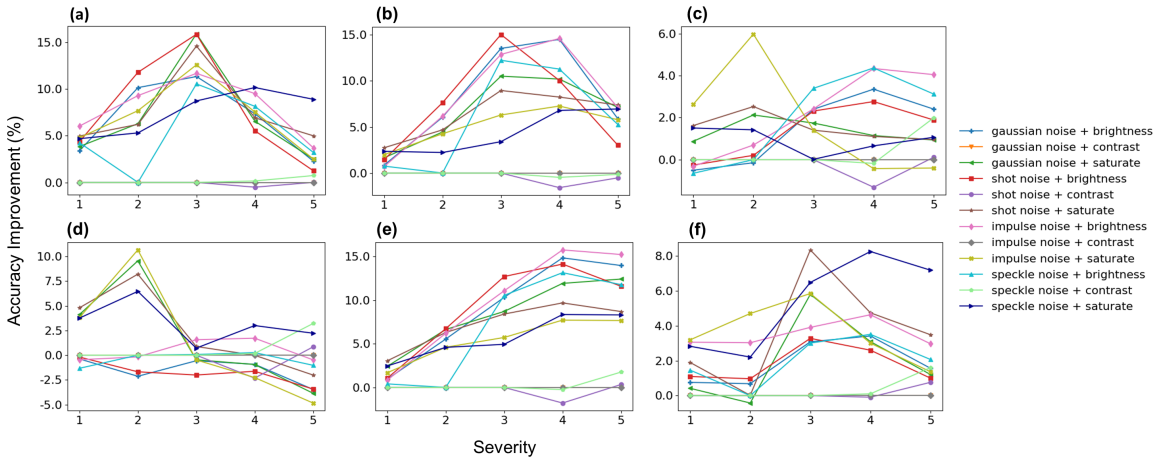

Figure 11: Accuracy improvements on composite ImageNet-C shifts for increasing severity evaluated on ImageNet classifiers trained with a) no data augmentation, b) AugMix, c) NoisyMix, d) DeepAugment, e) DeepAugment and AugMix, and f) PuzzleMix.

Appendix I. Experimental Details for CIFAR-100-C

For the CIFAR-100-C benchmark, we retrain only the operability classifier. We train on the same six surrogates, regenerated for CIFAR-100. For label generation, we use a Wide ResNet with depth 28 and widening factor 10 (WRN-28-10) trained on CIFAR-100 without data augmentation (Zagoruyko and Komodakis, 2016). The resulting operability classifier has an AUROC of 0.73 on a held out set of surrogate data. Predictions made by the operability classifier on surrogate and CIFAR-100-C shifts are shown in Table 6. Lastly, we update our stopping condition parameters to $\alpha = 0.9$, $\beta = 0.972$.

For evaluation, we reuse the WRN-28-10. We also evaluate on WRNs trained with AugMix (WRN-40-2), NoisyMix (WRN-28-4), and PuzzleMix (WRN-28-10).

Table 6: Percentage of images that the operability classifier labels as inoperable for surrogate and CIFAR-100-C shifts. Shifts gamma type A and gamma type B refer to γ correction with $\gamma > 1$ and $\gamma < 1$, respectively. Shifts sigmoid type A and sigmoid type B refer to sigmoid correction with parameters that increase and decrease the standard deviation of the image, respectively. In general, the classifier labels blur-type shifts as (with the exception of glass blur) inoperable most often. Conversely, the classifier labels noise-type shifts as operable most often.

Surrogates	
shift	**% inoperable**
uniform noise	0.002
median blur	43.88
gamma type A	6.72
gamma type B	11.30
sigmoid type A	7.05
sigmoid type B	15.09

CIFAR-100-C	
shift	**% inoperable**
gaussian noise	0.004
shot noise	0.03
impulse noise	1.65
defocus blur	44.34
glass blur	2.59
motion blur	37.62
zoom blur	45.73
snow	3.42
frost	1.90
fog	25.83
brightness	9.41
contrast	20.97
elastic	24.73
pixelate	26.80
jpeg	15.39
speckle noise	0.04
gaussian blur	51.42
spatter	4.70
saturate	10.63

Appendix J. Full Experimental Results for `SuperStAR` on CIFAR-100-C Shifts

Table 7: Average accuracies (%) on each CIFAR-100-C shift with and without `SuperStAR` for Wide ResNet classifiers. Accuracy improvement is denoted by Δ = R (recovered) − S (shifted). Values are over 5 severity levels with 3 trials each.

	No Data Aug			AugMix			NoisyMix			PuzzleMix		
shift	S	R	Δ	S	R	Δ	S	R	Δ	S	R	Δ
none	81.13	81.13	0.00	76.28	76.28	0.00	81.29	81.29	0.00	84.01	84.01	0.00
gaussian noise	21.12	26.81	**5.70**	47.89	51.07	**3.18**	65.91	66.34	**0.43**	20.87	28.18	**7.31**
shot noise	29.96	36.34	**6.38**	55.69	58.24	**2.55**	70.39	70.72	**0.33**	31.12	39.37	**8.25**
impulse noise	19.21	26.09	**6.88**	59.68	59.09	-0.59	79.72	76.08	-3.64	37.18	37.01	-0.17
defocus blur	64.44	64.44	0.00	73.42	73.42	0.00	78.23	78.23	0.00	69.92	69.92	0.00
glass blur	20.68	26.61	**5.93**	54.08	56.09	**2.00**	58.82	60.48	**1.66**	31.07	37.62	**6.55**
motion blur	60.25	60.25	0.00	70.46	70.46	0.00	74.73	74.73	0.00	66.14	66.14	0.00
zoom blur	59.58	59.58	0.00	71.83	71.83	0.00	76.71	76.71	0.00	65.10	65.10	0.00
snow	59.07	59.07	0.00	65.83	65.83	0.00	71.83	71.83	0.00	70.99	70.99	0.00
frost	54.35	54.35	0.00	63.69	63.69	0.00	71.00	71.00	0.00	65.21	65.21	0.00
fog	71.25	71.25	0.00	66.59	66.59	0.00	72.84	72.84	0.00	77.21	77.21	0.00
brightness	76.94	76.94	0.00	73.50	73.50	0.00	78.32	78.32	0.00	80.13	80.13	0.00
contrast	62.97	62.97	0.00	65.25	65.25	0.00	68.03	68.03	0.00	72.88	72.88	0.00
elastic	64.23	64.23	0.00	68.09	68.09	0.00	73.43	73.43	0.00	68.58	68.58	0.00
pixelate	54.28	54.28	0.00	63.54	63.54	0.00	70.46	70.46	0.00	52.34	52.34	0.00
jpeg	50.40	50.40	0.00	62.11	62.11	0.00	69.24	69.24	0.00	51.71	51.71	0.00
speckle noise	31.58	35.76	**4.18**	58.11	59.33	**1.23**	71.67	71.57	-0.09	33.96	38.94	**4.98**
gaussian blur	54.11	54.11	0.00	71.48	71.48	0.00	76.74	76.74	0.00	61.11	61.11	0.00
spatter	61.23	62.23	**1.00**	72.28	71.25	-1.02	78.11	77.44	-0.67	79.73	78.90	-0.83
saturate	68.82	68.88	**0.06**	64.52	64.77	**0.25**	69.83	70.21	**0.39**	72.93	72.99	**0.05**

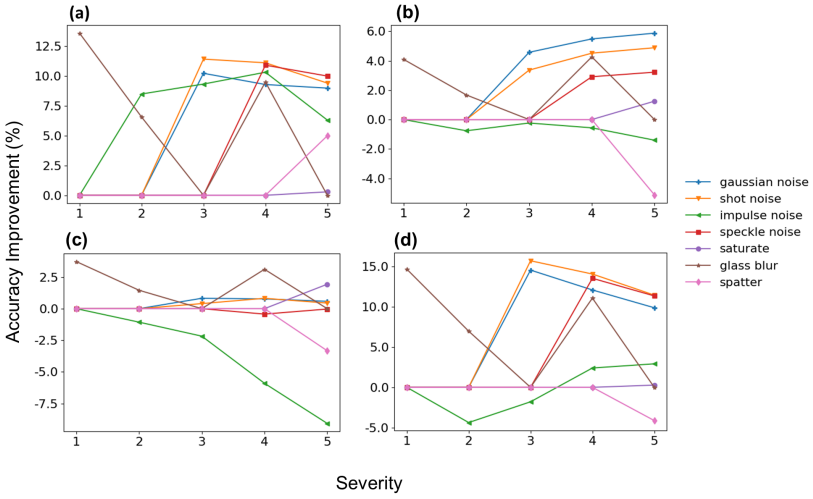

Figure 12: Accuracy improvements on CIFAR-100-C for increasing severity evaluated on CIFAR-100 classifiers trained with a) no data augmentation, b) AugMix, c) NoisyMix, d) PuzzleMix.

Appendix K. Effect of Surrogate Shifts on `SuperStAR` Performance

Here we examine the how our choice of surrogate image corruptions impacted `SuperStAR` performance. Figure 13 shows the projected state representations for 100 samples of our best performing shifts and each surrogate shift, sorted into $K = 2$ clusters by K-means clustering. The value of K was selected from the range $[1, 9]$ via the elbow method. We see that samples from ImageNet-C noise-type shifts are in the same cluster as those from our uniform noise shift. Likewise, brightness, contrast, and saturate are clustered with our γ correction shift. Thus, `SuperStAR` generalizes well from the surrogates to these ImageNet-C shifts.

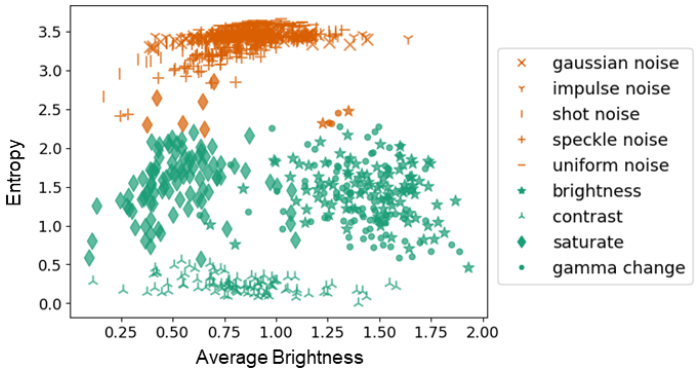

Figure 13: K-means clustering of sampled state representations for best-performing shifts from ImageNet-C and our surrogates (projected onto two dimensions). The surrogate shifts are clustered with the appropriate ImageNet-C shifts.

Appendix L. Limitations of SuperStAR

In this supplementary section, we explore the limitations of SuperStAR in greater detail.

Action Library. One important limitation of SuperStAR is the technique's reliance on an appropriate selection of the action library. Since the policy network selects corrective actions from this library, by design our method can only recover from distribution shifts targeted by the action library. For example, for the ImageNet-C benchmark, our action library caters to noise-related and γ-level-related distribution shifts. The result is that SuperStAR overlwhelmingly performs better on these types of ImageNet-C shifts, in contrast to blur-related, weather-related, and corruption-related shifts. Furthermore, it is not feasible to simply select a large, comprehensive action library, as this prohibitively increases the search space that must be traversed by the reinforcement learning algorithm.

Surrogate Corruptions. Much like the action library, the selection of surrogate corruptions can pose a challenge in implementing SuperStAR. Our method hinges on the reinforcement learning agent interpolating knowledge about the surrogate corruptions to apply to the test-time corruptions (e.g., ImageNet-C or CIFAR-100-C). If there is large deviation between the set of surrogate corruptions and the set of test-time corruptions, the successful application of knowledge to the test data is less likely. However, this is a natural consequence of any data-driven solution, and the flexibility lent by using an offline approach mitigates this challenge.

Responsiveness. To estimate the Wasserstein distance in our reinforcement learning reward function, our method requires a set of data samples already subject to distribution shift. In high dimensions, the sample complexity of this estimate grows (Ramdas et al., 2017). Even with dimensionality reduction techniques, a large sample size may be required, as the reduction in dimensions cannot be so large as to lose essential information in the data. In these cases, the responsiveness of SuperStAR may be limited in speed, as SuperStAR must wait for a large number of samples to begin adapting and recovering from distribution shift. We note that a quick response time is not an essential requirement in our setting, as we assume that when distribution shift arises, it persists for a certain duration of time. This is not a stringent assumption for the naturally occurring distribution shifts we consider in this work.

Theoretical Guarantees. Since our method views the end model we wish to adapt as a black box, we can make no theoretical guarantees on how SuperStAR affects the class-wise performance of the model. However, we provide theoretical guarantees (see Theorem 2) that selecting transformations that minimize the distance between the training distribution and the corrupted distribution in turn minimizes the overall loss in performance on the corrupted data after correction.