MixStyle-Based Contrastive Test-Time Adaptation: Pathway to Domain Generalization

Kota Yamashita
Meijo University
1-501 Shiojirikuchi,
Tenpaku, Nagoya 468-8502,
Japan
200442179@ccalumni.meijo-u.ac.jp

Kazuhiro Hotta
Meijo University
1-501 Shiojirikuchi,
Tenpaku, Nagoya 468-8502,
Japan
kazuhotta@meijo-u.ac.jp

Abstract

Recent advancements in domain generalization have increasingly focused on Test-time Adaptation (TTA), which adapts models to unknown domains during testing. Test-time Training (TTT) represents a prominent TTA approach, utilizing multi-task learning on training images by combining the main task with self-supervised tasks such as rotation prediction, and adapting the model to the test domain using only self-supervised tasks during testing. However, the selection of appropriate self-supervised tasks poses a challenge in TTT, as incorrect choices can degrade model performance. Common self-supervised tasks like rotation prediction are not specifically designed for domain generalization. TENT implements an unsupervised TTA technique utilizing entropy minimization without engaging in self-supervised tasks. Although it bypasses the need for self-supervised tasks, its performance can fall short of TTT in certain domains. To address TTT's challenges, we propose MixStyle-based Contrastive Test-time Adaptation (MCTTA) which employs the original method of MixStyle-based Contrastive Learning (MCL) to train feature extractors capable of extracting consistent features across different domains. The learning process is divided into Training and TTA phases. During the Training phase, the model is generalized to various domains through multi-task learning: main classification task and MCL. In the TTA phase, MCL is applied to the test data to adapt the feature extractor to the test domain. By experiments on the DomainBed benchmark library and three datasets (PACS, Office-Home, and Colored MNIST), MCTTA achieved the highest domain generalization accuracy, surpassing not only TTT but also other TTA methods and domain generalization methods.

1. Introduction

In recent years, deep learning technology has made remarkable progress with applications extending to autonomous driving, medical diagnostics, image recognition, and natural language processing. The advancements in this technology are supported by the significant increase in computational power provided by GPUs, the construction of large-scale datasets, and the development of new learning algorithms. While deep learning has achieved high accuracy in specific tasks, maintaining comparable performance across different domains and environments remains a challenging issue. To address this challenge, researches in domain generalization plays a vital role[23][19]. Domain generalization aims to develop models with high versatility that perform well across different domains, serving as a bridge between deep learning technology and real-world applications. Researches in domain generalization have evolved over decades through various approaches such as data augmentation, improvements in feature extraction techniques, and meta-learning. Data augmentation, for example, increases the diversity of training data, aiding models in learning generalized features. Meanwhile, meta-learning enhances the model's ability to learn efficiently from different tasks, improving adaptability to new domains[10]. Recently, TTA[13][20] has emerged as a focal point in domain generalization research. TTA enables models to adapt to unknown domain data during testing, ensuring consistent performance even on new domains that were not accessible during training. TTT[16] is a leading method within TTA, enhancing the generalization capability to new domains through multi-task learning that combines the desired task with a self-supervised task, such as rotation prediction. However, TTT faces several issues. Incorporating common self-supervised tasks, such as rotation prediction, into multi-task learning does not directly contribute to domain generalization. Moreover, choosing inappro-

priate self-supervised tasks risks degrading model performance. The reliance on intuition to select self-supervised tasks presents a significant obstacle in TTA, where quick adaptation is crucial. In contrast to approaches that incorporate self-supervised tasks, TENT[18] utilizes unsupervised learning on test data without engaging in multi-task learning like TTT during the training phase. Specifically, it focuses solely on the main task during training and adapts to the test data through unsupervised learning, employing entropy minimization. TENT offers a straightforward method by only requiring entropy minimization for the test data. However, it is important to note that its effectiveness can vary across domains, potentially resulting in lower accuracy compared to TTT in some cases.

We propose MCTTA as a simple yet powerful solution for TTT. Unlike conventional contrastive learning which relies on color transformations or random cropping, MCTTA employs a novel approach to contrastive learning, named MCL specialized for domain generalization. MCL utilizes MixStyle[22] in the feature space for data augmentation in order to consistently extract features across various domains. MCL employs two feature extractors initialized with different parameters but sharing the same architecture. This is the difference from conventional contrastive learning[6][7]. These extractors are not only pivotal in MCL but also enhance the stability and accuracy in classification through ensemble classification. The training process of MCTTA is divided into the Training phase, focusing on multi-task learning with ensemble classification and MCL for domain generalization, and the TTA phase, where MCL is applied to the test data to adapt the feature extractors to new data. This approach positions MCTTA as a learning algorithm specialized for domain generalization, capable of adapting to a wider range of domains compared to TTT and other TTA methods.

In experiments, we utilized the DomainBed[4] benchmark library alongside three datasets: PACS[9], Office-Home[17], and Colored MNIST[1]. Compared to TTT, our approach achieved an accuracy improvement of 1.2% on PACS, 2.9% on Office-Home, and 0.9% on Colored MNIST, with an average increase of 1.7% across these datasets. Furthermore, MCTTA outperformed other techniques, including TENT, which employs entropy minimization, SHOT[12], which uses pseudo-labeling for TTA, and conventional domain generalization methods that do not adapt the model at test time, establishing itself as the most accurate approach.

This research aims to endow models with further domain generalization capabilities by resolving issues with TTT, making three primary contributions:

- The introduction of MCL, a contrastive learning approach specifically designed for domain generalization.
- The use of two feature extractors with identical architectures but different initial values in MCL, which not only serves the purposes of MCL but also improves the stability and accuracy of class predictions through ensemble classification.
- Experiments on the DomainBed benchmark library show that the proposed MCTTA achieved higher accuracy than TTT, other TTA, and conventional domain generalization methods.

The structure of this paper is as follows. Section 2 discusses the main approaches to domain generalization and TTA as related work. Section 3 explains the details of our proposed method, MCTTT. Section 4 presents the experimental setup and results on the DomainBed and three datasets. Finally, Section 5 describe conclusion and future works.

2. Related Work

Domain Generalization. The objective of domain generalization research is to train models using data from different domains of test data, enabling these models to perform well on test data from unseen domains. As the application of deep learning technology expands across various fields, such as autonomous driving technology and medical image analysis, the ability of models to maintain stable performance across unknown domains is increasingly demanded. Domain generalization is anticipated to remain a vital research theme, bridging deep learning and real-world applications. Approaches to domain generalization can be broadly divided into three main categories, and our proposed method encompasses all of these categories.

Firstly, data-level approaches include techniques like data augmentation. For example, Mixup[21] creates new training images by blending images from different domains to prevent models from overfitting to domain-specific features. Unlike Mixup which mixes domain information at the image level, our proposed method employs MixStyle which mixes domain information at the feature level. MixStyle blends the mean and standard deviation of features from the shallow layers of CNN[11] across batches through instance normalization, producing features with intermixed domain information.

Secondly, there is a model-level approachs that focuses on the architecture of the model, designed to extract features common across different domains. For instance, Style-Agnostic Networks (SagNet)[14] employs two networks named the content-biased network and the style-biased network. Through these networks, SagNet separates the content (such as shape) from the style (e.g., color tones and textures) of images. This separation ensures that the model makes decisions based on content without being influenced by style differences.

Thirdly, algorithm-level approaches modify the learning process to enhance the model's generalization capability

across domains. Meta-Learning for Domain Generalization (MLDG)[10] is a meta-learning method designed specifically for domain generalization, using data from multiple domains for meta-training to ensure the model performs well on new, unseen domains. Additionally, the increasingly recognized TTA also falls into this category.

Test-time adaptation. In recent years, the field of domain generalization has seen TTA emerge as a significant breakthrough. The primary goal of TTA is to allow models to adapt in real-time during testing, thereby maintaining consistent performance on new domains that were not encountered during training. Closely related to TTA is another approach, Unsupervised Domain Adaptation (UDA)[15][3], which trains models using labeled data from the source domain and unlabeled data from the target domain. Unlike UDA, TTA is characterized by its real-time adaptability and the fact that it fundamentally does not require access to source data during model adaptation. TTT is a quintessential method in TTA, involving multi-task learning with target tasks and self-supervised tasks such as rotation prediction using training images. Subsequently, for the test data, it performs self-supervised learning without accessing the source data, enabling the feature extractor to bridge the gap between the domains of training and test data. While TTT has shown impressive results, it also encounters significant challenges. Firstly, multi-task learning using common self-supervised tasks like rotation prediction does not directly contribute to domain generalization. Secondly, the choice of self-supervised task is critical, and selecting an inappropriate task can degrade model performance. The selection of self-supervised tasks relies on intuition rather than quantitative criteria, posing a significant challenge for TTA that requires real-time adaptation. Contrarily, TENT employs unsupervised learning instead of self-supervised tasks to adapt the model to test data. Diverging from TTT's multi-task learning approach during the training phase, TENT concentrates exclusively on the primary task. When faced with test data, it utilizes unsupervised learning, specifically entropy minimization, to fine-tune the model's batch normalization layers, thereby adapting it to the new data. The simplicity of TENT, necessitating merely the application of entropy minimization to test data, offers an uncomplicated yet effective method. However, its performance can be inferior to TTT in certain domains. Alongside TTT and TENT, SHOT is another high-accuracy TTA method. SHOT uses pseudo-labeling for TTA, adapting the feature extractor to test data through cross-entropy loss with assigned pseudo-labels. Nevertheless, SHOT faces challenges when pseudo-labeling fails, potentially leading to decreased classification accuracy.

3. Proposed Method

We propose MCTTA as a simple yet potent solution to the challenges outlined in Section 2. As a solution to the primary challenge, MCTTA employs MCL, a unique contrastive learning approach specifically designed for domain generalization. Unlike conventional contrastive learning approaches that rely on color transformations and random cropping, MCL uses MixStyle for data augmentation in the feature space. This enables the feature extractors to identify consistent features across various domains, making MCL more specialized in domain generalization compared to conventional contrastive learning. In addition, MCL distinguishes itself from conventional contrastive learning by utilizing two feature extractors with distinct initial values but identical architecture. The dual extractor approach not only serves MCL's purposes but also contributes to improved classification performance through ensemble classification. For the second challenge, we demonstrate in Section 4 through evaluation experiments that MCTTA can be universally applied across various domains. It is noteworthy that, despite incorporating all three approaches to domain generalization described in Section 2, MCTTA remains a remarkably simple method. It incorporates MixStyle for the data-level approach, ensemble classification with two feature extractors for the model-level approach, and adaptation to the test domain via MCL for the algorithm-level approach.

Section 3.1 explains the details of Training phase, employing multi-task learning that combines ensemble classification and MCL using training data. Section 3.2 focuses on the TTA phase of our method, wherein the feature extractor is adapted to the domain of the test data through MCL.

3.1. Training Phase

The goal of the Training phase in the MCTTA approach is to develop a feature extractor that can identify consistent features across various domains through a multi-task learning framework combining ensemble classification and MCL. The comprehensive view of the Training phase is illustrated in Figure 1. Training data are defined as x, without preprocessing. Each x is fed into two feature extractors with the same structure but different initial values, producing two types of outputs. As depicted in Figure 2, one output comes from straightforward feature extraction, while the other is obtained by applying MixStyle which mixes domain information across batches. In each learning step, the layer for MixStyle application is randomly selected from those nearer to the input than the feature extractor's intermediate convolutions. We focus on shallower layers because domain information is primarily represented in these initial layers of the CNN[22]. The choice of random selection aims to enrich the variation in the mixing of domain information. All features are utilized in both ensemble classification and

MCL. Ensemble classification enhances the stability and accuracy of classification. When our method is combined with MixStyle, it further promotes domain generalization compared to conventional classification methods. In MCL, a loss based on cosine similarity is calculated between features that have undergone MixStyle, with domain information mixed between batches, and those that have not. By maximizing the similarity between features, the feature extractor is enabled to consistently extract features across various domains, thereby advancing domain generalization.

The ensemble classification is formulated by employing cross-entropy loss. Probability distributions obtained from the classifier are denoted as $\mathcal{P}mix$ and $\mathcal{P}ori$ for the features with and without the application of MixStyle, respectively. $\mathcal{P}'_{mix}$ and $\mathcal{P}'_{ori}$ represent these distributions from the other classifier. y indicates the ground truth label and CE stands for cross-entropy loss.

$$Pre = \frac{\mathcal{P}_{ori} + \mathcal{P}'_{ori}}{2} \tag{1}$$

$$Pre' = \frac{(\mathcal{P}_{mix} + \mathcal{P}'_{mix})}{2} \tag{2}$$

$$\mathcal{L}_{ce} = CE(Pre, y) + CE(Pre', y) \tag{3}$$

Next, we introduce the formulation of MCL, where $\mathcal{Z}mix$ and $\mathcal{Z}ori$ denote the features extracted with and without the application of MixStyle, respectively, from one of the feature extractors. Similarly, $\mathcal{Z}'_{mix}$ and $\mathcal{Z}'_{ori}$ are the features from the other extractor, processed in the same manner. In addition, $\cdot$ and $\|\|$ represent the dot product and vector norm, respectively.

$$\mathcal{L}_{cos} = 2 - \left(\frac{\mathcal{Z}_{\mathbf{ori}} \cdot \mathcal{Z}'_{\mathbf{mix}}}{\|\mathcal{Z}_{\mathbf{ori}}\|\|\mathcal{Z}'_{\mathbf{mix}}\|} + \frac{\mathcal{Z}'_{\mathbf{ori}} \cdot \mathcal{Z}_{\mathbf{mix}}}{\|\mathcal{Z}'_{\mathbf{ori}}\|\|\mathcal{Z}_{\mathbf{mix}}\|}\right) \tag{4}$$

The final loss in the Training phase is defined as

$$\mathcal{L}_{train} = \mathcal{L}_{ce} + \alpha\mathcal{L}_{cos} \tag{5}$$

where α is a hyperparameter. During the Training phase, the objective is to minimize $\mathcal{L}_{train}$ by training two feature extractors and two classifiers.

3.2. TTA Phase

The goal of the TTA phase in MCTTA is to adapt the feature extractor to the test data using MCL. In TTT, the Training phase involves multi-task learning that combines common self-supervised tasks such as rotation prediction with classification. By performing self-supervised tasks during the TTA phase, the feature extractor can absorb differences between the domains of training data and test data. In MCTTA, we leverage MCL, a self-supervised task specifically designed for domain generalization. This approach

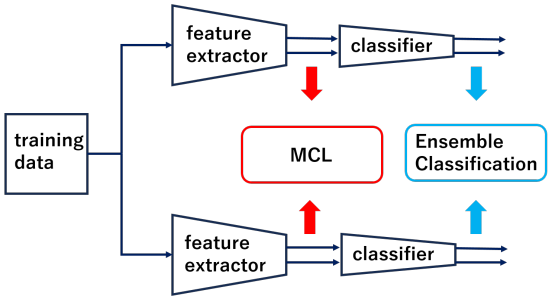

Figure 1. The comprehensive view of the Training phase. Training data is fed into two feature extractors and classifiers with identical structures but different initial values. Beyond serving as input for the classifiers, the output from the feature extractors is utilized in our proposed MCL. This enhances the extractors' ability to identify universal features across domains. The classifiers' output is employed for ensemble classification.

Figure 2. Internal Structure of the Feature Extractor. MixStyle is applied to random shallow layers of the CNN, and domain information is mixed across batches.

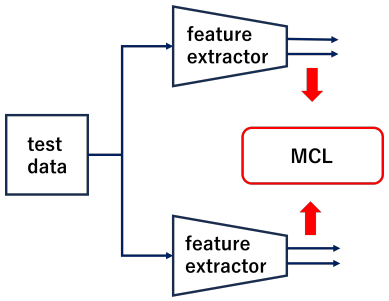

Figure 3. The comprehensive view of the TTA phase. Unlike the training phase, ensemble classification is not performed due to the absence of ground truth labels for test data.

allows us not only to absorb differences across domains using test data but also to significantly enhance the model's generalization capabilities to various domains. This phase is implemented online during inference on test data, with the updated parameters being retained for subsequent tests. For the test data, feature extraction is performed in the same manner as during the training phase. However, as the test data lack class labels, ensemble classification is not conducted, and only MCL is executed. It should be noted that ensemble classification is utilized for inference.

The final loss in the TTA phase is represented by Equa-

tion (6), utilizing Equation (4) for its calculation.

$$\mathcal{L}_{tta} = \mathcal{L}_{cos} \tag{6}$$

During the TTA phase, only two feature extractors are trained to minimize $\mathcal{L}_{tta}$. The learning rate is set to β times that of the Training phase, where β is a hyperparameter.

4. Experiment

4.1. Setting

Datasets. To ensure objectivity in our experiments, we employed the DomainBed benchmark library, selecting three datasets: PACS, Office-Home and Colored MNIST. **PACS** includes 9,991 images across seven classes and four domains (Photo, Art painting, Cartoon, Sketch), making it one of the most extensively used resources in the field of domain generalization. **Office-Home** comprises 15,588 images, 65 classes, and four domains (Art, Clipart, Product, Real world). **Colored MNIST**, a modification of MNIST[8] with color added to create three domains, contains 70,000 images and two classes.

Training and evaluation details. For our study, images were partitioned for each domain into an 8:2 ratio (train:validation), with the split dynamically selected by DomainBed for each trial. The model undergoes training in 5,000 iterations using all domains except the test domain, after which it is evaluated on the test domain. The model selection for testing employs a strategy within DomainBed known as training-domain validation set selection. This approach calculates the average accuracy on the validation data across all training domains at predetermined steps defined by DomainBed, selecting the model that achieves the highest average accuracy. In all experiments, the feature extractor is ResNet18[5] pre-trained on ImageNet[2]. Each trial is conducted 10 times with varied initial values, and the seed value, learning rate, and augmentation techniques are dynamically set following DomainBed, with a uniform batch size of 32. The unique hyperparameters of our proposed method, α and β (refer to Section 3.1 and Section 3.2), are set to $\alpha = 1$ and $\beta = 0.1$ across all experiments. Setting $\alpha = 1$ aims to equally leverage ensemble classification and MCL in the model's training, ensuring a balanced contribution from both methods. The choice of $\beta = 0.1$ is intended to prevent the model from overfitting to the test data. Continuous over-adaptation may compromise the feature extractor's ability to consistently extract features across various domains.

4.2. Results

In this section, we first present and discuss the detailed results obtained from the PACS, Office-Home, and Colored MNIST datasets. Subsequently, we provide a summary and discussion of the findings across these datasets. The values in the Tables represent the average accuracy over 10 trials, expressed in percentage terms. For instructions on how to read the table, see the detailed description in Table 1. In TTT's self-supervised task, rotation prediction is used.

PACS. Referencing Table 1, our method achieved an average accuracy of 84.6% on PACS, marking a 1.2% improvement over TTT. When our method is compared with TENT which adapts the model to the test domain through unsupervised learning, our method was superior by an average of 1.6%. Even against SHOT, a TTA technique using pseudo-labeling, our approach leads by an average of 1.0%. Furthermore, it outperformed conventional domain generalization methods such as Mixup, SagNet, and MLDG using meta-learing, which do not adapt the model to the test data, showing the superior domain generalization capability of our method over both other TTA techniques and conventional domain generalization methods. The rank score comparison further confirmed that our method is the most effective. It is also noteworthy that TTA methods consistently outperformed the baseline method, which relies on a simple approach using a ResNet18 pre-trained on ImageNet and cross-entropy loss, reaffirming the efficacy of TTA methods for domain adaptation.

Office-Home. As shown in Table 2, our proposed method achieved an average accuracy of 63.5% on the Office-Home dataset, marking a 2.9% improvement over TTT. When we compared with TENT, our method is superior by 0.9% on average and is on par with SHOT. When we compared our method with conventional domain generalization methods like Mixup, SagNet, and MLDG, our proposed method achieved the second highest accuracy following SelfReg. This demonstrates that our method is an effective learning approach with superior generalization across domains while falling just short of SelfReg. In terms of rank score, SelfReg leads, with our method tying for second place with CORAL, SD, and SHOT. When comparing TTT to the baseline, TTT shows lower accuracy in the test domains as follows: Art by 2.6%, Product by 1.8%, Real world by 0.3%, with an overall average reduction of 0.9%. TTT only surpasses the baseline in the Clipart domain. This phenomenon aligns with the discussion in Section 2, where a mismatch between the domain and the self-supervised task can lead to decreased accuracy with TTT.

Colored MNIST. As seen in Table 3, our proposed method achieved an average accuracy of 53.2% on the Colored MNIST dataset, outperforming TTT by 0.9%. When we compared to TENT, our method leads by an average of 1.3%, and it also surpasses SHOT by the same

Table 1. Comparison results on PACS. Accuracy shows the results from 10 trials for each method across four different test domains: Photo, Art painting, Cartoon, and Sketch. Each model was trained on the three domains other than the test domain. The "average" denotes the mean accuracy across these four domains, while "rank score" aggregates the ranking positions of each method. For instance, if a method ranked the 5th in the Photo domain, the 10th in Art painting, the 15th in Cartoon, and the 10th in Sketch, the rank score would be 40. In cases of tied ranks, the lower score is attributed to the methods. For example, if three methods tie for the 1st place in the Photo domain, each receives a score of 1, and the next highest accuracy method would receive a score of 4. TENT, SHOT, TTT, and our method fall under the category of TTA, which adapt to test data at the time of testing. In contrast, other methods mentioned do not perform adaptation on test data, following conventional domain generalization approaches. The baseline represents a simplistic learning approach that does not utilize any domain generalization method; it is derived from TENT by excluding the adaptation to test data.

Algorithm	**Photo**	**Art painting**	**Cartoon**	**Sketch**	**average**	**rank score↓**
ERM	94.8	81.6	72.3	73.9	80.6	51
Fish	94.9	82.6	79.3	70.7	81.9	34
GroupDRO	95.4	77.4	74.5	73.4	80.2	50
Mixup	93.2	84.0	73.1	67.0	79.3	57
MLDG	94.5	84.1	74.9	74.9	82.1	36
CORAL	94.8	81.7	78.4	73.7	82.1	38
MTL	94.1	73.2	76.2	76.5	80.0	50
SagNet	95.6	80.5	76.3	78.6	82.7	29
ARM	92.3	82.3	77.0	77.0	82.1	35
VREx	94.8	79.7	73.6	76.8	81.2	43
SD	96.2	80.7	74.7	76.5	82.0	32
ANDMask	93.5	75.8	73.2	65.5	77.0	71
SelfReg	94.6	82.1	74.3	76.0	81.7	43
TRM	94.9	79.6	77.2	75.8	81.9	38
baseline	94.5	79.6	74.4	68.9	80.2	60
TENT	96.6	81.8	80.9	72.9	83.0	29
SHOT	96.7	85.0	81.6	71.1	83.6	19
TTT	96.0	82.2	81.3	74.1	83.4	24
ours	95.8	83.5	82.1	76.8	84.6	13

Table 2. Comparison results on Office-Home. Accuracy shows the results from 10 trials for each method across four different test domains.

Algorithm	**Art**	**Clipart**	**Product**	**Real world**	**average**	**rank score↓**
ERM	48.7	45.1	69.0	69.1	58.0	62
Fish	52.4	48.4	66.5	70.6	59.5	46
GroupDRO	49.8	45.5	67.0	68.4	57.7	61
Mixup	53.2	49.9	70.4	71.7	61.3	32
MLDG	48.8	45.5	67.4	69.1	57.7	61
CORAL	55.5	50.3	72.3	73.6	62.9	14
MTL	48.9	47.8	66.5	68.0	57.8	61
SagNet	53.2	48.9	69.8	70.4	60.6	37
ARM	49.2	46.7	65.9	68.0	57.4	66
VREx	50.5	47.9	66.8	70.0	58.8	51
SD	55.7	50.8	71.0	74.4	63.0	14
ANDMask	47.3	47.7	65.9	67.4	57.1	69
SelfReg	56.6	50.2	73.3	74.0	63.5	11
TRM	51.8	47.3	69.2	69.6	59.5	49
baseline	54.4	47.7	71.1	73.0	61.5	31
TENT	55.0	50.3	71.7	73.2	62.6	19
SHOT	56.7	52.6	71.9	72.7	63.5	14
TTT	51.8	49.5	68.3	72.7	60.6	38
ours	53.6	54.1	72.9	73.4	63.5	14

margin. Although it is slightly behind ARM by an average of 0.2%, it is the best accuracy among the other domain generalization methods. Notably, our method ranked the highest when we evaluated with rank score. Additionally, when we compare TTT with the baseline, we observed a reduction in accuracy by 0.4% and 0.3% for test domains at +90% and +80% respectively, with an overall average decrease of 0.1%, indicating that TTT's self-supervised

Table 3. Comparison results on Colored MNIST. Accuracy illustrates the results from 10 trials for each method across three different test domains.

Algorithm	+90%	+80%	-90%	average	rank score↓
ERM	71.2	72.8	9.8	51.3	49
Fish	72.3	73.6	10.1	52.0	28
GroupDRO	72.4	72.1	10.3	51.6	31
Mixup	72.0	72.4	10.1	51.5	42
MLDG	73.1	72.7	10.6	52.1	20
CORAL	71.8	73.9	10.2	51.9	26
MTL	71.6	72.7	10.4	51.6	33
SagNet	71.6	73.4	10.5	51.9	26
ARM	78.9	71.0	10.3	53.4	15
VREx	73.2	73.2	10.1	52.2	24
SD	72.0	73.7	10.1	52.0	30
ANDMask	73.2	73.3	10.2	52.2	19
SelfReg	72.1	72.7	10.2	51.7	34
TRM	67.9	72.0	10.2	50.0	46
baseline	73.1	74.1	10.0	52.4	24
TENT	72.3	73.0	10.3	51.9	27
SHOT	72.4	73.1	10.1	51.9	31
TTT	72.7	73.8	10.4	52.3	14
ours	73.2	73.5	13.0	53.2	9

Table 4. Rank score over three datasets. Under the names of three datasets (PACS, Office-Home, Colored MNIST), we show the average accuracy for each test domain. "average" represents the mean value across the three datasets, while "rank score" is recalculated specifically for this table.

Algorithm	PACS	Office-Home	Colored MNIST	average	rank score↓
ERM	80.6	58.0	51.3	63.3	46
Fish	81.9	59.5	52.0	64.5	29
GroupDRO	80.2	57.7	51.6	63.2	46
Mixup	79.3	61.3	51.5	64.0	43
MLDG	82.1	57.7	52.1	64.0	29
CORAL	82.1	62.9	51.9	65.6	21
MTL	80.0	57.8	51.6	63.1	47
SagNet	82.7	60.6	51.9	65.1	24
ARM	82.1	57.4	53.4	64.3	25
VREx	81.2	58.8	52.2	64.1	31
SD	82.0	63.0	52.0	65.7	21
ANDMask	77.0	57.1	52.2	62.1	43
SelfReg	81.7	63.5	51.7	65.6	27
TRM	81.9	59.5	50.0	63.8	40
baseline	80.2	61.5	52.4	64.7	25
TENT	83.0	62.6	51.9	65.8	20
SHOT	83.6	63.5	51.9	66.3	13
TTT	83.4	60.6	52.3	65.4	16
ours	84.6	63.5	53.2	67.1	4

task is not effective for the Colored MNIST.

Summary of three Tables. Table 4 serves as a summary of Table 1, Table 2, and Table 3 which show the average accuracy for each test domain under the names of three datasets (PACS, Office-Home, Colored MNIST). The "average" indicates the mean accuracy across these datasets, while "rank score" has been recalculated specifically for this Table. The proposed method shows a significant improvement in accuracy compared to TTT, with increases of 1.2% for PACS, 2.9% for Office-Home, 0.9% for Colored MNIST, and an overall 1.7% for average, demonstrating effective learning across various domains. When our method is compared with TENT, the proposed method is superior with improvements of 1.6% for PACS, 0.9% for Office-Home, 1.3% for Colored MNIST, and

Table 5. The effectiveness of MCL. To confirm the effectiveness of MCL, we compare the accuracy of conventional contrastive learning, which performs random color transformations on an image-by-image basis (referred to as "Color"), with MCL that applies MixStyle at the feature level (referred to as "MixStyle"). Note that all other algorithms employed are identical to the proposed method.

Color	MixStyle	PACS	Office-Home	Colored MNIST	average
✓	-	74.5	59.6	54.9	63.0
-	✓	84.6	63.5	53.2	67.1

Table 6. Effectiveness of two models. To investigate the effectiveness of using two models with the same structure but different initial values, we compare the case of performing MCL with a single model without ensemble classification to the case of conducting MCL and ensemble classification with two models.

Two models	PACS	Office-Home	Colored MNIST	average
-	83.4	62.7	52.9	66.3
✓	84.6	63.5	53.2	67.1

1.3% for average. Although TENT outperformed TTT by 0.4% on average, it falls short by 0.4% on PACS and Colored MNIST, indicating that it may not be more effective across all domains as suggested in Section 2. SHOT using pseudo-labeling achieved an average of 66.3%, surpassing TTT by 0.9%, yet our proposed method exceeded SHOT by an additional 0.8% on average. Furthermore, our method is compared with conventional domain generalization methods like Mixup, SagNet, and MLDG using meta-learning which do not adapt the model at test time as discussed in Section 2, and MCTTA achieved the highest mean accuracy. The proposed method also proved to be the most superior when we compared in the rank score scenario, outperforming TTA and conventional domain generalization techniques.

4.3. Ablation Study

Table 5 contrasts the accuracy between conventional contrastive learning, which applies random color transformations individually to each image, and MCL, which integrates MixStyle at the feature level, to evaluate the efficacy of MCL. All other algorithms are identical to the proposed method. MCL using MixStyle has shown a 4.1% improvement in accuracy over random color transformations in the average, and this result indicates MCL's significant effectiveness. However, for the Colored MNIST, created by adding colors to MNIST to form different domains, MCL using MixStyle resulted in a 1.7% lower performance. This inferior performance in Colord MNIST is attributed to the exceptional compatibility of conventional contrastive learning with Colored MNIST, as it is designed to learn and extract consistent features across domains with varying colors.

To explore the effectiveness of utilizing two models with identical structures but distinct initial values, Table 6 compares the scenario of executing MCL with a single model, excluding ensemble classification, with the scenario where MCL and ensemble classification are implemented using two models. The usage of two models has resulted in improved accuracy: 1.2% in PACS, 0.8% in Office-Home, 0.3% in Colored MNIST, and an overall increase of 0.8% in average, and the result shows the effectivness of this approach.

5. Conclusion

In this paper, we introduced MCTTA as a notably simple yet effective solution to the challenges of TTT. MCTTA leverages MCL, our novel contrastive learning approach which tailored specifically for domain generalization. Unlike conventional contrastive learning which commonly employs color transformations and random cropping, MCL innovates by utilizing MixStyle in feature space for data augmentation. This enables to consistently extract features across various domains. MCL utilizes two feature extractors with varying initial settings yet the same structure, setting it apart from standard contrastive learning. The use of two extractors not only serves MCL's purposes but also enhances classification accuracy via ensemble classification. Experiments on the DomainBed benchmark library and three datasets demonstrated that MCTTA achieved an average accuracy improvement of 1.7% over TTT, outperforming other TTA methods and conventional domain generalization techniques. Additionally, the effectiveness of employing MCL and two models was demonstrated. The reduction of computational overhead due to the usage of two models is a subject for future works.

Acknowledgements

This research is partially supported by JSPS KAKENHI Grant Number 24K15020.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 2

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 5

[3] Yuqi Fang, Pew-Thian Yap, Weili Lin, Hongtu Zhu, and Mingxia Liu. Source-free unsupervised domain adaptation: A survey. *Neural Networks*, page 106230, 2024. 3

[4] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020. 2

[5] 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. 5

[6] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1): 2, 2020. 2

[7] Pranjal Kumar, Piyush Rawat, and Siddhartha Chauhan. Contrastive self-supervised learning: review, progress, challenges and future research directions. *International Journal of Multimedia Information Retrieval*, 11(4):461–488, 2022. 2

[8] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[9] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 2

[10] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 1, 3

[11] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou. A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 2021. 2

[12] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020. 2

[13] Jian Liang, Ran He, and Tieniu Tan. A comprehensive survey on test-time adaptation under distribution shifts. *arXiv preprint arXiv:2303.15361*, 2023. 1

[14] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8690–8699, 2021. 2

[15] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 3

[16] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020. 1

[17] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017. 2

[18] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *ICLR 2021 Spotlight*, 2020. 2

[19] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022. 1

[20] Zixin Wang, Yadan Luo, Liang Zheng, Zhuoxiao Chen, Sen Wang, and Zi Huang. In search of lost online test-time adaptation: A survey. *arXiv preprint arXiv:2310.20199*, 2023. 1

[21] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ICLR 2018*, 2017. 2

[22] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *ICLR 2021*, 2021. 2, 3

[23] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 1