Dynamic Test-Time Augmentation via Differentiable Functions

SHOHEI ENOMOTO¹, MONIKKA ROSLIANNA BUSTO¹, and TAKEHARU EDA¹, Member, IEEE
¹NTT, Musashino, Tokyo 180-8585, Japan
Corresponding author: Shohei Enomoto (e-mail: alesana882@gmail.com).

ABSTRACT Distribution shifts, which often occur in the real world, degrade the accuracy of deep learning systems, and thus improving robustness to distribution shifts is essential for practical applications. To improve robustness, we study an image enhancement method that generates recognition-friendly images without retraining the recognition model. We propose a novel image enhancement method, DynTTA, which is based on differentiable data augmentation techniques and generates a blended image from many augmented images to improve the recognition accuracy under distribution shifts. In addition to standard data augmentations, DynTTA also incorporates deep neural network-based image transformation, further improving the robustness. Because DynTTA is composed of differentiable functions, it can be directly trained with the classification loss of the recognition model. In experiments with widely used image recognition datasets using various classification models, DynTTA improves the robustness with almost no reduction in classification accuracy for clean images, thus outperforming the existing methods. Furthermore, the results show that robustness is significantly improved by estimating the training-time augmentations for distribution-shifted datasets using DynTTA and retraining the recognition model with the estimated augmentations. DynTTA is a promising approach for applications that require both clean accuracy and robustness. Our code is available at https://github.com/s-enmt/DynTTA.

INDEX TERMS Distribution-shift, Image enhancement, Robustness, Test-time augmentation, Visual recognition.

I. INTRODUCTION

With the development of deep learning, the field of visual recognition has made great progress, and services using deep learning are becoming more practical. Many deep learning models are trained and validated on images from the same distribution. However, images in the real world are subject to distribution shifts due to various factors such as weather conditions, sensor noise, blurring, and compression artifacts. Deep learning models do not usually take these distribution shifts into account, causing the accuracy to reduce in the presence of such artifacts. This is fatal for safety-critical applications, such as automated driving, where the environment changes frequently.

There are two main approaches to solving this problem: training robust recognition models and image enhancement. The former uses data augmentation [13], [15], [19], or training algorithms such that the recognition model has high robustness. The latter uses test-time augmentation [20] or deep neural networks (DNNs) [47] to transform the distorted images into recognition-friendly images that are easily recognized by the recognition model. These approaches can be used in combination to achieve higher robustness. This paper focuses on the image enhancement approach, which is highly practical because it is used before inference by the pre-trained recognition model without retraining the recognition model. We found that DNN-based image enhancement [47] overfits to particular transformation patterns, resulting in excessive image transformation even for clean images and reduced clean accuracy. Test-time augmentation-based image enhancement [20], which dynamically selects the best of several data augmentations for an input image, improves robustness while maintaining clean accuracy. However, its augmentation search space is limited, so the improvement in robustness is small.

To remove these limitations, we propose a novel image enhancement method, DynTTA. DynTTA uses differentiable data augmentation techniques [9], [28], [40] and image blending [13], [15], [19], [56] to dynamically generate the recognition-friendly image for an input image from a huge augmentation space. DynTTA incorporates a DNN-based im-
Dynamic Test-Time Augmentation via Differentiable Functions

A. ROBUSTNESS OF DNNs

DNNs are known to be vulnerable to image distortions, which often occur in the real world. These image distortions have been studied in several works. Diamond et al. [3] showed that defects such as noise and blur in real-world sensors degrade performance in image recognition networks. Pei et al. [36] found that haze reduces the accuracy of image classification and experimented that the dehaze method for human visibility does not effectively improve image classification performance. Pei et al. [37] empirically studied real-world image degradation problems for nine kinds of degraded images: — hazy, motion-blurred, fish-eye, underwater, low resolution, salt-and-peppered, white Gaussian noise, Gaussian-blurred, and out-of-focus. Various datasets have been proposed to evaluate the robustness of DNNs. Geirhos et al. [7] made a style-transformed dataset to demonstrate that DNNs recognize objects by their texture rather than their shape. Hendrycks and Dietterich [12] proposed 19 algorithmically generated corruption datasets in five levels, belonging to four categories: noise, blur, weather, and digital. They showed that DNNs trained on a clean dataset have lower accuracy on these corruption datasets. In addition, they introduced a dataset of naturally occurring adversarial examples in the real world [14] and a dataset containing distribution shifts that occur in the real world, such as image style, image blurriness, geographic location, camera operation, and more [11]. In this study, we evaluate our method on these datasets.

B. TRAINING ROBUST CLASSIFICATION MODELS

Many works have been studied to improve the robustness of DNNs to naturally occurring image distortions. One way to improve the robustness is the data augmentation. Mixing multiple data augmentations [5], [13], [15], [19], [54], [56] and adding noise during training [42], [53] have been proposed as effective data augmentations to improve robustness. Mintun et al. [33] found that the similarity between data augmentation and test-time image corruption is strongly correlated with performance. Training algorithms to improve robustness have also been studied. Disentangled learning via auxiliary batch normalization [4], [32], [58], [61], a method to improve adversarial training, improves not only robustness but also clean accuracy. Li et al. [27] proposed an algorithm to train the models paying complementary attention to the shape and texture of the objects. Since these methods are used when training recognition models, they can be used in combination with image enhancement methods, which assume that the recognition models are pretrained. Our experimental results show that combining the recognition models pretrained by these methods with image enhancement further improves the robustness.

C. IMAGE ENHANCEMENT

This section presents two image enhancement approaches, test-time augmentation and image transformation. Table 1 shows a summary of the pros and cons of these approaches and DynTTA.

II. RELATED WORK

The main contributions of this paper are as follows.

• We propose DynTTA, a novel image enhancement method based on differentiable data augmentation techniques and image blending. DynTTA generates diverse images by transformations with various data augmentations, including a DNN, which significantly improves robustness while maintaining clean accuracy.

• We propose a novel method to estimate effective training-time augmentations and show that retraining the classification model with the estimated augmentations significantly improves accuracy.

• Our extensive experiments show that DynTTA improves robustness compared to existing methods while maintaining clean accuracy. In addition, experimental results in the blind setting show that DynTTA is practical under various distribution shifts.
1) Test-time Augmentation

Recognition accuracy can be improved by using data augmentation during not only the training time but also the test time. Various test-time augmentation methods have been proposed, using simple geometric transformations such as flip and crop [2], [10], [23], [38], [48], [59], mixup [24], [35], and augmentation in the embedding space [1]. Shanmugam et al. [44] experimentally analyzed why and when test-time augmentation works, and Kimura et al. [21] gave theoretical guarantees. These studies use only specific data augmentations for all test set images. Dynamically selecting the best data augmentation for each input image is expected to improve the accuracy. Lyzhov et al. [31] proposed a greedy search-based test-time augmentation method to find the augmentation policy on the test set, but it is not optimal for each input image. Kim et al. [20] proposed a module called Loss Predictor to predict the classification loss of augmented images, allowing dynamic selection of the best one for the input image from 12 data augmentations. However, despite the augmentation space actually being infinite, these methods have significant limitations on the augmentation space due to computational cost and the improvement in robustness is small. Our method eliminates this limitation and achieves better robustness than existing methods by generating the best augmented image from a large augmentation space. In addition, we propose the training-time augmentation estimation method for distribution-shifted datasets using test-time augmentation and show that retraining the classification model by estimated augmentations significantly improves accuracy.

2) Image Transformation

Several methods have been proposed to improve the recognition accuracy by image transformation using DNNs. Sharma et al. [45] proposed convolutional neural network-based enhancement filters that enhance image-specific details to improve recognition accuracy. Subsequent work [6], [25], [29], [34], [47] used DNN to transform corrupted images into recognition-friendly ones. Milanfar et al. [49] showed that not only image enhancement but also resizing at the same time improves the recognition accuracy. These methods use DNNs to transform images, but tend to perform excessive transformations to remove distortions, which reduces the accuracy of clean images. By considering these image enhancement methods as a kind of data augmentation, we treat them as part of DynTTA. DNN-based transformations reduce clean accuracy because they overfit particular transformation patterns that removes distortions. We integrate DNN transformation and data augmentations to enable learning of diverse image transformations, which avoids overfitting and improves robustness while maintaining clean accuracy.

### III. DynTTA

#### A. KEY IDEAS

Many data augmentations have continuous magnitude parameters (e.g., rotation has a continuous magnitude parameter from 0 to 360 degrees). In addition, there are many combinations of two or more data augmentations (e.g., augmentation is performed in order of rotation, contrast, and sharpness). These facts make the augmentation space infinite. To solve the test-time augmentation challenge of selecting the best combination of data augmentations and their optimal magnitude parameters for the test image from an infinite space, we introduce two key ideas. The first is to find the local optimal magnitude parameter by optimization. Since most data augmentations are differentiable with respect to the magnitude parameter [9], [28], [40], [41], the local optimal magnitude parameter is obtained by optimization algorithms, such as optimization algorithms, such as
gradient descent. The second is to represent the combination of augmentation with image blending, which is the process of affine combination of multiple images. For example, if the weights are [1, 0, 0], only the 1st augmented image is selected, and if the weights are [0.5, 0, 0.5], the 1st and 3rd augmented images are combined. Note that image blending does not provide the performing order of the augmentation. We call these ideas “Optimization” and “Blending”, respectively. On the basis of these key ideas, we propose a novel image enhancement method, DynTTA.

B. OVERVIEW OF DYNTTA
The overview of DynTTA is shown in Fig. 1. The differentiable data augmentations used in DynTTA and their range of magnitude parameters $M$ are defined. Most data augmentations, such as rotation, have a magnitude parameter. For data augmentations that do not have a magnitude parameter, such as auto-contrast, $M$ is not defined. Section III-C details the data augmentations of DynTTA.

Any DNN can be used as the backbone of DynTTA by modifying its output layer. DynTTA takes an image as input and outputs the magnitude parameters $m_1, \cdots, m_K$ and blend weights $w_1, \cdots, w_K$. $K$ is the number of data augmentations. The magnitude parameter is mapped to a value in the range $(0, 1)$ or $(-1, 1)$ using an activation function $f_k^{\text{act}}$ such as sigmoid function or hyperbolic tangent function. It is then multiplied by the predefined magnitude range $M$. The blend weights are converted to weights that sum to 1 by the softmax function. $m$ and $w$ are shown in the following equation.

$$\hat{m}_k = \begin{cases} M_k f_k^{\text{act}}(m_k) & \text{(if $M_k$ and $f_k^{\text{act}}$ are defined)} \\ \emptyset & \text{(else)} \end{cases}$$

$$\hat{w}_k = \frac{\exp(w_k)}{\sum_{i=1}^{K} \exp(w_i)}.$$  

By using the magnitude parameters and blend weights, DynTTA generates a recognition-friendly image. $K$ data augmentations are performed with magnitude parameters $\hat{m}$, then DynTTA generates a blended image by linearly combining augmented images with $\hat{w}$. DynTTA outputs the image using the following equation:

$$\hat{x} = \sum_{k=1}^{K} \hat{w}_k O_k(x, \hat{m}_k),$$

where $x$ is the input image, $O_k$ is the $k$-th augmentation operation, and $\hat{x}$ is the output image.

C. AUGMENTATION SPACE
The data augmentations and their $f_k^{\text{act}}$ and $M$ used by DynTTA are shown in Table 2. Because DynTTA obtains magnitude parameters through optimization, only differentiable data augmentations are used. This study uses the kornia library [40] and deals only with general data augmentation. We do not employ data augmentation such as shear because we believe it is ineffective when used during testing. Flip is a typical test-time augmentation, but it is not used because it is non-differentiable. Placing low-pass and high-pass filters within the DNN architecture [16], [39], [55], [62] or using them as training-time augmentation [17], [46], [60] is known to improve robustness. Inspired by these studies, we try to improve the robustness by using low-pass and high-pass filters as test-time augmentation. These filters have a filter size parameter that is not obtained via gradient descent. Therefore, this parameter is discretized, and multiple filters are prepared. 19 low-pass and 19 high-pass filters are made, each with a filter size from 0.05 to 0.95 in increments of 0.05. URIE uses a DNN to generate recognition-friendly images. Such an image transformation model is applied to DynTTA by considering it as a kind of augmentation. In this case, the image transformation model is trained simultaneously with DynTTA.

DynTTA handles many data augmentations but is also computationally expensive. By not executing data augmentation with small blend weights, the number of data augmentation executions is significantly reduced while maintaining accuracy. The details are described in the Appendix.

D. TRAINING AND TESTING
DynTTA is trained by applying it before inference by a pre-trained classification model. When DynTTA is trained, the classification model is frozen. The output image of DynTTA is used as input to the classification model to calculate the cross entropy loss. Because DynTTA is differentiable, it minimizes this loss and outputs recognition-friendly images. During testing, the DynTTA is frozen and the DynTTA output image is input into the classification model to obtain the prediction results.

E. RETRAINING WITH ESTIMATED TRAINING-TIME AUGMENTATIONS
We propose a different use of DynTTA, improving accuracy by reproducing unknown distribution shifts with data augmentations and retraining the classification model with it. For a distribution-shifted dataset, DynTTA outputs the blend weights that weight effective test-time augmentations. We hypothesize that the distribution shift would consist of inverse operations of the highly weighted test-time augmentations. For example, consider an unknown distribution shift dataset for which DynTTA estimates that LPF is effective. Since LPF removes high-frequency components, this dataset is expected to contain high-frequency components. To make a classification model robust against high-frequency components, high-frequency components should be included in the training distribution. To achieve this, the inverse operation of LPF, i.e., HPF, is used to include the high-frequency components in the training dataset and retrain the model.

Algorithm 1 shows how to estimate effective augmentations at training-time. We rank the effectiveness of augmentation by the blend weights and obtain the top $k$ inverse operations of estimated augmentations. If the inverse operation does not exist, it is omitted. The inverse operation for each augmentation is defined in Table 2. The estimated augmenta-
TABLE 2. Data augmentations and magnitude parameter ranges. A-Contrast, LPFs, and HPFs denote auto-contrast, low-pass filters, and high-pass filters, respectively.

| Augmentation | \( f^{\text{act}} \) | M | Inverse Augmentation | \( f^{\text{act}} \) | M | Inverse |
|--------------|----------------|---|----------------------|----------------|---|---------|
| Rotate       | tanh           | 30 | Rotate               |                 |   |         |
| Scale        | tanh           | 0.3 | Scale                |                 |   |         |
| Saturate     | sigmoid        | 5.0 | Saturate             |                 |   |         |
| Contrast     | sigmoid        | 3.0 | Contrast             |                 |   |         |
| Sharpness    | sigmoid        | 10 | Sharpness            |                 |   |         |
| Brightness   | tanh           | 0.6 | Brightness           |                 |   |         |
| A-Contrast   |                 |   | A-Contrast           |                 |   |         |
| Hue          | tanh           | 2.0 | Hue                  |                 |   |         |
| Equalize     |                 |   | Equalize             |                 |   |         |
| Invert       |                 |   | Invert               |                 |   |         |
| Gamma        | sigmoid        | 3.0 | Gamma                |                 |   |         |
| LPFs         |                 |   | LPFs                 |                 |   |         |
| HPFs         |                 |   | HPFs                 |                 |   |         |
| URIE         |                 |   | URIE                 |                 |   |         |

Algorithm 1: Estimation of effective training-time augmentations

**Data:** A set of samples from an unknown distribution;
**Result:** Effective training-time augmentations for a given distribution;
Initialize an empty list \( L \);
for each estimating step do
- Sample a mini-batch from the dataset of unknown distribution;
- Output the blend weights and magnitude parameters by DynTTA;
- Append the blend weights to \( L \);
end
Average the blend weights for each augmentation in \( L \);
Sort and rank the blend weights of the augmentations in descending order;
return Inverse operations of the top \( k \) augmentations

Our method was evaluated on two image recognition datasets (CUB [57] and ImageNet [43]) and one domain generalization dataset (PACS [26]). All experiments except ImageNet were run three times, and the average results are reported. Loss Predictor (LP) \((k = 1)\) [20] and URIE [47] were used as comparison methods. Classification models were prepared in advance, and the backbone network of DynTTA and LP was prepared by fine-tuning the ResNet18 [10] pretrained on ImageNet. For training in the non-blind setting, corruption was randomly selected for each mini-batch to train the enhancement models. The training corruptions consist of 15 \( Seen \) corruptions in five levels and clean (no corruption). For testing, four \( Unseen \) corruptions were used. In the blind setting, the enhancement models were trained using AugMix+DeepAugment [11] for ImageNet and AugMix for the other datasets. For testing, all 19 corruptions were used. When evaluating on the corruptions dataset, the average accuracy was used as an evaluation metric. In addition to evaluating the robustness of these distribution shift datasets, the clean accuracy on the standard dataset was also evaluated. The baseline did not use the enhancement model, and the difference from the baseline is reported in the results. In the tables showing experimental results, the best result is shown in **bold**, and the second best result is shown in *underline*.
C. PERFORMANCE EVALUATION

1) Classification on the CUB

The results for classification accuracy on the CUB dataset are shown in Table 3. In the non-blind setting, URIE improves robustness but reduces the clean accuracy by a maximum of about 3.3 percentage points. LP does not decrease the clean accuracy, but the improvement in robustness is small. DynTTA outperforms the comparison methods in terms of both clean accuracy and robustness and significantly improves the robustness with almost no decrease in clean accuracy. In the blind setting, similar to the non-blind setting, URIE tends to reduce the clean accuracy by a maximum of about 1.0 percentage points. LP improves both clean accuracy and robustness slightly. DynTTA has less degradation in clean accuracy than URIE, and it is also more robust. In particular, DynTTA significantly improves the robustness of Mixer-B16 [51] compared to the comparison methods. Our experimental results show that AugMix-trained enhancement models further improve the robustness of AugMix-trained classification models. The results indicate that our method can be used in conjunction with existing robustness improvement methods. Furthermore, experimental results on generalizability for different classification models at training and testing, the effects of the backbone network, augmentation space, and magnitude range, and comparison to simple baselines are provided in the Appendix.

2) Classification on the ImageNet

The results for classification accuracy on the ImageNet dataset are shown in Table 4. In the non-blind setting, URIE improves robustness but reduces the clean accuracy by a maximum of about 1.6 percentage points. LP does not decrease the ResNet50 clean accuracy, but the improvement in robustness is small. In DeiT [52], LP shows a decrease in both clean accuracy and robustness. DynTTA slightly reduces the clean accuracy, but it improves robustness more than the comparison methods. In the blind setting, URIE and LP experiments show almost the same trend in results as in the non-blind setting. DynTTA improves robustness over URIE without degrading the clean accuracy.

Furthermore, the DynTTA performance was evaluated on distribution shift datasets Stylized-ImageNet [7] (Stylized), ImageNet-A [14] (A), and ImageNet-R [11] (R) other than the corruption dataset in the blind setting. The results are shown in Table 5. URIE significantly decreases accuracy when the dataset is A and the classification model is DeiT, indicating that URIE is overfitting to the Corruption and R distribution. LP barely affects the accuracy with ResNet50 and worsens the accuracy with DeiT. While URIE and LP sometimes show decreased accuracy, DynTTA consistently improves generalization performance on these complex distribution shift environments without overfitting a particular distribution.

3) Classification on the PACS

The effect of changing the training domain in the blind setting was evaluated, and the average accuracy is shown in Table 6. Other experimental settings followed DomainBed [8]. Image enhancement models trained in the blind setting (trained with AugMix) are also effective for the domain generalization dataset, and DynTTA outperforms URIE in terms of average accuracy.

4) Retraining with Estimated Augmentations

The effect of retraining with estimated augmentations by DynTTA on the PACS dataset was evaluated. A small split
TABLE 4. Classification accuracy on the ImageNet dataset. The top table shows the results for the non-blind setting, and the bottom table shows the results for the blind setting. The numbers in parentheses indicate the differences from the baseline.

| Non-blind Classifier | Enhancer | Clean | Unseen |
|----------------------|----------|-------|--------|
| ResNet50             | URIE     | 74.56 (1.57) | 49.05 (3.96) |
|                      | LP       | 76.12 (-0.01) | 46.03 (0.94) |
|                      | DynTTA   | 75.89 (-0.24) | 50.55 (5.46) |
| DeiT-base            | URIE     | 81.01 (-0.73) | 67.50 (2.19) |
|                      | LP       | 81.09 (-0.65) | 64.22 (-1.09) |
|                      | DynTTA   | 81.25 (-0.50) | 67.92 (2.61) |

| Blind Classifier | Enhancer | Clean | Corruption |
|------------------|----------|-------|------------|
| ResNet50         | URIE     | 74.71 (-1.42) | 44.19 (4.62) |
|                  | LP       | 76.13 (0.00)  | 40.29 (0.72) |
|                  | DynTTA   | 76.04 (-0.09) | 44.87 (5.30) |
| DeiT-base        | URIE     | 80.69 (-1.06) | 62.69 (1.07) |
|                  | LP       | 80.76 (-0.98) | 60.38 (-1.34) |
|                  | DynTTA   | 81.75 (0.01)  | 62.71 (1.08) |

TABLE 5. Classification accuracy in the blind setting on the ImageNet dataset variants. The top table shows the results using ResNet50 as the classification model for an in-depth analysis original. The numbers in parentheses indicate the differences from the baseline.

| ResNet50 Enhancer | Stylized | A | R |
|-------------------|----------|---|---|
| URIE              | 9.38 (2.00) | 0.59 (0.59) | 38.17 (2.00) |
| LP                | 7.16 (-0.01) | 0.01 (0.01) | 36.18 (0.01) |
| DynTTA            | 9.80 (2.63) | 0.16 (0.16) | 37.48 (1.31) |

| DeiT Enhancer | Stylized | A | R |
|---------------|----------|---|---|
| URIE          | 19.93 (1.91) | 24.99 (-2.86) | 46.96 (1.61) |
| LP            | 16.23 (-1.80) | 25.73 (-2.12) | 44.17 (-1.18) |
| DynTTA        | 20.12 (2.09) | 26.81 (0.76) | 45.64 (0.29) |

of the test domain was treated as a given unknown dataset to estimate effective data augmentations. Note that a small split was not used in the final evaluation. For a fair comparison with an equal number of augmentations, a scenario involving original 13 augmentations (as outlined in the official AugMix code) was compared to a scenario comprising 9 default augmentations plus 4 estimated augmentations. The accuracy in each scenario is shown in Table 7. The estimated data augmentations consistently improve accuracy over the original.

D. DETAILED EXPERIMENTS
This section describes experiments on the CUB dataset using ResNet50 as the classification model for an in-depth analysis of DynTTA.

1) Effects of Key Ideas
We conducted an ablation study of our key ideas using ResNet50 on the CUB dataset and compared it to LP. LP selects the best augmentation for a test image by learning the classification losses of 12 augmented images as a label. One of our key ideas, “Blending”, improves upon LP by using not one image but a combination of many augmented images. “Blending” is lightweight because it does not require the computation of classification losses, which has the advantage of being easily extendable in augmentation space. Specifically, LP requires the same number of classification model inferences as the number of data augmentations used, while “Blending” requires only one. Here we define DynTTA (BL), which has the same augmentation and magnitude parameters as LP (see footnote) and outputs only blend weights. The effect of the “Blending” was measured by using DynTTA (BL). In addition, to measure the effect of “Optimization”, we define DynTTA (BL+OPT) that extends the DynTTA (BL) and simultaneously outputs magnitude parameters. At this time, the magnitude parameter ranges were set the same as in Table 2 except for Zoom, and Identity was not used because it was included in some data augmentations (e.g., a rotation of 0 degrees). The results are shown in Table 8. DynTTA (BL) is much more robust than LP, meaning that blending multiple images is better than choosing one best image. In the non-blind setting, DynTTA (BL+OPT) is less robust than DynTTA (BL). This is because the magnitude parameters used by LP and “Blending” may have been tuned by the authors for their non-blind setting experiment. On the other hand, in the blind setting, DynTTA (BL+OPT) is more robust than DynTTA (BL). This is because “Optimization” automatically finds the local optimal magnitude parameters for the blind setting using gradient descent (here we use Adam [22], see Section III and Appendix for details). Moreover, “Optimization” eliminates magnitude hyperparameters. For example, LP has 10-magnitude hyperparameters, but these are difficult to tune in the blind setting. “Optimization” eliminates these hyperparameters, making DynTTA more practical.

2) Avoiding Overfitting of DNN-based Transformation
When a model overfits, it tends to use the same set of transformations more (less diverse), which happens in URIE. The mean squared error (MSE) between the raw and enhanced images was measured as the amount of image transformation to show that DynTTA reduces overfitting. The results are shown in Fig. 2. URIE almost always shows the same MSE, regardless of severity, and performs almost the same set of transformations on any image, resulting in useless transformations even on clean images. In contrast, DynTTA shows a large MSE at high severity, a small MSE at low severity, and a particularly small MSE for clean images. Furthermore, DynTTA has a large variance of MSEs, indicating that it learns to use different transformations depending on the severity and type of corruption. The data augmentations introduced by DynTTA work as a hint to learn diverse trans-

Identity; [-20, 20] Rotate; [0.8, 1.2] Zoom; [0.5, 2.0] Saturate; Auto-contrast; [0.2, 0.5, 2.0, 4.0] Sharpness
formations, avoiding overfitting and enhancing images under various distributions.

3) Backbone Network of DynTTA
Because any neural network can be used as the backbone network of DynTTA, the effect of backbone network selection was examined. Four models were used as backbone network: ResNet18, ResNet50 [10], EfficientNet-b0 [50], and MobileNetV3 [18]. The results are shown in the Table 9. ResNet18 shows the highest robustness in the non-blind setting. EfficientNet-b0 and MobileNetV3 are more robust than ResNet in the blind setting, but their clean accuracy degrades more in the non-blind setting. ResNet50 has higher classification accuracy than ResNet18, but there is no correlation between the accuracy of the model as a classification model and the performance of the model as an enhancement model.

4) Performance Evaluation Using a Classification Model Different from the One Used During Training
This section discusses the generalizability of image enhancement models to classification models other than what was used for its training. Usually, there is a one-to-one correspondence between image enhancement models and classification models. We investigated whether an image enhancement model trained together with one classification model is also beneficial for another classification model. The results of the image enhancement models trained with ResNet50, MLP-Mixer, and DeiT in the non-blind setting on the CUB dataset are shown in Table 10. URIE trained on ResNet50 significantly reduces clean accuracy and does not improve robustness. URIE trained with MLP-Mixer or DeiT reduces clean accuracy by about 1-2 percentage points but improves robustness. LP improves robustness with almost no reduction in clean accuracy. DynTTA trained with ResNet50 reduces clean accuracy about 1 percentage point but improves robustness. DynTTA trained with MLP-Mixer or DeiT improves robustness over the comparison methods with almost no reduction in clean accuracy. Our experimental results show that when the coupled classification model is a high accuracy model such as MLP-Mixer or DeiT, the image enhancement model has high generalizability. In particular, DynTTA improves robustness over the comparison methods with almost no reduction in clean accuracy, which shows a high generalizability. DynTTA trained with a highly accurate classification model shows its potential to be applied to a variety of classification models.

5) Effects of Augmentation Space
In this study, the 14 data augmentations in Table 2 were used for DynTTA. We investigated the contribution of each data augmentation to classification accuracy in the non-blind

### TABLE 6.
Classification accuracy of ResNet50 in the blind setting on the PACS datasets. Each column title indicates the training domain, and the numerical values represent the average accuracy in the test domains. The numbers in parentheses indicate the differences from the baseline.

| Enhancer | Art       | Cartoon    | Photo      | Sketch     | Avg.       |
|----------|-----------|------------|------------|------------|-----------|
| URIE     | 77.81 (2.48) | 80.48 (3.30) | 51.06 (5.59) | 33.60 (8.90) | 60.74 (5.07) |
| DynTTA   | **78.65** (3.32) | **80.77** (3.59) | **50.33** (4.86) | **34.27** (9.57) | **61.01** (5.34) |

### TABLE 7.
Classification accuracy of retrained ResNet50 on the PACS datasets. The numbers in parentheses indicate the differences from the baseline.

| Augmentation | Art       | Cartoon    | Photo      | Sketch     | Avg.       |
|--------------|-----------|------------|------------|------------|-----------|
| Original     | 75.62 (0.29) | 79.01 (1.83) | 47.07 (1.60) | 32.80 (8.10) | 58.62 (2.95) |
| Estimated    | **78.03** (2.70) | **79.88** (2.70) | **52.84** (7.37) | **37.63** (12.93) | **62.10** (6.43) |

### TABLE 8.
Measuring the effect of our two key ideas on the CUB dataset. The top table shows the results for the non-blind setting, and the bottom table shows the results for the blind setting. The numbers in parentheses indicate the differences from the LP.

| Enhancer   | Non-blind | Blind |
|------------|-----------|-------|
| DynTTA (BL) | Clean | Corruption |
|            | 81.64 (-0.06) | 52.49 (1.61) |
| DynTTA (BL+OPT) | Clean | Corruption |
|            | 81.68 (-0.02) | 52.15 (1.27) |
| DynTTA (BL+OPT) | Clean | Corruption |
|            | 81.71 (-0.04) | 51.12 (2.51) |
| DynTTA (BL+OPT) | Clean | Corruption |
|            | 81.73 (-0.02) | 51.51 (2.90) |

### TABLE 9.
Classification accuracy for each backbone network on the CUB dataset. The top table shows the results for the non-blind setting, and the bottom table shows the results for the blind setting. The numbers in parentheses indicate the differences from the baseline.

| Non-blind Backbone Network | Clean | Corruption |
|----------------------------|-------|------------|
| ResNet18                   | 81.58 (-0.13) | 58.02 (9.62) |
| ResNet50                   | 81.74 (0.03) | 57.56 (9.16) |
| EfficientNet-b0            | 81.27 (-0.44) | 57.37 (8.97) |
| MobileNetV3                | 81.08 (-0.63) | 57.65 (9.25) |

| Blind Backbone Network | Clean | Corruption |
|------------------------|-------|------------|
| ResNet18               | 81.65 (-0.06) | 51.97 (3.39) |
| ResNet50               | 81.73 (0.02) | 52.07 (3.49) |
| EfficientNet-b0        | 81.73 (0.02) | 52.95 (4.37) |
| MobileNetV3            | 81.73 (0.02) | 53.00 (4.42) |
setting using ResNet50 on the CUB dataset. Fig. 3 shows the accuracy when each augmentation was excluded from DynTTA before training. All data augmentations, except Equalize, contribute to improving robustness with almost no reduction in clean accuracy. In particular, URIE significantly contributed to improving robustness, but it tends to degrade clean accuracy. DynTTA blends URIE with other data augmentations to maintain clean accuracy. This result indicates that each data augmentation contributes differently to robustness, with some being effective and others being ineffective. Effective data augmentations can be predefined when the type of distribution shift is known in advance, as in the non-blind setting. For example, we estimate that Sharpness is effective in environments where blurring often occurs.

6) Effects of Magnitude Range
DynTTA requires predefined of the magnitude range $M$. We investigated the effect of different magnitude ranges on classification accuracy. In the experiments in the blind setting here, we also experimented with the stylized CUB dataset [7] in addition to the corruption dataset. The ranges in Table 2

![FIGURE 2. Measuring diverse transformations by mean and standard error of the MSE. ResNet50 was used as the classification model in the blind setting on the CUB dataset.](image)

![FIGURE 3. Impact of individual augmentation on classification accuracy. The y-axis is the difference between excluding one augmentation and using all data augmentations. Positive values mean that they contribute to improved accuracy. A-Contrast, LPFs, and HPFs denote auto-contrast, low-pass filters, and high-pass filters, respectively.](image)
TABLE 11. Effect of magnitude range on classification accuracy on the CUB dataset. The top table shows the results for the non-blind setting, and the bottom table shows the results for the blind setting. The numbers in parentheses indicate the differences from the standard range DynTTA.

| Range      | Clean | Unseen |
|------------|-------|--------|
| **Non-blind** |       |        |
| Small      | 81.58 (0.00) | 58.41 (-0.39) |
| Large      | 81.26 (-0.32) | 57.80 (-0.22) |
| **Blind**  |       |        |
| Small      | 81.65 (0.00) | 52.62 (0.65) | 18.35 (0.19) |
| Large      | 81.61 (-0.04) | 52.32 (0.35) | 18.50 (0.34) |

multiplied by 1.5 and 0.5 are referred to as “Large” and “Small”. The results are shown in Table 11. In the blind setting, “Small” shows a better result for the corruption dataset and “Large” for the stylized dataset. The clean accuracy of “Large” is a little lower than the others. A large magnitude range is observed to be effective for large distribution shifts such as stylized datasets, but large image transformations degrade clean accuracy. In the non-blind setting, “Small” is more robust than the standard range. This may have been caused by “Small” having a smaller search space and the optimization being more stable.

7) Visualization of DynTTA Output
The output of DynTTA in the non-blind setting is visualized. Fig. 4 shows augmented images and output image by DynTTA for level-five Unseen corruptions. The bottom two rows of augmented images are almost black, except for the bottom right, so it is hard to see what they show, but this is because high-pass filters remove low-frequency domains that are visible to humans. Speckle Noise image is denoised and Gaussian Blur image is sharpened. The corruptions in the Spatter and Saturate images have not been removed, but the shape and texture of the object are enhanced.

V. LIMITATIONS
Although DynTTA significantly improves robustness with almost no reduction in clean accuracy, it has computational overhead. First, to discuss network overhead, Table 12 presents the number of parameters and multiply-accumulate operations (MACs). DynTTA exhibits nearly identical numbers of parameters and MACs as URIE, yet it significantly improves performance and achieves high robustness with fewer overheads than ResNet152. Moreover, URIE (Large), which increases the number of layers in URIE, shows no performance improvement. These results indicate that data augmentations integrated by DynTTA are much more effective with little overhead than increasing model size. Next, the computational cost of performing many data augmentation runs is discussed. The Appendix presents a technique designed to decrease the number of data augmentation runs required. While we acknowledge that we have not completely solved this issue, which remains a limitation of our work, we maintain that it does not affect the practical usability of DynTTA. As an example, utilizing the idle resources of edge devices for image enhancement effectively mitigates the concern over computational expenses [30].

VI. CONCLUSION
In this paper, we proposed DynTTA, a novel image enhancement method based on differentiable data augmentation techniques and image blending. DynTTA uses a gradient descent algorithm to find magnitude parameters and blend weights from a huge augmentation space. This augmentation space includes deep neural networks (DNNs) such as URIE as well as standard data augmentations. Image enhancement with DNN and data augmentation can learn diverse transformations and thus avoid overfitting. By applying DynTTA before inference by the pretrained classification model, DynTTA improves robustness while maintaining clean accuracy. In addition to the existing scenario evaluated under the assumption of knowing the type of test-time distribution shift, we introduced a practical training and evaluation scenario that does not assume the type of test-time distribution shifts. Our experimental results show that DynTTA outperforms existing methods and is effective in practical settings. Furthermore, DynTTA estimates effective training-time augmentations for the distribution-shifted datasets and shows that retraining with estimated augmentations significantly improves accuracy. However, the overhead during DynTTA inference has not been completely solved and is a limitation of this study. In this study, we experimented only with the image classification task. We will apply and evaluate DynTTA to other practical tasks such as object detection and segmentation. Future work includes exploring the optimal backbone network architecture for DynTTA and adding differentiable data augmentation to further improve robustness.

REFERENCES
[1] Arsenii Ashukha, Andrei Atanov, and Dmitry Vetrov. Mean embeddings with test-time data augmentation for ensembling of representations. arXiv preprint arXiv:2106.08038, 2021.
[2] Christian Clausner, Apostolos Antonacopoulos, and Stefan Pletschacher. Icdar2019 competition on recognition of documents with complex layouts - rdc2019. In International Conference on Document Analysis and Recognition (ICDAR), 2019.
[3] Steven Diamond, Vincent Sitzmann, Frank Juca-Aguilar, Stephen Boyd, Gordon Wetzstein, and Felix Heide. Dirty pixels: Towards end-to-end image processing and perception. ACM Transactions on Graphics (TOG), 2021.
FIGURE 4. DynTTA output images. From top to bottom: Speckle Noise, Gaussian Blur, Spatter, and Saturate. The images in each row are, from left to right: input image, augmented images, output image, and difference between input and output images. For ease of viewing, auto-contrast is used for the high-pass filters image. The numbers in the input and output images indicate the loss values.
driving in bad weather. In IEEE International Conference on Robotics and Automation (ICRA), 2021.

[26] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and tighter domain generalization. In IEEE/CVF International Conference on Computer Vision (ICCV), 2017.

[27] Yingwei Li, Qiang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, et al. Shape-texture debiased neural network training. In International Conference on Learning Representations (ICLR), 2020.

[28] Yanggang Li, Guosheng Hu, Yongtao Wang, Timothy M. Hospedales, Neil Martin Robertson, and Yongxin Yang. DADA: differentiable automatic data augmentation. In European Conference on Computer Vision (ECCV), 2020.

[29] Ding Liu, Bihan Wen, Xiainning Liu, Zhangyang Wang, and Thomas S. Huang. When image denoising meets high-level vision tasks: A deep learning approach. In International Joint Conference on Artificial Intelligence (IJCAI), 2018.

[30] Yan Lu, Shiqi Jiang, Ting Cao, and Yuanchao Shu. Turbo: Opportunistic enhancement for edge video analytics. In ACM Conference on Embedded Networked Sensor Systems, 2022.

[31] Alexander Lyzhov, Yuliia Molchanova, Arseniy Ashukha, Dmitry Molchanov, and Dmitry Vetrov. Greedy policy search: A simple baseline for learnable test-time augmentation. In Conference on Uncertainty in Artificial Intelligence (UAI), 2020.

[32] Jieru Mei, Yucheng Han, Yutong Bai, Yixiao Zhang, Yingwei Li, Xianhang Li, Alan Yuille, and Cihan Xie. Fast advprop. In International Conference on Learning Representations (ICLR), 2022.

[33] Eric Mintun, Alexander Kirillov, and Saining Xie. On interaction between augmentations and corruptions in natural corruption robustness. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[34] Takayuki Okatani, Xing Liu, and Masanori Suganuma. Improving generalization ability of deep neural networks for visual recognition tasks. In Computational Color Imaging Workshop (CCW), 2019.

[35] Tianyu Pang, Kun Xu, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. In International Conference on Learning Representations (ICLR), 2019.

[36] Yanting Pei, Yaping Huang, Qi Zou, Yuhang Lu, and Song Wang. Does haze removal help cnn-based image classification? In European Conference on Computer Vision (ECCV), 2018.

[37] Yanting Pei, Yaping Huang, Qi Zou, Xingyuan Zhang, and Song Wang. Effects of image degradation and degradation removal to cnn-based image classification. In IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2021.

[38] Juan C Perez, Motasem Alfarra, Guillaume Jeanneret, Laura Rueda, Ali Thabet, Bernard Ghanem, and Pablo Arbelaez. Enhancing adversarial robustness via test-time transformation ensembling. In IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), 2021.

[39] Yongming Rao, Weiliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[40] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2020.

[41] Cédric Rommel, Thomas Moreau, and Alexandre Gramfort. Deep invariant networks with differentiable augmentation layers. In Advances in Neural Information Processing Systems (NeurIPS), 2022.

[42] Evgeniya Rusak, Lukas Schott, Roland S Zimmermann, Julian Bitterwolf, Oliver Bringmann, Matthias Bethge, and Wieland Brendel. A simple way to make neural networks robust against diverse image corruptions. In European Conference on Computer Vision (ECCV), 2020.

[43] Olga Russakovskiy, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. International journal of computer vision (IJCV), 2015.

[44] Divya Shanmugam, Davis Blalock, Guha Balakrishnan, and John Guttag. Better aggregation in test-time augmentation. In IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

[45] Vivek Sharma, Ali Diba, Davy Neven, Michael S. Brown, Luc Van Gool, and Rainer Stiefelhagen. Classification-driven dynamic image enhancement. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[46] Ryan Soklaski, Michael Yee, and Theodoros Tsiligkaridis. Fourier-based augmentations for improved robustness and uncertainty calibration. In Workshop on Distribution Shifts: Connecting Methods and Applications (DistShift), 2021.

[47] Taeyoung Son, Jiwon Kang, Namyup Kim, Sunghyun Cho, and Suha Kwa. Uire: Universal image enhancement for visual recognition in the wild. In European Conference on Computer Vision (ECCV), 2020.

[48] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[49] Hossein Talebi and Peyman Milanfar. Learning to resize images for computer vision tasks. In IEEE/CVF International Conference on Computer Vision (CVPR), 2021.

[50] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Conference on Machine Learning (ICML), 2019.

[51] Ilya O Tolskikhin, Neil Houlsby, Alexandre Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mpl architecture for vision. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[52] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jougou. Training data-efficient image transformers & distillation through attention. In International Conference on Machine Learning (ICML), 2021.

[53] Theodoros Tsiligkaridis and Athanasios Tsiligkaridis. Diverse gaussian noise consistency regularization for robustness and uncertainty calibration. In International Joint Conference on Neural Networks (IJCNN), 2023.

[54] Puru Vashisht, Shunxin Wang, and Nicola Strisciuglio. Fourier-basis functions to bridge augmentation gap: Rethinking frequency augmentation in image classification. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024.

[55] Cristina Vasconcelos, Hugo Larochelle, Vincent Dumoulin, Rob Romijnders, Nicolas Le Roux, and Ross Goroshin. Impact of aliasing on generalization in deep convolutional networks. In IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

[56] Haotao Wang, Chaowei Xiao, Jean Kossaifi, Ziheng Yu, Anima Anandkumar, and Zhangyang Wang. Anmax: Adversarial composition of random augmentations for robust training. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[57] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical report, California Institute of Technology, 2010.

[58] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le. Adversarial examples improve image recognition. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[59] Xueyi Yang, Zeng Zeng, Sin G Teo, Li Wang, Vijay Chandrasekhar, and Steven Hoi. Deep learning for practical image recognition: Case study on Kaggle competitions. In ACM SIGKDD international conference on knowledge discovery & data mining (KDD), 2018.

[60] Dong Yin, Raphael Gontijo Lopes, Jon Shiens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. In Advances in Neural Information Processing Systems (NeurIPS), 2019.

[61] Jinghao Zhang, Zhenhua Feng, Guosheng Hu, Changbin Shao, and Yaochu Jin. Mixprop: Towards high-performance image recognition via dual batch normalisation. In British Machine Vision Conference (BMVC), 2022.

[62] Xueyan Zou, Fanyi Xiao, Zhiding Yu, and Yong Jae Lee. Delving deeper with convolutions. In European Conference on Computer Vision (ECCV), 2018.
Monikka Roslianna Busto is a Researcher at NTT Software Innovation Center. She graduated from the Electrical and Electronics Engineering Institute, College of Engineering, the University of the Philippines in 2017, and received a master’s degree from the Department of Information and Communications Engineering, Tokyo Institute of Technology in 2021. She joined Nippon Telegraph and Telephone Corporation in the same year. Her research interests include computer vision, collaborative intelligence for edge computing, remote sensing image analysis and multi-modal AI.

Takeharu Eda received his B.S. in Mathematics from Kyoto University in 2001 and his M.S. in Engineering from the Nara Institute of Science and Technology in 2003. He joined NTT Laboratories in 2003, focusing on research in various aspects of machine learning and systems.

In 2011, he transitioned to NTT Communications, where he launched a web hosting service utilizing CloudStack/OpenStack-based infrastructure and migration tools. He managed international development teams with members from the US, Germany, and Japan.

In 2015, he moved to the NTT Software Innovation Center, developing a scalable surveillance video system using deep learning-based computer vision techniques. Since 2022, he has been involved in the research and development of a space computing project with SpaceCompass.

Eda is a member of the Information Processing Society of Japan (IPSJ), the Association for Computing Machinery (ACM), and IEEE.

***
In this Supplementary Material, we provide additional analysis for the DynTTA.

APPENDIX. RELATED WORK

A. APPLICATIONS OF IMAGE ENHANCEMENT

We present applications using image enhancement. Image enhancement has the benefit of not requiring the classification model to be retrained. By taking advantage of this benefit, use cases have been proposed to improve the performance of models in the cloud, where only APIs are provided [11], [14]. In addition, model retraining involves the risk of catastrophic forgetting [2], [9], [12], but image enhancement avoids this, making image enhancement more useful than model updating [4], [5], [13].

APPENDIX. COMPUTATIONAL COST REDUCTION FOR DYNNTA

DynTTA is more computationally expensive than Loss Predictor because it performs many data augmentations during inference. To reduce the inference cost of DynTTA, we tried to avoid executing data augmentations with small blend weights. With this method, we achieved a 72% reduction in the number of data augmentation executions while maintaining the accuracy of DynTTA. We describe the details here.

First, we set a threshold. If all the blend weights of a data augmentation in a mini-batch are below the threshold, we do not execute that data augmentation. With this method, we achieved a 72% reduction in the number of data augmentation executions while maintaining the accuracy of DynTTA. We describe the details here.

Table 13 shows the classification accuracy and average number of data augmentation executions per inference when varying the threshold. The numbers in parentheses indicate the differences from the original DynTTA (threshold is 0).

| Threshold | Number of Executions | Clean | Corruption |
|-----------|----------------------|-------|------------|
| 0.005     | 34.76 (-15.24)       | 81.58 (0.05)  | 36.61 (0.71) |
| 0.01      | 28.71 (-21.29)       | 81.65 (0.12)  | 36.68 (0.78) |
| 0.02      | 21.85 (-28.15)       | 81.36 (-0.17) | 35.81 (-0.09) |
| 0.05      | 13.94 (-36.06)       | 81.15 (-0.38) | 35.92 (0.02)  |
| 0.1       | 9.29 (-40.71)        | 80.95 (-0.58) | 34.58 (-1.32) |
| 0.2       | 3.88 (-46.12)        | 80.88 (-0.65) | 19.89 (-16.01) |

Results are shown in Table 13. When the threshold is 0, DynTTA executes all 50 data augmentations. When the threshold is less than 0.01, DynTTA reduces the average number of data augmentation executions further improving accuracy. We suspect this to be because we were able to prevent the execution of data augmentations that should not have been executed, which were output with blend weights near zero. When the threshold is 0.05, DynTTA reduces the number of data augmentation executions by about 72% while maintaining accuracy. When the threshold is greater than 0.1, DynTTA reduces the number of data augmentation executions, but it also reduces the accuracy.

APPENDIX. DETAILS OF DATASET AND TRAINING

A. CLASSIFICATION ON THE CUB

CUB is an image dataset of 200 class bird species. It consists of 5994 training images and 5794 test images. We used ResNet50 [6], Mixer-B16 [15] and DeiT-base [16] as the classification model. For pretrained ResNet we used the weight trained on IMAGENET1K_V1 from torchvision.models [10]. For pretrained Mixer and Deit, we downloaded the weight from official github. For optimization, we employ Adam [8] with learning rate 0.001 for URIE, 0.001 for DynTTA and
Loss Predictor, and decay the learning rate every 10 epochs by 2. These models are trained in 60 epochs.

B. CLASSIFICATION ON THE IMAGENET

ImageNet is the most widely used 1000-class image recognition dataset. It is composed of 1.3 million training images and 50,000 test images. We used ResNet50 and DeiT as the classification model. For optimization, we employ Adam with learning rate 0.001 for URIE, 0.0001 for DynTTA and Loss Predictor, and decay the learning rate every 8 epochs by 10. These models are trained in 30 epochs.

C. CLASSIFICATION ON THE DOMAIN GENERALIZATION DATASETS

PACS consists of four domains (art, cartoons, photos, sketches), 9,991 images and 7 classes. VLCS [3] consists of four photographic domains (Caltech101, LabelMe, SUN09, VOC2007), 10,729 images and 5 classes. OfficeHome [17] consists of four domains (art, clipart, product, real), 15,588 images and 65 classes. TerraIncognita [1] consists of four camera trap domains (L100, L38, L43, L46), 24,788 images and 10 classes. We used ResNet50 as the classification model. For optimization, we employ SGD with learning rate 0.0005 and batch size is 96. Since we used DomainBed, all other settings follow DomainBed. We show in the Figure 5 the details of splitting the dataset for training, validation, and testing.

APPENDIX. EFFECTS OF OPTIMIZATION AND BLENDING

We show the results of the simple test-time augmentation using random magnitude parameters and random blend weights in Table 14. We used the same data augmentation as DynTTA. All test-time augmentations using random magnitude are less accurate, except for A-Contrast. Random blending is also less accurate than without test-time augmentation. These results indicate the validity of the magnitudes and weights estimated by DynTTA.

APPENDIX. COMPARISON WITH SIMPLE TEST-TIME AUGMENTATION

The Loss Predictor paper made a comparison with some simple test-time augmentation (center crop, horizontal flip, random augmentation selection), and the results are that the Loss Predictor is better. Therefore, DynTTA should also be better than simple test-time augmentation. On the other hand, training time augmentation, which randomly combines (blends) multiple images to improve robustness (e.g., AugMix [7], MixUp [18]), has been proposed. Random blending of multiple images during training has the effect of improving robustness, however, it is not clear whether it has the same effect during testing. To verify the effect of randomly blending images during testing, we experimented with test-time AugMix using ResNet50 in the blind setting on the CUB dataset. The resulting clean accuracy is 67.64% and corruption accuracy is 36.12%, which is significantly worse than the baseline (without image enhancement model). This result indicates that simple randomly blending is ineffective and that blending trained by DynTTA is effective in improving robustness.

APPENDIX. COMPARISON WITH TRAINING ROBUST MODEL ALGORITHMS

We compare DynTTA with algorithms for training robust classification models. AguMix, MixUp, and CutMix are data augmentations that improve data diversity by mixing multiple images. MixProp, Fast AdvProp, and AdvProp are disentangled learning methods that improve robustness by training out-of-distribution data through auxiliary batch normalization layers. We show the results in Table 15. Note here that we used 15 corruptions in our evaluation. DynTTA can be used in conjunction with these algorithms to further improve robustness.

APPENDIX. CLASSIFICATION ON THE OTHER DOMAIN GENERALIZATION DATASETS

We show the experimental results on the VLCS dataset, the OfficeHome dataset and the TerraIncognita dataset in Table 16. DynTTA outperforms URIE on all datasets except VLCS.
TABLE 15. Comparison with robust training algorithms on the CUB dataset. We trained DynTTA in the blind setting.

| Algorithm        | Clean | 15 Corruption |
|------------------|-------|---------------|
| Vanilla          | 81.71 | 48.40         |
| AugMix           | 82.55 | 59.97         |
| MixUp            | 83.04 | 58.52         |
| CutMix           | 81.65 | 48.11         |
| MixProp          | 83.77 | 56.80         |
| Fast AdvProp     | 82.90 | 51.22         |
| AdvProp          | 81.45 | 53.44         |
| Vanilla+DynTTA   | 81.65 | 51.38         |
| AugMix+DynTTA    | 82.84 | 60.59         |

APPENDIX. RETRAINING WITH ESTIMATED AUGMENTATION ON THE CUB

In the blind setting, we split the level five four corruption (speckle noise, Gaussian blur, spatter, and saturation) test set into a validation set and a test set in a 2:8 ratio. Figure 6 shows a visualization of the magnitude parameters and the blend weights, which are the outputs of DynTTA for the validation set. DynTTA outputs high blend weights of low-pass filters, which are effective for high-frequency corruptions such as Speckle Noise and Spatter, and of saturation, which is effective for Saturate. The maximum weights of low-pass filters, gamma, contrast, and saturation are high for validation set. The magnitude parameters of rotate and scale have high variance. This is because when the blend weight is close to zero, any value of the magnitude parameter has almost no effect on the output image.

Next, from the test-time augmentation that was effective for the validation set, we estimate the training-time augmentation that is effective for the target corruption. Based on the results in Figure 6, we focus on the maximum value of the blend weights rather than the magnitude parameter. We choose the four data augmentations used to retrain the classification model: high-pass filters, gamma, contrast, and saturation, which are the inverse operations of four data augmentations that were effective for validation set. We experiment by varying the data augmentation space used by the AugMix. The results of the test set are shown in Table 17. “Normal” is the normal AugMix and uses nine data augmentations. “All” is the normal nine data augmentations plus saturation, contrast, brightness and sharpness, referenced from the official AugMix implementation and “Estimated” is the normal nine data augmentations plus saturation, contrast, high-pass filters and gamma. The classification model trained with “Estimated” achieved the best accuracy for all corruptions except Saturate. On average, it was approximately 10 points more accurate than “Normal”, and, in particular, approximately 20 points more accurate than “Normal” for Speckle Noise.

APPENDIX. URIE SIZE AND PERFORMANCE

We show the relationship between the size and performance of URIE in Table 18 URIE (small) means one SKDown and

APPENDIX. ENSEMBLE OF MULTIPLE DATA AUGMENTATIONS

Loss Predictor improves accuracy by ensembling k candidate augmentations and further improves accuracy by ensembling with hflip. In Table 19 we show the accuracy of Loss Predictor (k = 2) in the non-blind setting, and also in combination with hflip.

The increase in clean accuracy and robustness with k = 2 is +0.44 pt and +0.09 pt, respectively, and k = 1 was used in all main experiments due to the significantly lower robustness of Loss Predictor (k = 2) compared to URIE and DynTTA. Loss Predictor also improved clean accuracy and robustness by +0.84 pt and +0.69 pt, respectively, when combined with hflip. However, hflip is also applicable to DynTTA, which improved by +0.74 pt and +0.70 pt, respectively.

APPENDIX. POTENTIAL INTEGRATION OF DYNTTA WITH OTHER MACHINE LEARNING FRAMEWORKS

Even if it is a machine learning algorithm other than a neural network, DynTTA can be applied to any algorithm that is trained by gradient descent with a loss function. Furthermore, if a differentiable data augmentation can be defined for modalities other than image (e.g., language), DynTTA can be applied.

APPENDIX. VISUAL COMPARISON

We show in Figure 7 the corrupted image, the enhanced image, and the difference between the enhanced image and the corrupted image. URIE transforms any corruption in such a way that the object shape is enhanced. DynTTA performs noise-removing transformations for noise corruption.
TABLE 16. Classification accuracy of ResNet50 in the blind setting on the domain generalization datasets.

| Enhancer | VLCS   | PACS   | OfficeHome | TerraIncognita | Average |
|----------|--------|--------|------------|----------------|---------|
| URIE     | 63.94 (2.40) | 60.74 (5.07) | 55.69 (0.41) | 33.75 (1.27) | 53.53 (2.29) |
| DynTTA   | 63.91 (2.37) | 61.01 (5.34) | 55.77 (0.49) | 34.87 (1.39) | 53.89 (2.65) |

TABLE 17. Classification accuracy on level five test set when varying the augmentation space used by the AugMix. Average indicates the average accuracy of the four corruptions.

| Augmentation | Speckle Noise | Gaussian Blur | Spatter | Saturate | Average |
|--------------|---------------|---------------|---------|----------|---------|
| Normal       | 50.27         | 30.49         | 35.33   | 36.75    | 38.21   |
| All          | 49.90         | 33.18         | 35.61   | 44.97    | 40.92   |
| Estimated    | 70.25         | 37.22         | 43.36   | 42.74    | 48.39   |

TABLE 18. Classification accuracy, the number of parameters and MACs in the blind setting on the CUB dataset. We use ResNet50 as the classification model.

| Enhancer      | Clean | Corruption | Params (M) | MACs (G) |
|---------------|-------|------------|------------|----------|
| URIE (Small)  | 80.37 | 71.15      | 25.67      | 5.75     |
| URIE          | 80.68 | 71.47      | 26.23      | 7.16     |
| URIE (Medium) | 80.61 | 71.21      | 28.47      | 8.56     |
| URIE (Large)  | 80.61 | 71.47      | 37.43      | 9.96     |

TABLE 19. Additional experimental results for ResNet50 on the CUB dataset in the non-blind setting. The numbers in parentheses indicate the difference from the normal enhancement. LP indicates Loss Predictor.

| Enhancer      | Clean | Corruption     |
|---------------|-------|----------------|
| LP(k = 2)     | 82.14 (0.44) | 50.97 (0.09)   |
| LP(k = 2) + hflip | 82.54 (0.84) | 51.57 (0.69)   |
| DynTTA + hflip | 82.32 (0.74) | 58.72 (0.70)   |

and color transformations for saturation corruption. DynTTA performs diverse transformations depending on the type of corruption.

APPENDIX. DETAILS OF THE CUB DATASET EXPERIMENT

In this section, we show the accuracy for each corruption and severity results of the CUB dataset experiment presented in Section 4.2. The results of the non-blind setting are shown in Figure 8-13. For Gaussian Noise and Gaussian Blur, DynTTA improves accuracy over comparison methods regardless of severity. For Spatter, DynTTA has about the same improvement in accuracy as the comparison method. For Saturate, DynTTA improves accuracy at higher severity compared to the comparison methods.

The results of the blind setting are shown in Figure 14-19. Several corruptions are more difficult for Loss Predictor to improve compared to other methods due to its small augmentation space. DynTTA outperforms the comparison methods in most experiments. In particular, in experiments where the classification model is Mixer-B16 w/AugMix and the corruptions are Speckle Noise and Contrast, DynTTA improves accuracy while other methods do not.
From top to bottom: speckle noise, Gaussian blur, spatter, and saturation, all at a severity level of five. We trained the enhancer in the blind setting.
FIGURE 8. Classification accuracy of ResNet50 for each corruption and severity on the CUB dataset.

FIGURE 9. Classification accuracy of ResNet50 w/AugMix for each corruption and severity on the CUB dataset.

FIGURE 10. Classification accuracy of Mixer-B16 for each corruption and severity on the CUB dataset.

FIGURE 11. Classification accuracy of Mixer-B16 w/AugMix for each corruption and severity on the CUB dataset.
FIGURE 12. Classification accuracy of DeiT for each corruption and severity on the CUB dataset.

FIGURE 13. Classification accuracy DeiT w/AugMix for each corruption and severity on the CUB dataset.
FIGURE 14. Classification accuracy of ResNet50 for each corruption and severity on the CUB dataset.
FIGURE 15. Classification accuracy of ResNet50 w/AugMix for each corruption and severity on the CUB dataset.
FIGURE 16. Classification accuracy of Mixer-B16 for each corruption and severity on the CUB dataset.
FIGURE 17. Classification accuracy of Mixer-B16 w/AugMix for each corruption and severity on the CUB dataset.
FIGURE 18. Classification accuracy of DeiT for each corruption and severity on the CUB dataset.
FIGURE 19. Classification accuracy of DeiT w/AugMix for each corruption and severity on the CUB dataset.
REFERENCES

[1] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In European Conference on Computer Vision (ECCV), 2018.

[2] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In European Conference on Computer Vision (ECCV), 2020.

[3] Chen Fang, Ye Xu, and Daniel N. Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In IEEE/CVF International Conference on Computer Vision (ICCV), 2013.

[4] Yulu Gan, Xianzheng Ma, Yihang Lou, Yan Bai, Renrui Zhang, Nian Shi, and Lin Liao. Decorate the newcomers: Visual domain prompt for continual test time adaptation. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2023.

[5] Jin Gao, Jialing Zhang, Xihui Liu, Trevor Darrell, Evan Shelhamer, and Dequan Wang. Back to the source: Diffusion-driven test-time adaptation. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[7] 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. In International Conference on Learning Representations (ICLR), 2020.

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In International Conference on Learning Representations (ICLR), 2015.

[9] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 2017.

[10] TorchVision maintainers and contributors. Torchvision: Pytorch’s computer vision library. https://github.com/pytorch/vision, 2016.

[11] Changdae Oh, Hyeji Hwang, Hee-young Lee, Yong-Taek Lim, Geunyoung Jung, Jiyong Jung, Hosik Choi, and Kyungwoo Song. Blackvip: Black-box visual prompting for robust transfer learning. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023.

[12] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. Neural networks, 2019.

[13] Mihir Prabhudesai, Tsungwei Ke, Alex Li, Deepak Pathak, and Katerina Fragkiadaki. Test-time adaptation of discriminative models via diffusion generative feedback. Advances in Neural Information Processing Systems (NeurIPS), 2024.

[14] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Denoised smoothing: A provable defense for pretrained classifiers. Advances in Neural Information Processing Systems (NeurIPS), 2020.

[15] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

[16] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In International Conference on Machine Learning (ICML), 2021.

[17] Hernanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[18] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. In International Conference on Learning Representations (ICLR), 2018.

***