Foresee What You Will Learn:
Data Augmentation for Domain Generalization in Non-stationary Environment

Qiuhao Zeng[1], Wei Wang[1], Fan Zhou[2], Charles Ling[1], Boyu Wang[1*]

[1]University of Western Ontario, Canada
[2] Beihang University, China

Abstract

Existing domain generalization aims to learn a generalizable model to perform well even on unseen domains. For many real-world machine learning applications, the data distribution often shifts gradually along domain indices. For example, a self-driving car with a vision system drives from dawn to dusk, with the sky darkening gradually. Therefore, the system must be able to adapt to changes in ambient illumination and continue to drive safely on the road. In this paper, we formulate such problems as Evolving Domain Generalization, where a model aims to generalize well on a target domain by discovering and leveraging the evolving pattern of the environment. We then propose Directional Domain Augmentation (DDA), which simulates the unseen target features by mapping source data as augmentations through a *domain transformer*. Specifically, we formulate DDA as a bi-level optimization problem and solve it through a novel meta-learning approach in the representation space. We evaluate the proposed method on both synthetic datasets and real-world datasets, and empirical results show that our approach can outperform other existing methods.

1 Introduction

One common assumption in conventional machine learning methods is that the training and test data are sampled from the same distribution. However, in many real-world problems, this assumption does not hold, and the data distribution can shift in changing environments. Consequently, a model learned from training data often fails to generalize well on the data sampled from a shifting distribution, especially when the target data is not accessible. To address the problem of domain shift, *domain generalization* (DG) is proposed to train a model with source domains that can generalize to unseen target domains.

Most existing DG methods aim to extract domain-invariant features by either statistical distance minimization (Muandet, Balduzzi, and Schölkopf 2013; Albuquerque et al. 2019; Shui, Wang, and Gagné 2022; Zhou et al. 2021a) or adversarial learning (Li et al. 2018b; Volpi et al. 2018; Zhou et al. 2020a), implicitly assuming that all the domains are independently sampled from a static en-
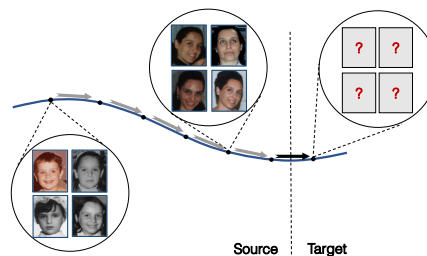
Figure 1: The data distribution shifts along a smoothing curve in a low-dimensional manifold: The appearance of a female changes over years. We capture the data evolving direction (grey arrow) between observed source domains and predict the evolving direction (black arrow) towards the unseen domain beyond the last observed distribution.

vironment (Muandet, Balduzzi, and Schölkopf 2013; Arjovsky et al. 2019a; Sagawa et al. 2019). These methods may collapse when the learning tasks are collected from a non-stationary environment. For example, environmental changes due to illumination, seasons, or weather conditions can pose significant challenges for an outdoor robot equipped with vision systems (Wulfmeier, Bewley, and Posner 2018; Hoffman, Darrell, and Saenko 2014; Lampert 2015). Since the evolving patterns are not taken into account, the existing popular DG methods are not able to handle such problems properly (Muandet, Balduzzi, and Schölkopf 2013; Li et al. 2018b; Arjovsky et al. 2019b).

To alleviate the aforementioned issues, one can take advantage of data shift patterns in non-stationary environments. For example, when deploying a face recognition system to search for a missing person, the system may be built only on photos from childhood to adolescence that were taken decades ago. In such a scenario, the system can benefit from modelling the domain shift along the age to predict a person's current appearance (Fig. 1). In this work, we address this problem under the *evolving domain generalization* (EDG) scenario (Nasery et al. 2021; Qin, Wang, and Li 2022), where the source domains are sampled from a changing environment, and the objective is to train a model that generalizes well on an unseen target domain by capturing and leveraging the evolving pattern of the environment.
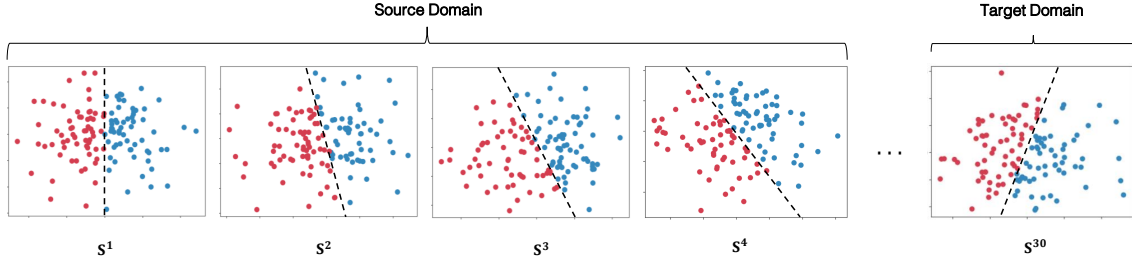
Figure 2: The Rotated Gaussian datasets (dashed lines are the ground truth of decision boundaries)

To this end, we propose *directional domain augmentation* (**DDA**) for EDG. First, we generate augmented features along the direction of the domain shift, such that the augmentations can mimic the next unobserved target domain feature. To achieve this, we design an attention-based *domain transformer* to capture the evolving pattern between consecutive domains by leveraging the power of the attention mechanism to capture the temporal pattern (Girdhar and Grauman 2021; Vaswani et al. 2017) for predicting the feature of future unseen domains. Furthermore, we demonstrate in an illustrative example (Sec. 2.2) that the training process can be formulated as a bi-level programming problem that allows us to effectively capture and leverage the domain shift patterns via a meta-learning scheme.

To summarize, the contribution of our work is trifold:

1. Our work provides a framework to mitigate the impact of lacking data from the target domain under non-stationary environments with an attention-based domain transformer. We show that the optimal domain transformer can generate augmentations whose decision boundaries are aligned with target data without any explicit distribution divergence loss.

2. We formulate the training process as a bi-level optimization problem with meta-learning. We also demonstrate that the meta-parameter of the shared classifier could be effectively adapted to the unseen target domain. Our analysis then leads to a novel algorithm, namely *directional domain augmentation* (**DDA**), for the EDG problems, which can capture the evolving patterns of domains and predict the future feature effectively.

3. We evaluate the algorithm with both synthetic and real-world datasets showing that DDA improves the performance over the state-of-the-art DG algorithms for the EDG problem.

2 Preliminary

2.1 Problem Setup

Let $\mathcal{D}_t$ be the probability distribution that characterizes t-th domain in Evolving Domain Generalization (EDG), and $S_t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$ is a set of n_t instances drawn from $\mathcal{D}_t$, where $x_i^t \in \mathcal{X}$ is the i-th data point in the t-th domain, and $y_i^t \in \mathcal{Y}$ is its label. For every instance, we encode it with a feature extractor $\phi : \mathcal{X} \rightarrow \mathcal{Z}$, and we obtain the embedded instance $z_i^t \in \mathcal{Z}$ by $z_i^t = \phi(x_i^t)$. The goal of EDG is to learn a robust and generalizable model from T source domains by

capturing and leveraging the evolving pattern so that it can perform well on the unseen target domain $\mathcal{D}_{T+1}$.

To this end, we propose a generative approach to EDG which simulates the features for the target domain $\mathcal{D}_{T+1}$ by learning a domain transformer $\psi : \mathcal{Z} \rightarrow \mathcal{Z}$. Intuitively, given the data set of the t-th domain S_t, if ψ can properly capture the evolving pattern, a predictive model $\tilde{h}_{t+1}^*$ trained on the simulated data set $\tilde{S}_{t+1} = \{(\tilde{z}_i^{t+1}, y_i^t)\}_{i=1}^{n_t}$, where $\tilde{z}_i^{t+1} = \psi(z_i^t)$, should perform well on S_{t+1}. Likewise, a model h_{t+1}^* trained on the real data set S_{t+1} should also performs well on $\tilde{S}_{t+1}$. Note that we implicitly assume that the evolving pattern is consistent across all consecutive domains (*i.e.*, $\forall t$, ψ can map instances from t to $t + 1$-th domain), which is reasonable in real-world applications. Otherwise, it is impossible to capture the evolving pattern if the environment varies arbitrarily (*e.g.*, it is extremely challenging to predict the stock market tomorrow).

2.2 An Illustrative Example

As an illustrative example, we consider the rotated Gaussian data sets as shown in Figure 2, where the instances for binary classification are generated by a d-dimensional Gaussian distribution with a mean of 0, and the decision boundary for each next domain is rotated by 12 degrees counterclockwise. In this example, the domain transformer ψ can be characterized by a rotation matrix: $\psi \in \mathbb{R}^{d \times d}$, and the transform process from t to $t + 1$ is simply given by $\tilde{x}_i^{t+1} = \psi x_i^t$ (here we apply an identity mapping as a featurizer ϕ, so $z_i^t = x_i^t$).

As analyzed in Section 2.1, the decision boundaries trained on S_{t+1} and $\tilde{S}_{t+1}$ should be well aligned if ψ can capture the rotation pattern. Thus, we aim to learn ψ in a way such that, for any two consecutive domains, a linear classifier $h_{t+1}^* \in \mathbb{R}^d$ trained on S_{t+1} also performs well on $\tilde{S}_{t+1}$, leading to the following bi-level optimization problem:

$$\min_{\psi} \ \|Y^t - \tilde{X}^{t+1} h_{t+1}^*\|_2^2 \tag{1}$$

$$\text{s.t.} \quad h_{t+1}^* = \operatorname*{argmin}_{h} \|Y^{t+1} - X^{t+1} h\|_2^2 \tag{2}$$

$$\forall t \in \{1, \cdots, T-1\}$$

where $Y^t = [y_1^t, \ldots, y_{n_t}^t]^\mathsf{T}$, $\tilde{X}^{t+1} = [\tilde{x}_1^{t+1}, \ldots, \tilde{x}_{n_t}^{t+1}]^\mathsf{T}$, $X^{t+1} = [x_1^{t+1}, \ldots, x_{n_t}^{t+1}]^\mathsf{T}$, and T is the transpose operator. Here, we adopt the squared loss for classification in order
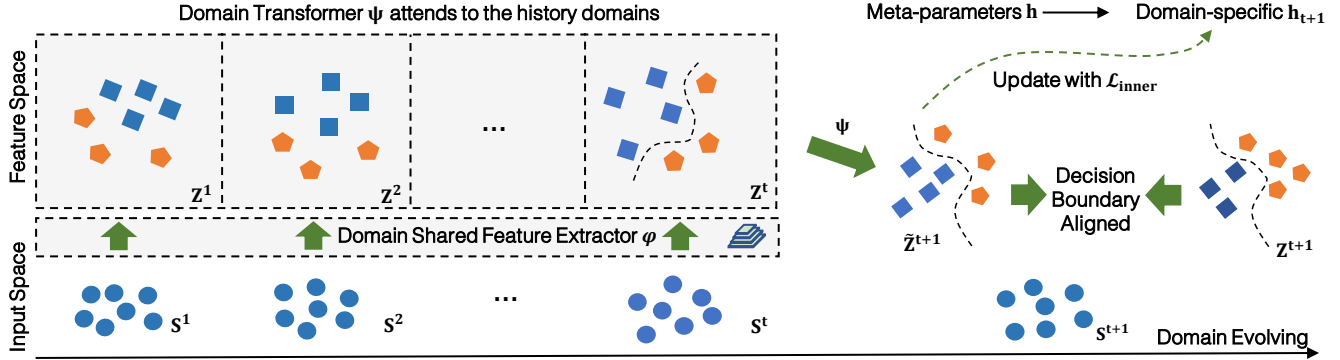
Figure 3: Illustration of the proposed Directional Domain-Augmentation (DDA) model. The domain transformer generates the augmented features in the direction of the unseen domain. With the bi-level optimization, the decision boundary of the augmented embeddings and the embeddings from the unseen domain gets aligned. The meta-parameters are then updated with the softened version classification loss on the augmented features.

ERM	CIDA	EAML	LSSAE	LDDA
59.2 ± 1.1	50.5 ± 1.5	61.0 ± 2.8	88.4 ± 0.8	94.6 ± 0

Table 1: Experiment Results (accuracy %) on Synthetic Rotated Gaussian dataset

to obtain an analytical solution, which also corresponds to Fisher's linear discriminant (Bishop 2006).

Setting the derivative with respect to h_{t+1} to zero for Eqn. 2, we obtain the optimal solution $h_{t+1}^* = [(X^{t+1})^\mathsf{T} X^{t+1}]^{-1}(X^{t+1})^\mathsf{T} Y^{t+1}$. Similarly, setting the derivative with respect to ψ to zero for Eqn. 1, we obtain the following equation for ψ (detailed derivation can be found in Section A of the supplementary materials):

$$A^t \text{vec}(\psi) = B^t, \qquad (3)$$

where function $\text{vec}(\cdot)$ is the vectorization operator, $A^t = \left(h_{t+1}^*(h_{t+1}^*)^\mathsf{T}\right) \otimes \left(X^{t^\mathsf{T}} X^t\right)$, $B^t = \text{vec}(X^{t^\mathsf{T}} Y^t h_{t+1}^{*\mathsf{T}})$, and $\otimes$ is the Kronecker product operator.

As the rotation is consistent across all domains, Eqn. 3 holds for any $t = 1, \ldots, T-1$, which gives

$$[(A^1)^\mathsf{T}, \ldots, (A^{T-1})^\mathsf{T}]^\mathsf{T} \text{vec}(\psi) = [(B^1)^\mathsf{T}, \ldots, (B^{T-1})^\mathsf{T}]^\mathsf{T}. \qquad (4)$$

As $\text{rank}(A^t) = d$, Eqn. 4 will be well-posed when $d \leq T-1$. Then, for the rotated Gaussian shown in Fig. 2, the domain transformer ψ^* learned by solving Eqn. 4 is given by

$$\psi^* = \begin{bmatrix} 0.9824 & 0.2031 \\ -0.2107 & 0.9720 \end{bmatrix} \approx \begin{bmatrix} \cos 12° & \sin 12° \\ -\sin 12° & \cos 12° \end{bmatrix},$$

which is very close to the ground-truth rotation matrix.

We denote this method as the linear DDA (LDDA), and its classification accuracy on the unseen target domain is shown in Table 1 (see Section 5.1 for the details of other baseline algorithms), from which it can be observed that LDDA achieves the best performance.

3 Method

In the rotated Gaussian example, LDDA aims to learn a domain transformer ψ to capture the evolving patterns of the environment, which motivates an effective solution to the EDG problem by designing a bi-level optimization problem. Note that ψ of LDDA in the illustrative example is assumed as a linear mapping in order to obtain an analytical solution and is only applicable to low-dimensional cases (*i.e.*, solving Eqn.4 requires $d \leq T-1$). We will illustrate our DDA framework in this section which extends to the general cases, including the non-linear cases.

3.1 Method Overview

In this section, we extend the proposed method to deep models by learning the domain transformer in the representation space and solving the bi-level optimization problem through a novel meta-learning scheme. Specifically, the proposed model consists of three components: a feature extractor ϕ, a domain transformer ψ, and a classification model h, which are parameterized by θ_ϕ, θ_ψ, and θ_h, respectively.

DDA's overall design is illustrated in Fig. 3. The sampled inputs first get projected into feature space by ϕ. ψ generates the augmentations to mimic the data from the next unseen domain by leveraging the evolving pattern. h takes fast adaptations on augmentations and the optimized h will perform well in the classification task of the next domain.

3.2 Domain Transformer

To capture the evolving patterns, the domain transformer ψ is designed to generate augmentations by transforming the features from historical domains into the next domain. In Section 2, only one preceding domain is utilized to simulate the next domain in a point-wise manner (*i.e.*, ψ only maps z_i^t to $\tilde{z}_i^{t+1}$). In order to take full advantage of the source data, we design ψ with an attention module, taking its strength to extract sequential information (Vaswani et al. 2017; Zeng, Fu, and Chao 2020), which allows the information to propagate over the evolving domains. Consequently, it can lever-

age the data from all historical source domains to effectively capture evolving patterns and simulate the target data.

Specifically, we first calculate the similarity score $s_{i,j}^{t,t'}$ between i-th sample from domain t and j-th sample from a historical domain t',

$$s_{i,j}^{t,t'} = \frac{\psi_q(z_i^t)\psi_k(z_j^{t'})}{\sqrt{d}} \quad (5)$$

where $1 \leq i,j \leq B$, $1 \leq t' \leq t \leq T$, B is the batch size, $\psi_q(\cdot)$ and $\psi_k(\cdot)$ denote the transformation that maps input features into the query and key embeddings, and $z_j^{t'}$ is a sampled feature from the historical domains in a batch. The similarity score $s_{i,j}^{t,t'}$ is normalized by the dimension of each transformed embedding to avoid small gradients caused by subsequent *softmax* functions (Vaswani et al. 2017). It measures how close the current sample is to the historical domain samples. Therefore, ψ will attend to the most relevant samples by reviewing the entire domain evolving history. Then, the directional transform augmentations $\tilde{z}_i^{t+1}$ can be obtained by a weighted sum of embedded values in the history domain plus the output of a skip-connection network:

$$\tilde{z}_i^{t+1} = \sum_{t'=1}^{t}\sum_{j=1}^{B} \frac{\exp s_{i,j}^{t,t'}}{\sum_{t''=1}^{t}\sum_{k=1}^{B}\exp s_{i,k}^{t,t''}}\psi_v(z_j^{t'}) + \psi_{sc}(z_i^t) \quad (6)$$

where $1 \leq i,j,k \leq B$, $1 \leq t',t'' \leq t \leq T$, $\psi_v(\cdot)$ denotes the transform that maps input embeddings into value embeddings. ψ_{sc} is a skip-connection network to help stabilize the learning (Zhang et al. 2018). In our case, it also helps to preserve the instance-level information. Thus, the domain transformer $\psi = \{\psi_k, \psi_q, \psi_v, \psi_{sc}\}$. We name augmentation $\tilde{z}_i^{t+1}$ *directional transform augmentation*, as it is generated according to the domain-evolving direction and transformed from the samples of the historical domains. It is noted that a sample input z_i^t is taken as a query, and the samples from history domains are taken as keys and values. The domain transformer aggregates information across domains (Xu et al. 2021) and generates $\tilde{z}_i^{t+1}$ in a way such that its decision boundary can also correctly classify the features z_i^{t+1} from the next domain.

3.3 Bi-level Optimization with Meta-learning

As shown in Eqn. 1 and Eqn. 2, we aim to optimize ψ through a bi-level optimization scheme so that $\tilde{S}^{t+1}$ and S^{t+1} can share the same predictive model h_{t+1}^*. One issue with this scheme is that each h_{t+1}^* in the inner loop (*i.e.,* Eqn. 2) is only learned from a single domain, which may lead to the overfitting problem. In order to take advantage of the transferred knowledge from all the other source domains, instead of learning h_{t+1}^* for each domain individually, we learn a good initialization θ_h that is shared across all the domains, and each domain-specific classifier, parameterized by θ_{h_t}, can be learned from S_t via *fast adaptations* (Finn, Abbeel, and Levine 2017).

Therefore, learning θ_ψ and θ_h can be seamlessly integrated into a single bi-level optimization problem that can

---

Algorithm 1: Directional Domain Augmentation

Input: *The feature extractor ϕ, the domain-shared classifier h, the domain transformer ψ, the learning rate of the inner loop α, the learning rate of the outer loop β and the batch size B.*

Initialize θ ($\theta = \{\theta_h, \theta_\psi, \theta_\phi\}$)

for *sampled mini-batch* $\{\{x_i^t, y_i^t\}_{i=1}^B\}_{t=1}^T$ **do**

  Calculate instance features $\{Z^t\}_{t=1}^T$;

  $\mathcal{L} = 0$;   // Initialize the loss for this episode

  for $t = 1$ *to* $T-1$ **do**

    for every sample i in each domain t:

      $\tilde{z}_i^{t+1} = \psi(z_i^t | \{Z^{t'}\}_{t'=1}^{t-1})$ in Eqn. 6;

    for *k=1 to num. of inner-loop steps* **do**

      Calculate the inner loss $\mathcal{L}_{inner}$ in Eqn. 7

      $\theta_{h_{t+1}} = \theta_h - \alpha\nabla_{\theta_h}\mathcal{L}_{inner}$

    Calculate the outer loss $\mathcal{L}_{outer}$ in Eqn. 9

    $\mathcal{L} = \mathcal{L} + \mathcal{L}_{outer}$

  Update $\theta \leftarrow \theta - \beta\nabla_\theta\mathcal{L}$;

return *trained model parameters θ*

---

be solved by meta-learning, resulting in more effective use of data. Specifically, we apply the episodic training scheme in (Finn, Abbeel, and Levine 2017), which consists of two steps: inner-loop updates and outer-loop updates. The training protocol is shown in Algorithm 1.

In each episode, we sample B data points of each domain from domain 1 to domain T, yielding $\{\{x_i^t, y_i^t\}_{i=1}^B\}_{t=1}^T$. Let $Z^t = \{z_i^t\}_{i=1}^B$ and $\tilde{Z}^{t+1} = \{\tilde{z}_i^{t+1}\}_{i=1}^B$, respectively, be features of sampled batch instances from the t-th domain and its directional transform augmentations. We randomly select two consecutive domains, domain t and domain $t+1$. Then, θ_h is learned with the loss $\mathcal{L}_{inner}$ on $\tilde{Z}^{t+1}$ in the inner loop:

$$\mathcal{L}_{inner}(S_t; \theta_h, \theta_\psi, \theta_\phi) = \frac{1}{B}\sum_{i=1}^{B}[\lambda \cdot \mathcal{L}_{cls}(y_t^i, h(\tilde{z}_{t+1}^i))$$

$$+ (1-\lambda)\mathcal{D}_{KL}(\sigma(h_t(z_t^i))/\tau_{temp}||\sigma(h(\tilde{z}_{t+1}^i))/\tau_{temp})] \quad (7)$$

where $\mathcal{L}_{cls}$ is the cross-entropy loss, σ is the softmax function, and λ is a trade-off parameter. $\mathcal{D}_{KL}(\cdot||\cdot)$ is the Kullback-Leibler (KL) divergence, which is adopted as a distillation loss (Hinton, Vinyals, and Dean 2015). It can be regarded as a softened softmax at a temperature τ_{temp} and able to reserve the instance semantics. Then, the domain-specific classifier $\theta_{h_{t+1}}$ is given by

$$\theta_{h_{t+1}} = \theta_h - \alpha\nabla_{\theta_h}\mathcal{L}_{inner}(S_t; \theta_h, \theta_\psi, \theta_\phi) \quad (8)$$

where α is the inner-loop learning rate. $\theta_{h_{t+1}}$ is the classifier optimized with $\tilde{Z}^{t+1}$, which is shared with the target domain's instances. Therefore, in the outer loop of each episode, h_{t+1} is evaluated on Z^{t+1}, and the corresponding loss function $\mathcal{L}_{outer}$ is given by

$$\mathcal{L}_{outer}(S_t, S_{t+1}; \theta_{h_{t+1}}, \theta_\psi, \theta_\phi) \quad (9)$$

$$= \frac{1}{B}\sum_{i=1}^{B}\mathcal{L}_{cls}(y_{t+1}^i, h_{t+1}(z_{t+1}^i|\theta_{h_{t+1}}))$$

Then, overall DDA parameters is updated by

$$\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}_{\text{outer}}, \qquad \theta = \{\theta_h, \theta_\psi, \theta_\phi\} \qquad (10)$$

where β is the outer-loop learning rate.

In the inference stage, we first simulate a set of the feature augmentations of size N: $\tilde{Z}^{T+1} = \{\tilde{z}_i^{T+1}\}_{i=1}^N$ from historical source domain features, and obtain the parameter $\theta_{h_{T+1}}$ for the target classifier on $\tilde{Z}^{T+1}$ via fast adaptation from θ_h.

4 Related Work

Domain Generalization (DG) Distribution matching is one predominant approach in domain generalization, where domain-invariant representation learning (Ganin et al. 2016) is intuitive and has been extensively studied. (Mancini et al. 2018) proposed to ensemble a unified model with generalization capability. Meta-learning has also been investigated for generalization (Li et al. 2018a, 2020; Balaji, Sankaranarayanan, and Chellappa 2018; Li et al. 2019). It is common sense that existing DG methods can not handle extrapolation well (Gulrajani and Lopez-Paz 2020; Nguyen et al. 2021), which makes it not suitable for our problem setup. In the early stage, the researchers mainly focused on aligning feature marginal distributions(Ganin et al. 2016), which has been proved not enough in case there exists concept-shift across domains. As a consequence, many recent works proposed to align the joint distributions(Li et al. 2021; Nguyen et al. 2021). However, joint distribution alignment is much harder than marginal distribution alignment, which results in the introduction of varieties of techniques such as information theory(Li et al. 2021).

Data augmentation Conventional data augmentation operations include cropping, flipping, rotation, scaling, and nosing. Data augmentation has been applied to improve the generalization capability of DG models. Besides conventional data augmentation methods, there are also a large number of generating-based methods (Rahman et al. 2019; Zhang et al. 2017) trying to generate all new instances. For example, (Zhou et al. 2020a) trains a transformation network for data augmentation.

Recent approaches (Volpi and Murino 2019; Shi et al. 2020; Zhou et al. 2020b, 2021b) have studied the data augmentation methods in DG by generating either augmented samples or intermediate embeddings to improve the generalization performance on the unseen domains. However, the domain shift patterns are absent in these kinds of approaches making the methods lack the ability to learn the non-stationary evolving patterns.

Evolving Domain Adaptation (EDA) / Evolving Domain Generalization (EDG) Several existing works have formulated a similar scenario as *evolving domain adaptation* (Hoffman, Darrell, and Saenko 2014; Lampert 2015; Wang, He, and Katabi 2020; Wulfmeier, Bewley, and Posner 2018), where the environment can change in a continuously evolving way. (Kumagai and Iwata 2016) predicts future classifiers on the basis of variational Bayesian inference by incorporating the vector auto-regressive model to capture the dynamics of a decision boundary. (Wulfmeier, Bewley, and Posner 2018; Wang, He, and Katabi 2020) learn the representations that are time-invariant using adversarial methods. We emphasize that EDA still has access to unlabeled data from upcoming target domains to help learn evolving patterns, while EDG has no access to the target data at all.

There are very few works (Nasery et al. 2021; Qin, Wang, and Li 2022) tackling EDG problems. (Nasery et al. 2021) learns an operator that captures the evolving dynamics of the time-varying data distribution. (Qin, Wang, and Li 2022) proposes a novel probabilistic framework named LSSAE by incorporating variational inference to identify the continuous latent structures of *concept shift* and *covariate shift* under EDG settings. Both two EDG methods design complex Neural-Network structures and did not utilize the decision boundary alignment to mitigate the evolving domain shift, which has been verified as efficient in our illustrated example and the corresponding analysis.

5 Experiment

To evaluate our method, we demonstrate our method on several toy datasets including Sine and Rotated Gaussian toy datasets and also on the real-world datasets, including **Portraits**, **Cover Type**, **Ocular Disease** and **Caltran** (We delegate description of the datasets to the supplementary materials B). Extensive ablation studies are conducted to show the effectiveness of the meta-learning of our method.

5.1 Experiment Settings

We evaluate the proposed method with the following baselines: (1) **ERM** (Vapnik 1991); (2) **GroupDRO** (Sagawa et al. 2019); (3) **IRM** (Arjovsky et al. 2019a); (4) **CORAL** (Sun and Saenko 2016); (5) **MMD** (Li et al. 2018b); (6) **MLDG** (Li et al. 2018a); (7) **SagNet** (Nam et al. 2021); (8) **SelfReg** (Kim et al. 2021); (9) **DAML** (Shu et al. 2021); (10) **CIDA** (Wang, He, and Katabi 2020); (11) **EAML** (Liu et al. 2020); (12) **LSSAE** (Qin, Wang, and Li 2022); (13) **GI** (Nasery et al. 2021). All the baselines and experiments were implemented with DomainBed package (Gulrajani and Lopez-Paz 2020) under the same settings, which guarantees fair and sufficient comparisons. For all benchmarks, we conduct the leave-one-domain-out evaluation. We train our model on the validation splits of all seen source domains (domain 1, 2, ..., T) and select the best model on the validation of all source domains. For testing, we evaluate the selected model on all images of the held-out unseen target domain (domain $T + 1$).

5.2 Evaluation on Synthetic Dataset and Real-World Dataset

From Table 2 we can see, most algorithms fail on both Sine and rotated Gaussian datasets. Since it is a binary classification task, other methods with about 50% accuracy are doing random predictions. One reason is that their decision boundaries are static and do not make any adjustments to the direction of the data distribution shift. DDA could successfully capture such shifts and adaptively adjust the decision boundary to fit each domain including target domains. Fig. 4 visu-

DATASET	SINE	ROTATED GAUSSIAN	PORTRAIT	ROTATING MNIST	FOREST COVER	OCULAR DISEASE	CALTRAN	AVERAGE
ERM	56.3 ± 1.2	59.2 ± 1.1	90.3 ± 0.1	78.2 ± 0.2	59.8 ± 0.2	71.2 ± 0.3	96.6 ± 0.7	73.1
GROUPDRO	62.6 ± 1.5	80.8 ± 3.4	92.6 ± 0.2	79.1 ± 0.1	58.9 ± 0.5	71.3 ± 0.2	96.6 ± 0.4	77.4
IRM	51.1 ± 2.3	72.0 ± 2.2	91.3 ± 0.4	79.2 ± 0.3	58.8 ± 0.8	69.8 ± 0.4	94.9 ± 1.2	73.9
MMD	54.7 ± 4.7	56.8 ± 1.3	92.0 ± 0.2	77.4 ± 0.0	59.0 ± 0.3	67.7 ± 0.3	97.4 ± 0.2	72.1
CORAL	54.7 ± 5.4	56.8 ± 1.1	91.3 ± 0.2	78.9 ± 0.1	62.0 ± 1.1	67.8 ± 0.5	96.6 ± 0.5	72.6
MTL	54.2 ± 3.2	56.4 ± 1.4	92.0 ± 0.1	79.0 ± 0.2	60.4 ± 0.7	71.3 ± 0.4	97.5 ± 0.3	73.0
MLDG	54.7 ± 2.5	53.6 ± 2.1	91.5 ± 1.1	82.8 ± 0.2	60.9 ± 0.7	72.3 ± 0.3	97.3 ± 0.2	73.3
SAGNET	51.1 ± 3.1	52.0 ± 1.8	92.7 ± 0.2	80.9 ± 0.1	62.1 ± 2.0	69.3 ± 0.4	97.2 ± 0.1	72.2
SELFREG	55.8 ± 1.7	54.4 ± 1.0	90.6 ± 0.3	81.8 ± 0.5	60.1 ± 0.6	65.2 ± 0.1	96.5 ± 0.4	72.1
DAML	52.6 ± 0.7	62.3 ± 1.3	92.7 ± 0.3	84.0 ± 0.4	61.3 ± 0.6	71.2 ± 0.2	95.9 ± 0.3	74.3
CIDA	65.1 ± 3.7	50.5 ± 1.5	92.3 ± 0.4	83.6 ± 1.2	60.5 ± 0.9	71.4 ± 0.3	97.1 ± 0.7	74.4
EAML	49.0 ± 0.7	61.0 ± 2.8	90.1 ± 0.4	82.6 ± 0.2	60.8 ± 1.4	71.7 ± 0.6	96.5 ± 0.6	73.1
LSSAE	63.2 ± 1.5	88.4 ± 0.8	93.1 ± 0.3	84.7 ± 0.3	63.2 ± 0.4	72.4 ± 0.4	97.2 ± 1.0	80.3
GI	66.8 ± 0.7	85.1 ± 0.5	93.7 ± 0.2	83.4 ± 0.7	63.6 ± 0.4	73.1 ± 0.2	98.2 ± 0.8	80.6
OUR METHOD	$\mathbf{98.4 \pm 0.9}$	$\mathbf{99.6 \pm 0.6}$	$\mathbf{94.9 \pm 0.1}$	$\mathbf{86.2 \pm 0.3}$	$\mathbf{65.3 \pm 0.5}$	$\mathbf{74.1 \pm 0.1}$	$\mathbf{98.3 \pm 0.4}$	**88.1**

Table 2: Experiment Results (accuracy %) on Synthetic Dataset and Real-World Datasets among different methods

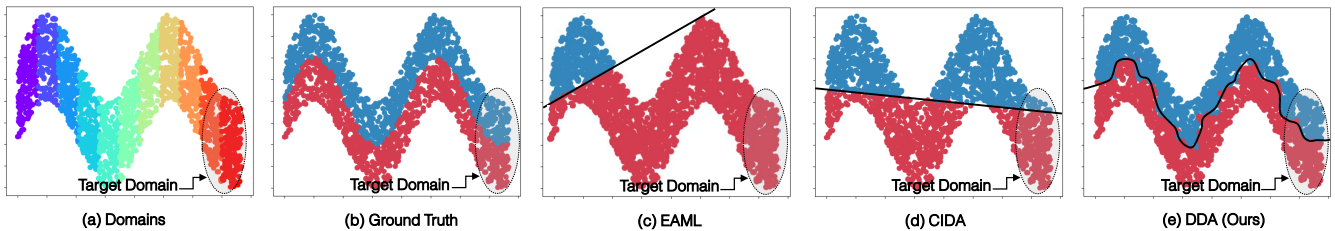(a) Domains (b) Ground Truth (c) EAML (d) CIDA (e) DDA (Ours)

Figure 4: Results on the Sine dataset with 11 domains. We set the classification model as a single linear layer which makes this task extremely difficult. The black line is the decision boundary predicted by the model. (a) Domains are indexed by color. The first 10 domains are source domains, marked by purple to orange color. The 11th domain is the test domain, marked by red color and a circle. (b) The ground truth of the decision boundaries separates positive and negative samples. (c) The prediction results of EAML on the source and target domains. (d) The prediction results of CIDA on the source and target domains. (e) The prediction results of DDA on the source and target domains.

alize the Sine dataset, which indicates an adaptive model as DDA with domain-specific classifiers can address the EDG problem properly.

DDA also achieves the best performance on real-world datasets. In the Rotating MNIST dataset, our algorithm could achieve 86.2% accuracy, which is 1.5% higher than the second best method among the baselines. Specifically, the results on the Rotating MNIST dataset are the average accuracies under different experimental settings (different number of domain intervals, different total number of domains). On Portrait and Ocular Disease datasets, DDA achieves 94.9% which is 1.2% higher than the best baselines. Ocular Disease contains medical photographs from 5,000 patients that vary with the age of the subjects. Our method improves the performance by 1.0% compared to the second best baseline, achieving 74.1% accuracy. Caltran contains images of traffic taken with stationary cameras over time. Likewise, our method also improves performance by capturing evolving patterns. They show the possibility that our method can be deployed in real-world applications.

MTL also augments the feature space with the marginal distribution of features. It indicates the superiority of data augmentations on DG problems but MTL fails to capture the domain evolving patterns and improve the performance by generalizing to unknown domains with random directions.

Hence, MTL is still worse than DDA. In our experiments, CIDA and EAML can not achieve good performance even with access to target unlabeled data. The reasons may be that both methods fail to capture the evolving pattern but instead learn domain-invariant encodings. This also shows capturing evolving patterns is critical to solving the EDG problem.

5.3 Non-stationary Environments with Multiple Target Domains

In practice, data can be streamed continuously from multiple future domains. Therefore, we also conducted experiments on Rotating MNIST by dividing more domains into target domains in Table 3. The experimental setup is to have 6 source domains with rotation degrees of $[0°, 15°, 30°, 45°, 60°, 75°]$ and 3 target domains with rotation degrees $[90°, 105°, 120°]$. The results in Table 3 demonstrates that our algorithm can also achieve better performance than other baselines in future steps. In order to generate augmentations in the $(t+2)$-th domain and the $(t+3)$-th domain, it is necessary to make some modifications to our algorithm. We show the modification details in the supplementary material E.
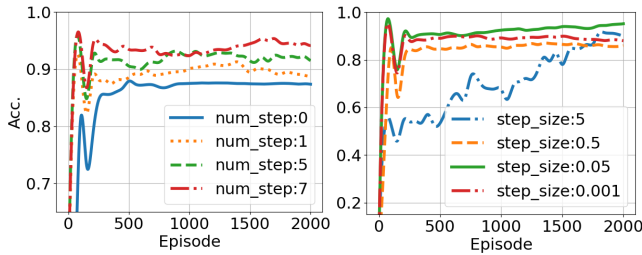
Figure 5: (Left) The convergence trajectory of the test accuracy with the different number of inner meta-updating steps on Portrait Dataset. (Right) The convergence trajectory of the test accuracy with the different step size α of inner meta-updating steps on Portrait Dataset.

DOMAINS	T+1	T+2	T+3
ERM	81.0 ± 0.2	56.5 ± 0.3	39.9 ± 0.3
MLDG	87.9 ± 0.3	66.1 ± 0.4	45.6 ± 0.4
CIDA	87.0 ± 1.3	73.5 ± 1.2	48.1 ± 1.4
EAML	88.6 ± 0.3	72.2 ± 0.3	49.9 ± 0.4
GI	89.6 ± 0.2	73.6 ± 0.3	52.4 ± 0.2
LSSAE	88.9 ± 0.3	74.2 ± 0.3	51.1 ± 0.3
OUR METHOD	$\mathbf{92.3 \pm 0.2}$	$\mathbf{77.0 \pm 0.2}$	$\mathbf{55.8 \pm 0.3}$

Table 3: Experiment Results (accuracy %) on the Rotating MNIST dataset with multiple target domains

5.4 Ablation Study

Comparison with different numbers of inner updating steps As (Finn, Abbeel, and Levine 2017) points out the number of update steps affects the convergence speed and performance, we test effects of the number of internal update steps in Fig. 5: Left. Multiple inner-loop steps will result in more computations. To reduce time complexity, we always update 2 steps on the source domains; meanwhile, we take 1 to 10 inner steps on the directional transform augmentations of the target domain. Results show the performance of DDA improves as the number of inner-loop steps increases.

The performance drops drastically by setting the number of steps to 0. With fast adaptations of the inner loop, the accuracy of the accuracy trajectory converges faster if the inner loop step size is set to 0. The performance drops drastically by setting the number of steps to 0, in which case the model is not equipped with a classification component parameterized with meta-parameters, but a domain-invariant classifier.

Comparison of step sizes in the inner-loop The step size α in Eqn. 8 is a factor related to the distance between domains. As the domain interval between the evolving domains is larger, it requires a larger α. From Fig. 5: Right we can see, setting α as 0.05 is the best choice for the Portrait dataset.

Different domain interval between domains In Table 4, the intervals of rotation degrees between domains are set to $10°$, $20°$, $30°$ and the total number of domains is fixed to 9. Our proposed method outperforms all the baselines. As the domain interval increases, we can see that all methods' performance degrades. This is because, with a larger domain discrepancy caused by the bigger domain interval, the model
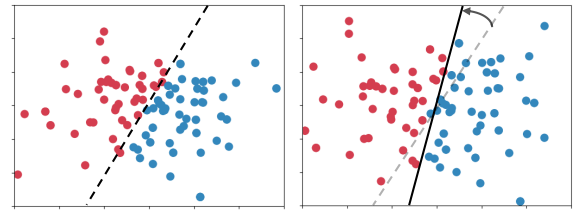

Figure 6: Visualizations of the directional augmentations in rotated Gaussian Datasets. The dashed line represents the decision boundary for data in T-th domain. The solid line represents the decision boundary for data in $T + 1$-th domain. (Left) Instance embeddings Z^T (Right) Directional transform augmentations $\tilde{Z}^{T+1}$

gets harder to capture the robust representations for classification tasks. Specifically, when the domain interval is $30°$, our method outperforms the best baseline LSSAE 1.1%.

INTERVAL	$10°$	$20°$	$30°$
ERM	90.2 ± 0.3	75.8 ± 0.3	62.0 ± 0.2
MLDG	92.2 ± 0.1	80.9 ± 0.3	70.6 ± 0.2
CIDA	92.0 ± 1.2	85.2 ± 1.4	72.1 ± 1.2
EAML	92.2 ± 0.5	84.7 ± 0.4	71.5 ± 0.4
LSSAE	92.5 ± 0.4	85.5 ± 0.3	72.4 ± 0.4
GI	93.3 ± 0.2	85.3 ± 0.1	71.8 ± 0.2
OUR METHOD	$\mathbf{95.1 \pm 0.2}$	$\mathbf{86.1 \pm 0.3}$	$\mathbf{73.5 \pm 0.2}$

Table 4: Experiment on Rotating MNIST with different intervals with total 9 domains

5.5 Visualizations of Directional Transform Augmentations

In Fig. 6, we visualize the augmentations and the source domain data in the rotated Gaussian dataset. From Fig. 6, the decision boundary of $\tilde{Z}^{T+1}$ corresponds to the $T + 1$-th domain's. It verifies that DDA successfully generates augmentations which have the same decision boundary as the instances in the next target domain.

6 Conclusion

In this paper, we address the challenging problem of Evolving Domain Generalization. We first show that a specially designed domain transformer learns to capture the domain shifts. Then, we introduce the meta-learning framework to solve the formulated bi-level optimization problem. We conduct extensive experiments on multiple datasets to demonstrate its superior effectiveness. We will further investigate when and how our method solves more complex nonstationary problems in future.

Ethics Statement

This paper proposes an algorithm that leverages evolving patterns to make predictions on the unseen target domain. The dataset we use is only intended to demonstrate the algorithm's superior performance on classification tasks.

Acknowledgements

References

Albuquerque, I.; Monteiro, J.; Darvishi, M.; Falk, T. H.; and Mitliagkas, I. 2019. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*.

Arjovsky, M.; Bottou, L.; Gulrajani, I.; and Lopez-Paz, D. 2019a. Invariant Risk Minimization. *ArXiv*, abs/1907.02893.

Arjovsky, M.; Bottou, L.; Gulrajani, I.; and Lopez-Paz, D. 2019b. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Balaji, Y.; Sankaranarayanan, S.; and Chellappa, R. 2018. Metareg: Towards domain generalization using meta-regularization. *NeurIPS*, 31: 998–1008.

Bishop, C. M. 2006. *Pattern recognition and machine learning*, volume 4. Springer.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, 1126–1135. PMLR.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1): 2096–2030.

Girdhar, R.; and Grauman, K. 2021. Anticipative video transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13505–13515.

Gulrajani, I.; and Lopez-Paz, D. 2020. In Search of Lost Domain Generalization. *CoRR*, abs/2007.01434.

Hinton, G. E.; Vinyals, O.; and Dean, J. 2015. Distilling the Knowledge in a Neural Network. *ArXiv*, abs/1503.02531.

Hoffman, J.; Darrell, T.; and Saenko, K. 2014. Continuous manifold based adaptation for evolving visual domains. In *Proceedings of the CVPR*, 867–874.

Kim, D.; Yoo, Y.; Park, S.; Kim, J.; and Lee, J. 2021. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the IEEE/CVF ICCV*, 9619–9628.

Kumagai, A.; and Iwata, T. 2016. Learning future classifiers without additional data. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Lampert, C. H. 2015. Predicting the future behavior of a time-varying probability distribution. In *Proceedings of the IEEE CVPR*, 942–950.

Li, B.; Wang, Y.; Zhang, S.; Li, D.; Keutzer, K.; Darrell, T.; and Zhao, H. 2021. Learning invariant representations and risks for semi-supervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1104–1113.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. 2020. Sequential learning for domain generalization. In *European Conference on Computer Vision*, 603–619. Springer.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2018a. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Li, H.; Pan, S.-J.; Wang, S.; and Kot, A.-C. 2018b. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE CVPR*.

Li, Y.; Yang, Y.; Zhou, W.; and H., T. 2019. Feature-critic networks for heterogeneous domain generalization. In *International Conference on Machine Learning*, 3915–3924. PMLR.

Liu, H.; Long, M.; Wang, J.; and Wang, Y. 2020. Learning to Adapt to Evolving Domains. In *NeurIPS*.

Mancini, M.; Bulo, S.-R.; Caputo, B.; and Ricci, E. 2018. Best sources forward: domain generalization through source-specific nets. In *2018 25th IEEE ICIP*, 1353–1357. IEEE.

Muandet, K.; Balduzzi, D.; and Schölkopf, B. 2013. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, 10–18. PMLR.

Nam, H.; Lee, H.; Park, J.; Yoon, W.; and Yoo, D. 2021. Reducing Domain Gap by Reducing Style Bias. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8686–8695.

Nasery, A.; Thakur, S.; Piratla, V.; De, A.; and Sarawagi, S. 2021. Training for the Future: A Simple Gradient Interpolation Loss to Generalize Along Time. *Advances in Neural Information Processing Systems*, 34: 19198–19209.

Nguyen, A. T.; Tran, T.; Gal, Y.; and Baydin, A. G. 2021. Domain Invariant Representation Learning with Domain Density Transformations. *arXiv preprint arXiv:2102.05082*.

Qin, T.; Wang, S.; and Li, H. 2022. Generalizing to Evolving Domains with Latent Structure-Aware Sequential Autoencoder. *arXiv preprint arXiv:2205.07649*.

Rahman, M.-M.; Fookes, C.; Baktashmotlagh, M.; and Sridharan, S. 2019. Multi-component image translation for deep domain generalization. In *2019 IEEE WACV*, 579–588. IEEE.

Sagawa, S.; Koh, P. W.; Hashimoto, T.-B.; and Liang, P. 2019. Distributionally Robust Neural Networks for Group Shifts: On the Importance of Regularization for Worst-Case Generalization. *ArXiv*, abs/1911.08731.

Shi, Y.; Yu, X.; Sohn, K.; Chandraker, M.; and Jain, A.-K. 2020. Towards Universal Representation Learning for Deep Face Recognition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6816–6825.

Shu, Y.; Cao, Z.; Wang, C.; Wang, J.; and Long, M. 2021. Open domain generalization with domain-augmented meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9624–9633.

Shui, C.; Wang, B.; and Gagné, C. 2022. On the benefits of representation regularization in invariance based domain generalization. *Machine Learning*, 111(3): 895–915.

Sun, B.; and Saenko, K. 2016. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *ECCV Workshops*.

Vapnik, V. 1991. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4.

Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. *ArXiv*, abs/1706.03762.

Volpi, R.; and Murino, V. 2019. Addressing Model Vulnerability to Distributional Shifts Over Image Transformation Sets. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 7979–7988.

Volpi, R.; Namkoong, H.; Sener, O.; Duchi, J.; Murino, V.; and Savarese, S. 2018. Generalizing to unseen domains via adversarial data augmentation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 5339–5349.

Wang, H.; He, H.; and Katabi, D. 2020. Continuously Indexed Domain Adaptation. *arXiv preprint arXiv:2007.01807*.

Wulfmeier, M.; Bewley, A.; and Posner, I. 2018. Incremental Adversarial Domain Adaptation for Continually Changing Environments. *2018 IEEE ICRA*, 1–9.

Xu, T.; Chen, W.; Pichao, W.; Wang, F.; Li, H.; and Jin, R. 2021. CDTrans: Cross-domain Transformer for Unsupervised Domain Adaptation. In *International Conference on Learning Representations*.

Zeng, Y.; Fu, J.; and Chao, H. 2020. Learning joint spatial-temporal transformations for video inpainting. In *European Conference on Computer Vision*, 528–543. Springer.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Zhang, Y.; Li, K.; Li, K.; Wang, L.; Zhong, B.; and Fu, Y. 2018. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, 286–301.

Zhou, F.; Jiang, Z.; Shui, C.; Wang, B.; and Chaib-draa, B. 2021a. Domain generalization via optimal transport with metric similarity learning. *Neurocomputing*, 456: 469–480.

Zhou, K.; Yang, Y.; H., T.; and X., T. 2020a. Deep domain-adversarial image generation for domain generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 13025–13032.

Zhou, K.; Yang, Y.; Hospedales, T.-M.; and Xiang, T. 2020b. Learning to Generate Novel Domains for Domain Generalization. *ArXiv*, abs/2007.03304.

Zhou, K.; Yang, Y.; Qiao, Y.; and Xiang, T. 2021b. Domain Generalization with MixStyle. *ArXiv*, abs/2104.02008.