ADAPTIVE INVARIANT REPRESENTATION LEARNING FOR NON-STATIONARY DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Although recent advances in machine learning have shown its success to learn from independent and identically distributed (IID) data, it is vulnerable to out-of-distribution (OOD) data in an open world. Domain generalization (DG) deals with such an issue and it aims to learn a model from multiple source domains that can be generalized to unseen target domains. Existing studies on DG have largely focused on stationary settings with homogeneous source domains. However, in many applications, domains may evolve along a specific direction (e.g., time, space). Without accounting for such non-stationary patterns, models trained with existing methods may fail to generalize on OOD data. In this paper, we study domain generalization in non-stationary environment. We first examine the impact of environmental non-stationarity on model performance and establish the theoretical upper bounds for the model error at target domains. Then, we propose a novel algorithm based on invariant representation learning, which leverages the non-stationary pattern to train a model that attains good performance on target domains. Experiments on both synthetic and real data validate the proposed algorithm.

1 INTRODUCTION

Many machine learning (ML) systems are built based on the assumption that training and test data are sampled independently and identically from the same distribution. However, this is commonly violated in real applications where the environment changes during model deployment, and there exist distribution shifts between train and test data. The problem of training models that are robust under distribution shifts is typically referred to as domain adaptation (or generalization), where the goal is to train a model on *source* domain that can generalize well on a *target* domain. Specifically, domain adaptation (DA) aims to deploy model on a *specific* target domain, and it assumes the (unlabeled) data from this target domain is accessible during training. In contrast, domain generalization (DG) considers a more realistic scenario where target domain data is unavailable during training; instead it leverages multiple source domains to learn models that generalize to *unseen* target domains.

For both DA and DG, various approaches have been proposed to learn a robust model with high performance on target domains. However, most of them assume both source and target domains are sampled from a *stationary* environment; they are not suitable for settings where the data distribution evolves along a specific direction (e.g., time, space). Indeed, evolvable data distributions have been observed in many applications. For example, satellite images change over time due to city development and climate change (Christie et al., 2018), clinical data evolves due to changes in disease prevalence (Guo et al., 2022), facial images gradually evolve because of the changes in fashion and social norms (Ginosar et al., 2015). Without accounting for the non-stationary patterns across domains, existing methods in DA/DG designed for stationary settings may not perform well in non-stationary environments. As evidenced by Guo et al. (2022), clinical predictive models trained under existing DA/DG methods cannot perform better on future clinical data compared to empirical risk minimization.

In this paper, we study *domain generalization* (DG) in non-stationary environments. The goal is to learn a model from a sequence of source domains that can capture the non-stationary patterns and generalize well to *unseen* target domains (Figure 1). We first examine the impacts of non-stationary distribution shifts and study how the model performance attained on source domains can be affected when the model is deployed on target domains. Based on the theoretical findings, we propose an

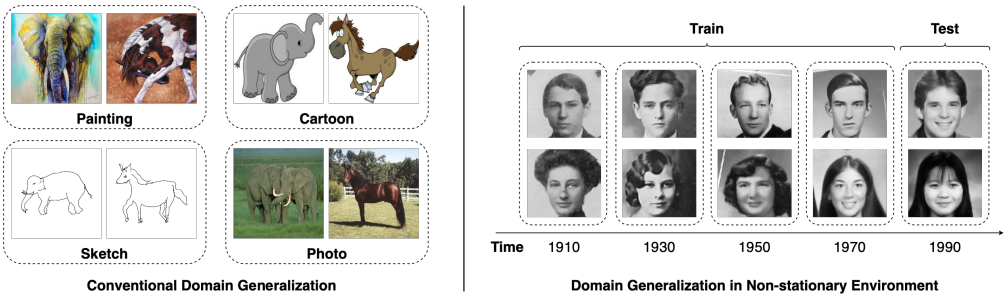


Figure 1: An illustrative comparison between conventional DG and DG in non-stationary environment: domains in conventional DG are independently sampled from a stationary environment, whereas DG in non-stationary environment considers domains that evolve along a specific direction. As shown in the right plot, data (i.e., images) changes over time and the model trained on past data may not have good performance on future data due to non-stationarity (i.e., temporal shift).

algorithm named Adaptive Invariant Representation Learning (AIRL); it minimizes the error on target domains by learning a sequence of representations that are *invariant* for every two consecutive source domains but are *adaptive* across these pairs of source domains.

In particular, AIRL consists of two components: (i) *representation network*, which is trained on the sequence of source domains to learn invariant representations between every two consecutive source domains, (ii) *classification network* that minimizes the prediction errors on source domains. Our main idea is to create adaptive representation and classification networks that can evolve in response to the dynamic environment. In other words, we aim to find networks that can effectively capture the non-stationary patterns from the sequence of source domains. At the inference stage, the representation network is used to generate the optimal representation mappings and the classification network is used to make predictions in the target domains, without the need to access their data. To verify the effectiveness of AIRL, we conduct extensive experiments on both synthetic and real data and compare AIRL with various existing methods.

2 Related Work

This work is closely related to the literature on domain generalization, continuous (or gradual) domain adaptation, continual learning. We introduce each topic and discuss their differences with our work.

Domain generalization (DG). The goal is to learn a model on multiple source domains that can generalize to the out-of-distribution samples from an unseen target domain. Depending on the learning strategy, existing works for DG can be roughly classified into three categories: (i) methods based on *domain-invariant representation learning* (Phung et al., 2021; Nguyen et al., 2021); (ii) methods based on *data manipulation* (Qiao et al., 2020; Zhou et al., 2020); (iii) methods by considering DG in general ML paradigms and using approaches such as *meta-learning* (Li et al., 2018a; Balaji et al., 2018), *gradient operation* (Rame et al., 2021; Tian et al., 2022), *self-supervised learning* (Jeon et al., 2021; Li et al., 2021), and *distributional robustness* (Koh et al., 2021; Wang et al., 2021). However, these works assume both source and target domains are sampled from a *stationary* environment and they do not consider the non-stationary patterns across domains; this differs from our setting.

Non-stationary environment DG. To the best of our knowledge, only a few concurrent works study domain generalization in non-stationary environments (Bai et al., 2022; Wang et al., 2022; Qin et al., 2022; Zeng et al., 2023). However, the problem settings considered in these works are rather limited. For example, Wang et al. (2022); Qin et al. (2022) only focus on the environments that evolve based on a *consistent* and *stationary* transition function; the approaches in Bai et al. (2022); Wang et al. (2022); Zeng et al. (2023) can only generalize the model to a *single subsequent* target domain. In contrast, this paper considers a more general setting where data may evolve based on non-stationary dynamics, and the proposed algorithm can generate models for multiple unseen target domains.

Continuous domain adaptation (DA). Unlike conventional DA/DG methods that only consider categorical domain labels, continuous DA admits continuous domain labels such as space, time (Ortiz-Jimenez et al., 2019; Wang et al., 2020; Mancini et al., 2019). Specifically, this line of research considers scenarios where the data distribution changes gradually and domain labels are continuous.

Similar to conventional DA, samples from target domain are required to guide the model adaptation process. This is in contrast to this study, which considers the target domains whose samples are inaccessible during training.

Gradual domain adaptation. Similar to continuous DA, Gradual DA also considers continuous domain labels, and the samples from the target domain are accessible during training (Kumar et al., 2020; Chen et al., 2020; Chen & Chao, 2021). The prime difference is that continuous DA focuses on the generalization from a single source domain to a target domain, whereas there are multiple source domains in gradual DA.

Continual learning (CL). The goal is to learn a model continuously from a sequence of tasks. The main focus in CL is to overcome the issue of catastrophic forgetting, i.e., prevent forgetting the old knowledge as the model is learned on new tasks (Chaudhry et al., 2018; Kirkpatrick et al., 2017; Mallya & Lazebnik, 2018). This differs from time-evolving DG (i.e., a special case of our setting) which aims to train a model on past data that can generalize to future.

3 Problem Formulation

We first introduce the notations used throughout the paper and then formulate the problem.

Notations. Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input and output space, respectively. We use capitalized letters X, Y to denote random variables that take values in $\mathcal{X}, \mathcal{Y}$ and small letters x, y their realizations. A *domain* D is specified by distribution $P_D^{X,Y} : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ and labeling function $h_D : \mathcal{X} \to \mathcal{Y}^\Delta$, where Δ is a probability simplex over $\mathcal{Y}$. For simplicity, we also use P_D^V (or $P_D^{V|U}$) to denote the induced marginal (or conditional) distributions of random variable V (given U) in the domain D.

Non-stationary domain generalization setup. Consider a domain generalization problem where a learning algorithm has access to data sampled from a sequence of T source domains $\{D_t\}_{t=1}^T$. We assume there exists a mechanism M that captures non-stationary patterns in the data. Specifically, M can generate a sequence of mapping functions $\{m_t\}_{t\in\mathbb{N}}$ in which $m_t : \mathcal{X} \times \mathcal{Y} \to \mathcal{X} \times \mathcal{Y}$ captures the transition from domain D_{t-1} to domain D_t. In other words, we can regard $P_{D_t}^{X,Y}$ as the push-forward distribution induced from $P_{D_{t-1}}^{X,Y}$ using the mapping function m_{t-1} (i.e., $P_{D_t}^{X,Y} := m_{t-1}\sharp P_{D_{t-1}}^{X,Y}$). Note that this setup should not be considered as multiple DA problems where m_t are not related to each other. In non-stationary domain generalization, m_t depends on previous mappings $\{m_1, m_2, \cdots, m_{t-1}\}$ via mechanism M.

Our goal is to learn a model $\widehat{h} : \mathcal{X} \to \mathcal{Y}^\Delta$ from samples collected in source domains that can perform well on *unseen* target domains $\{D_t\}_{t\in\mathbb{N},t\geq T+1}$. Specifically, we investigate under what conditions and by what algorithms we can ensure that attaining high accuracy at source domains $\{D_t\}_{t=1}^T$ implies high accuracy at unknown target domains $\{D_t\}_{t\geq T+1}$ under non-stationary environment. Formally, the accuracy can be measured using an error metric defined below.

Error metric. Consider a model $\widehat{h} : \mathcal{X} \to \mathcal{Y}^\Delta$ in a hypothesis class $\mathcal{H}$, we denote $\widehat{h}(x)_y$ as the element on y-th dimension which predicts $\Pr(Y = y|X = x)$. Then the *expected error* of $\widehat{h}$ under domain D for some loss function $L : \mathcal{Y}^\Delta \times \mathcal{Y} \to \mathbb{R}_+$ (e.g., 0-1, cross-entropy loss) can be defined as $\epsilon_D\left(\widehat{h}\right) = \mathbb{E}_{x,y\sim D}\left[L\left(\widehat{h}(X), Y\right)\right]$. Similarly, the *empirical error* of $\widehat{h}$ over n samples S_n drawn i.i.d. from $P_D^{X,Y}$ is defined as $\widehat{\epsilon}_D\left(\widehat{h}\right) = \frac{1}{n}\sum_{x,y\in S_n} L\left(\widehat{h}(x), y\right)$. We also denote a family of functions $\mathcal{L}_\mathcal{H}$ associated with loss function L and hypothesis class $\mathcal{H}$ as $\mathcal{L}_\mathcal{H} = \left\{(x, y) \to L\left(\widehat{h}(x), y\right) : \widehat{h} \in \mathcal{H}\right\}$.

4 Theoretical Results

In this section, we aim to understand how a model trained with source domain data would perform when deployed in a target domain under non-stationary distribution shifts. Specifically, we will develop theoretical upper bounds of the model error at target domains. These theoretical findings will provide guidance for the algorithm design in Section 5. All proofs are in Appendix A.

To learn a model $\widehat{h}$ that performs well on an unseen target domain, we need to take into account the non-stationary patterns across domains. However, these patterns are unknown and must be estimated from a sequence of available source domains. Therefore, in addition to the model $\widehat{h}$, we need to learn a hypothesis mechanism $\widehat{M}$ from a hypothesis class $\mathcal{M}$ that estimates the ground-truth M, and the model $\widehat{h}$ needs to be learned by leveraging the estimation $\widehat{M}$. Because the target data is inaccessible, we expect that the model performance on the target will highly rely on the accuracy of the hypothesis $\widehat{M}$. That is, the non-stationary patterns explored from the source data should well-estimate the target. To formally characterize the consistency in terms of performance of mechanism $\widehat{M}$ on capturing non-stationary patterns on source and target domains, we introduce a discrepancy measure below.

Definition 4.1. Given a sequence of source domains $\{D_t\}_{t=1}^{T}$ and a target domain D_{T+1}, the non-stationary consistency of hypothesis mechanism $\widehat{M}$ can be measured as:

$$\Phi\left(\widehat{M}\right) = \left| \mathcal{D}\left(P^{X,Y}_{D_{T+1}}, P^{X,Y}_{\widehat{D}_{T+1}}\right) - \frac{1}{T-1}\sum_{t=2}^{T} \mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right) \right|$$

where $P^{X,Y}_{\widehat{D}_t} = \widehat{m}_{t-1}\sharp P^{X,Y}_{D_{t-1}}$ is a push-forward distribution induced from $P^{X,Y}_{D_{t-1}}$ using mapping function $\widehat{m}_{t-1}$ generated by hypothesis mechanism $\widehat{M}$, and $\mathcal{D}(\cdot,\cdot)$ is a statistical distance that measures the distance between two distributions. In our study, we consider Kullback–Leibler (KL) divergence or Jensen–Shannon (JS) divergence as $\mathcal{D}(\cdot,\cdot)$.

Remark 4.2. $\Phi\left(\widehat{M}\right)$ consists of two quantities: average error of $\widehat{M}$ on sequence of source domains $\frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)$ and error of $\widehat{M}$ on the target domain $\mathcal{D}\left(P^{X,Y}_{D_{T+1}}, P^{X,Y}_{\widehat{D}_{T+1}}\right)$. In essence, $\Phi\left(\widehat{M}\right)$ measures how consistent in terms of performance of mechanism $\widehat{M}$ on capturing non-stationary patterns on source and target domains.

Next, we impose the following assumption on the learnability of mechanism M with respect to hypothesis class $\mathcal{M}$ as follows.

Assumption 4.3. We assume the mechanism M is (ϵ, δ)-learnable with respect to hypothesis class $\mathcal{M}$. That is, for $\epsilon, \delta > 0$, with probability at least $1-\delta$, we have $\Phi\left(M^*\right) \leq \epsilon$ where $M^* = \arg\min_{\widehat{M}\in\mathcal{M}} \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)$.

We note that Assumption 4.3 is mild because it is required only for the optimal mechanism M^*. This assumption implies that there exists at least one hypothesis in $\mathcal{M}$ under which the non-stationary patterns learned from source can generalize sufficiently well to the target (with bounded Φ). During training, we can feasibly achieve small error of M^* on sequence of source domains by selecting high-capacity hypothesis class (e.g., neural network). Thus, Assumption 4.3 also implies that error of M^* on target domain is small ($\approx \epsilon$).

In practice, the model has access to data sample only. To construct an analysis with finite data, we leverage *Rademacher complexity* which measures the richness of a class of real-valued functions with respect to a probability distribution. Definition of *Rademacher complexity* is given below.

Definition 4.4. Let $\mathcal{F}$ be a set of real-valued functions defined over a set $\mathcal{Z}$. Let $S_n = \{z_1, \cdots, z_n\} \in \mathcal{Z}^n$ be a set of examples drawn i.i.d. from P^Z. The empirical Rademacher complexity of $\mathcal{F}$ is defined as follows:

$$\widehat{\mathcal{R}}_n(\mathcal{F}) = \mathbb{E}_\sigma\left[\sup_{f\in\mathcal{F}}\left(\frac{1}{n}\sum_{i=1}^{n}\sigma_i f(z_i)\right)\right]$$

where $\sigma = (\sigma_1, \cdots, \sigma_n)$ are independent random variables uniformly chosen from $\{-1, 1\}$.

Based on the Rademacher complexity and the generalizability measure defined above, we can establish an upper bound on the expected error of the target domain D_{T+1} in Theorem 4.5 below.

Theorem 4.5. *Suppose loss function L is upper bounded by C and Assumption 4.3 holds. For any $0 < \delta < 1$, then with probability at least $1-\delta$, the following holds for all $\widehat{h} : \mathcal{X} \to \mathcal{Y}^{\Delta}$:*

$$\epsilon_{D_{T+1}}\left(\widehat{h}\right) \leq \widehat{\epsilon}_{D^*_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\tfrac{\log(4/\delta)}{2n}} + CK\sqrt{\epsilon + \tfrac{1}{T-1}\textstyle\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{D^*_t}\right)}$$

where $P^{X,Y}_{D^*_t} = m^*_{t-1}\sharp P^{X,Y}_{D_{t-1}}$ *is a push-forward distribution induced from* $P^{X,Y}_{D_{t-1}}$ *using mapping function* m^*_{t-1} *generated by* $M^* = \arg\min_{\widehat{M}\in\mathcal{M}} \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)$, n *is the sample size of domain* D_T, *and* $K = \frac{1}{\sqrt{2}}$ *(resp.* $K = \sqrt{2}$*) when* $\mathcal{D}$ *is KL-divergence (resp. JS-divergence).*

Theorem 4.5 suggests one way to minimize the error in target domain D_{T+1}: to minimize $\epsilon_{D_{T+1}}(\widehat{h})$, we need to (i) learn the optimal $M^* = \arg\min_{\widehat{M}\in\mathcal{M}} \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)$ from the sequence of T source domains; and (ii) find a classifier that minimizes the empirical error $\widehat{\epsilon}_{D^*_{T+1}}(\widehat{h})$ in domain D^*_{T+1} induced from domain D_T by function m^*_T.

Learning M^* requires the model to find the optimal mapping $m^*_{t-1} : \mathcal{X}\times\mathcal{Y} \to \mathcal{X}\times\mathcal{Y}$ that minimizes the distance of the joint distributions $\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)$. In our setting, we handle this problem by first minimizing the distance of output distribution between the two domains D_t, D_{t-1}, then finding an optimal mapping function in input space $\mathcal{X}$. That is, minimize the distance of joint distributions in output and input space separately. This approach is formally stated in Proposition 4.6 below.

Proposition 4.6. *Let* $P^{X,Y}_{D^w_{t-1}}$ *be the distribution induced from* $P^{X,Y}_{D_{t-1}}$ *by importance weighting with factors* $\{w_y\}_{y\in\mathcal{Y}}$ *where* $w_y = P^{Y=y}_{D_t}/P^{Y=y}_{D_{t-1}}$ *(i.e.,* $P^{X=x,Y=y}_{D^w_{t-1}} = w_y \times P^{X=x,Y=y}_{D_{t-1}}$*). Then for any mechanism* $\widehat{M}$ *that generates* $\{\widehat{m}_t : \mathcal{X}\to\mathcal{X}\}_{t\in\mathbb{N}}$*, we have the following:*

$$\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}^w_t}\right) = \mathbb{E}_{y\sim P^Y_{D_t}}\left[\mathcal{D}\left(P^{X|Y}_{D_t}, P^{X|Y}_{\widehat{D}^w_t}\right)\right]$$

where $P^{X,Y}_{\widehat{D}^w_t} = \widehat{m}_{t-1}\sharp P^{X,Y}_{D^w_{t-1}}$ *is a push-forward distribution induced from* $P^{X,Y}_{D^w_{t-1}}$ *using* $\widehat{m}_{t-1}$.

Proposition 4.6 suggests that to learn $m^*_t : \mathcal{X}\times\mathcal{Y} \to \mathcal{X}\times\mathcal{Y}$, we can first reweight $P^{X,Y}_{D_{t-1}}$ with factors $\{w_y\}_{y\in\mathcal{Y}}$ (i.e., to minimize the distance of output distribution between domains D_t, D_{t-1}), then learn $m^*_t : \mathcal{X}\to\mathcal{X}$ in input space that minimizes the distance of conditional distribution $\mathcal{D}\left(P^{X|Y}_{D_t}, P^{X|Y}_{\widehat{D}^w_t}\right)$.

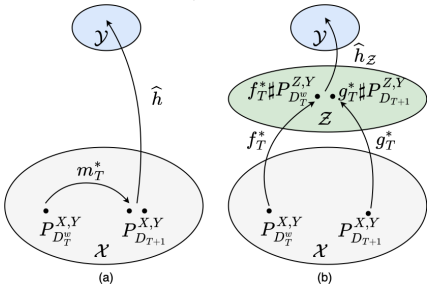


Figure 2: Visualization of learning in non-stationary environment. (a) Learning directly from input space $\mathcal{X}$. (b) Learning via representation space $\mathcal{Z}$.

Since the input space $\mathcal{X}$ may be of high dimension, estimating M^* in high-dimensional space can be challenging in practice. To tackle this issue, we leverage the *representation learning* approach to first map inputs to a representation space $\mathcal{Z}$, which often has a lower dimension than $\mathcal{X}$. In particular, for all $t < T$, instead of using $m^*_t : \mathcal{X}\to\mathcal{X}$ to map $P^X_{D^w_t}$ to $P^X_{D^*_{t+1}}$, we use $f^*_t : \mathcal{X}\to\mathcal{Z}$ and $g^*_t : \mathcal{X}\to\mathcal{Z}$ to map $P^X_{D^w_t}$ and $P^X_{D_{t+1}}$ to $f^*_t\sharp P^Z_{D^w_t}$ and $g^*_t\sharp P^Z_{D_{t+1}}$ such that $\mathbb{E}\left[\mathcal{D}\left(g^*_t\sharp P^{Z|Y}_{D_{t+1}}, f^*_t\sharp P^{Z|Y}_{D^w_t}\right)\right]$ is minimal. Then, we learn the classifier $\widehat{h}_{\mathcal{Z}} : \mathcal{Z}\to\mathcal{Y}^\Delta$ from representation to output spaces that minimizes the empirical error with respect to the distribution $f^*_T\sharp P^{Z,Y}_{D^w_T}$. This representation learning-based method is visualized in Figure 2 and is summarized below.

Remark 4.7 (Representation learning)**.** Given the sequence of T source domains, we estimate:

(i) Representation mappings $F^* = \{f^*_t : \mathcal{X}\to\mathcal{Z}\}$, $G^* = \{g^*_t : \mathcal{X}\to\mathcal{Z}\}$ with

$$F^*, G^* = \arg\min_{\widehat{F}\in\mathcal{F},\widehat{G}\in\mathcal{G}} \frac{1}{T-1}\sum_{t=2}^{T}\mathbb{E}\left[\mathcal{D}\left(\widehat{g}_{t-1}\sharp P^{Z|Y}_{D_t}, \widehat{f}_{t-1}\sharp P^{Z|Y}_{D^w_{t-1}}\right)\right],$$

where $\widehat{F} = \left\{\widehat{f}_t : \mathcal{X}\to\mathcal{Z}\right\}$, $\widehat{G} = \{\widehat{g}_t : \mathcal{X}\to\mathcal{Z}\}$, $\mathcal{F}$ and $\mathcal{G}$ are the function class of $\widehat{F}$ and $\widehat{G}$.

(ii) Classifier $\widehat{h}_{\mathcal{Z}} : \mathcal{Z}\to\mathcal{Y}^\Delta$ that minimizes the empirical error with respect to distribution $f^*_T\sharp P^{Z,Y}_{D^w_T}$.

Remark 4.8 (Comparison with conventional DG)**.** A key distinction from non-stationary DG is that the model evolves over the domain sequence to capture non-stationary patterns (i.e., learn

invariant representations between two consecutive domains but adaptive across domain sequence). This stands in contrast to the conventional DG (Ganin et al., 2016; Phung et al., 2021) which relies on an assumption that target domains lie on or are near the mixture of source domains, then enforcing fixed invariant representations across all source domains can help to generalize the model to target domains. We argue that this assumption may not hold in non-stationary DG where the target domains may be far from the mixture of source domains resulting in the failure of the existing methods. This argument is also validated in our experiment in Appendix C.

5 Proposed Algorithm

Overview. Based on Remark 4.7, we propose AIRL, a novel model that learns adaptive invariant representations from a sequence of T source domains. AIRL includes two components: (i) *representation network* that learns the sequences of mapping functions $F^* = [f_1^*, f_2^*, \cdots, f_{T-1}^*]$ and $G^* = [g_1^*, g_2^*, \cdots, g_{T-1}^*]$ from input space to representation space, (ii) *classification network* that learns the sequence of classifiers $H^* = [h_1^*, h_2^*, \cdots, h_{T-1}^*]$ [1] from representation to the output spaces. Figure 3 shows the overall architecture of AIRL; the technical details of each component are presented in Appendix B. The learning and inference processes of AIRL are formally stated as follows.

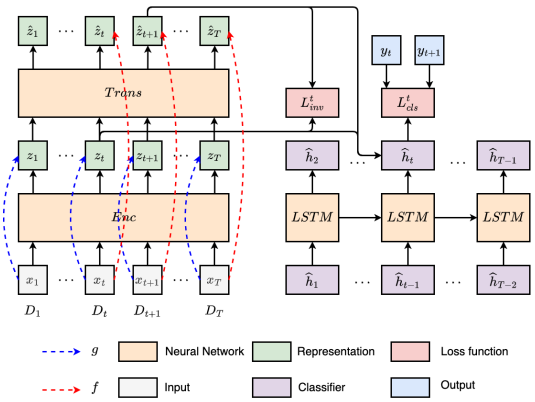


Figure 3: Overall architecture of AIRL and the visualization of its learning process.

5.1 Learning

Representation mappings F^* and G^*, and classifiers H^* in AIRL can be learned by solving an optimization problem over T source domains $\{D_t\}_{t=1}^T$:

$$F^*, G^*, H^* = \arg\min_{\widehat{F}, \widehat{G}, \widehat{H}} \sum_{t=1}^{T-1} \mathcal{L}_{cls}^t + \alpha \mathcal{L}_{inv}^t \tag{1}$$

where $\mathcal{L}_{cls}^t$ is the prediction loss on source domains D_t and D_{t+1}; $\mathcal{L}_{inv}^t$ enforces the representations are invariant across a pair of consecutive domains D_t, D_{t+1}; hyper-parameter α controls the trade-off between two objectives. Note that unlike the theoretical results in Section 4 which suggests minimizing the prediction loss on *last* source domain D_T, objective equation 1 minimizes the total prediction loss on previous source domains $\{D_t\}_{t<T}$. The reason is that we want to learn the representations that are not only invariant but also helpful for learning the sequence of classifiers H^*.

Next, we present the detailed architecture of the *representation network* and the *classification network*. To satisfy Assumption 4.3 and ensure the patterns learned from source can generalize well to the target, the function classes $\mathcal{F}$ and $\mathcal{G}$ should be chosen such that the discrepancy $\Phi(F^*, G^*)$ (adapted from Definition 4.1 for representation learning method) below is sufficiently small

$$\Phi(F^*, G^*) = \left| \mathcal{D}\left(g_T^* \sharp P_{D_{T+1}}^{Z,Y}, f_T^* \sharp P_{D_T^w}^{Z,Y}\right) - \frac{1}{T-1} \sum_{t=2}^{T} \mathcal{D}\left(g_{t-1}^* \sharp P_{D_t}^{Z,Y}, f_{t-1}^* \sharp P_{D_{t-1}^w}^{Z,Y}\right) \right|$$

In other words, the *representation network* needs to learn non-stationary patterns from source domains such that f_T^* and g_T^* generated from this network can minimize $\mathcal{D}\left(g_T^* \sharp P_{D_{T+1}}^{Z,Y}, f_T^* \sharp P_{D_T^w}^{Z,Y}\right)$.

In practice, the representation mappings may be complex (e.g., ResNet (He et al., 2016)), then explicitly capturing the evolving of these mappings is challenging. We surpass this bottleneck by capturing the evolving of representation space induced by these mappings instead. Formally, our *representation network* consists of an encoder Enc which maps from input to representation spaces,

[1] To avoid complex notation, we use h^* and $\widehat{h}$ instead of $h_{\mathcal{Z}}^*$ and $\widehat{h}_{\mathcal{Z}}$ in this section.

and Transformer layer Trans which learns the non-stationary pattern from a sequence of source domains using attention mechanism. Given the batch sample $\mathcal{B} := \{x_t, y_t\}_{t \leq T}$ from T source domains where $\{x_t, y_t\} = \{x_t^j, y_t^j\}_{j=1}^n$ are samples for domain D_t, the encoder first maps each input x_t^j to a representation $z_t^j = \mathrm{Enc}\left(x_t^j\right)$, $\forall t \leq T, j \leq n$. Then, Transformer layer (Vaswani et al., 2017) Trans is used to generate representation $\widehat{z}_t^j$ from the sequence $z_{\leq t}^j = \left[z_1^j, z_2^j, \cdots, z_t^j\right]$. Specifically, $\forall j, t$, Trans leverages four feed-forward networks Q, K, V, U to compute $\widehat{z}_t^j$ as follow:

$$\widehat{z}_t^j = \left(a_{\leq t}^j\right)^\top V\left(z_{\leq t}^j\right) + U\left(z_t^j\right) \quad \text{with} \quad a_{\leq t}^j = \frac{K\left(z_{\leq t}^j\right)^\top Q\left(z_t^j\right)}{\sqrt{d}}$$

In our design, the computational paths from x_{t+1}^j to z_{t+1}^j and from x_t^j to $\widehat{z}_t^j$ are considered as $\widehat{g}_t$ and $\widehat{f}_t$, respectively. In particular, $z_{t+1}^j = \widehat{g}_t(x_{t+1}^j)$ and $\widehat{z}_{t+1}^j = \widehat{f}_t(x_{t+1}^j)$. The main goal of this design is as follows: By incorporating historical data into computation, representation space constructed by $\widehat{f}_t$ can capture evolving pattern across domain sequence. However, this design requires access to the historical data during inference which might not be feasible in practice. To avoid it, we enforce $\widehat{g}_t$, which obviates the need to access historical data, to mimic representation space constructed by $\widehat{f}_t$.

As shown in Remark 4.7, our goal is to enforce invariant representation constraint (i.e., $\mathcal{L}_{inv}^t$ in objective (1)) for every pair of two consecutive domains D_t, D_{t+1} constructed by $\widehat{f}_t$ and $\widehat{g}_t$ instead of learning a network that achieves invariant representations for all source domains together. Thus, the representations constructed by $\widehat{f}_t$ and $\widehat{g}_t$ might not be aligned with the ones constructed by $\widehat{f}_{t'}$ and $\widehat{g}_{t'}$. This implies that we should also make the *classification network* adaptive to changes in domain to achieve the best performance. Due to the simplicity of the classifier (i.e., 1 or 2-layer network), we leverage long short-term memory (Hochreiter & Schmidhuber, 1997) LSTM to explicitly capture the evolving of the classifier over domain sequence. Specifically, the weights of previous classifiers $\widehat{h}_{<t} = \left[\widehat{h}_1, \widehat{h}_2, \cdots, \widehat{h}_{t-1}\right]$ are vectorized and put into LSTM to generate the weights of $\widehat{h}_t$.

Because $\widehat{f}_t, \widehat{g}_t, \widehat{h}_t \; \forall t < T$ are functions of the *representation network* and *the classification network*, the weights of these two networks are updated using the backpropagated gradients for objective (1). The pseudo-code of the complete learning process for `AIRL` is shown in Algorithm 1, Appendix B. Next, we present the details of each loss term used in optimization.

Prediction loss $\mathcal{L}_{cls}^t$: We adopt cross-entropy loss for classification tasks. Specifically, $\mathcal{L}_{cls}^t$ for the optimization over domains D_t, D_{t+1} is defined as follows.

$$\mathcal{L}_{cls}^t = \mathbb{E}_{D_t^w}\left[-\log\left(\frac{\widehat{h}_t(\widehat{f}_t(X))_Y}{\sum_{y' \in \mathcal{Y}} \widehat{h}_t(\widehat{f}_t(X))_{y'}}\right)\right] + \mathbb{E}_{D_{t+1}}\left[-\log\left(\frac{\widehat{h}_t\left(\widehat{g}_t(X)\right)_Y}{\sum_{y' \in \mathcal{Y}} \widehat{h}_t\left(\widehat{g}_t(X)\right)_{y'}}\right)\right] \tag{2}$$

Invariant representation constraint $\mathcal{L}_{inv}^t$: It aims to minimize the distance between $\widehat{f}_t \sharp P_{D_t^w}^{Z|Y=y}$ and $\widehat{g}_t \sharp P_{D_{t+1}}^{Z|Y=y}$, $\forall y \in \mathcal{Y}$, two conditional distributions induced from domains D_t^w and D_{t+1} using representation mappings $\widehat{f}_t, \widehat{g}_t$, respectively. In other words, for any inputs X from domains D_t^w and X' from D_{t+1} whose labels are the same, we need to find representation mappings $\widehat{f}_t, \widehat{g}_t$ such that the representations $\widehat{f}_t(X), \widehat{g}_t(X')$ have similar distributions. Inspired by correlation alignment loss (Sun & Saenko, 2016), we enforce this constraint by using the following as loss $\mathcal{L}_{inv}^t$:

$$\mathcal{L}_{inv}^t = \sum_{y \in \mathcal{Y}} \frac{1}{4d^2} \left\| C_t^y - C_{t+1}^y \right\|_F^2 \tag{3}$$

where d is the dimension of representation space $\mathcal{Z}$, $\|\cdot\|_F^2$ is the squared matrix Frobenius norm, and C_t^y and C_{t+1}^y are covariance matrices defined as follows:

$$\begin{aligned} C_t^y &= \frac{1}{n_y^t - 1}\left(\widehat{f}_t\left(\mathbf{X}_t^y\right)^\top \widehat{f}_t\left(\mathbf{X}_t^y\right) - \frac{1}{n_y^t}\left(\mathbf{1}^\top \widehat{f}_t\left(\mathbf{X}_t^y\right)\right)^\top \left(\mathbf{1}^\top \widehat{f}_t\left(\mathbf{X}_t^y\right)\right)\right) \\ C_{t+1}^y &= \frac{1}{n_y^{t+1} - 1}\left(\widehat{g}_t\left(\mathbf{X}_{t+1}^y\right)^\top \widehat{g}_t\left(\mathbf{X}_{t+1}^y\right) - \frac{1}{n_y^{t+1}}\left(\mathbf{1}^\top \widehat{g}_t\left(\mathbf{X}_{t+1}^y\right)\right)^\top \left(\mathbf{1}^\top \widehat{g}_t\left(\mathbf{X}_{t+1}^y\right)\right)\right) \end{aligned}$$

Table 1: Prediction performances (i.e., OOD_{Avg} and OOD_{Wrt}) of AIRL and baselines under Eval-D scenario ($K = 5$). We report average results (w. standard deviation) over 5 random seeds. For **CLEAR** dataset, due to only one split between train and test sets, OOD_{Avg} and OOD_{Wrt} are similar.

Algorithm	Circle		Circle-Hard		RMNIST		Yearbook		CLEAR
	OOD_{Avg}	OOD_{Wrt}	OOD_{Avg}	OOD_{Wrt}	OOD_{Avg}	OOD_{Wrt}	OOD_{Avg}	OOD_{Wrt}	OOD_{Avg} / OOD_{Wrt}
ERM	89.63 (0.89)	79.84 (1.84)	66.94 (1.69)	58.43 (0.05)	56.61 (1.83)	51.85 (4.15)	90.79 (0.16)	71.03 (1.74)	69.04 (0.18)
LD	76.60 (6.45)	56.88 (3.74)	58.13 (1.67)	51.58 (1.87)	37.54 (2.77)	25.80 (4.12)	77.10 (0.30)	57.97 (0.88)	57.01 (2.15)
FT	85.57 (1.82)	71.99 (4.11)	59.02 (5.20)	50.80 (2.79)	60.73 (0.87)	47.30 (3.77)	87.04 (0.58)	66.83 (2.22)	66.71 (0.46)
DANN	88.80 (1.17)	78.32 (3.23)	65.10 (0.93)	56.68 (0.59)	58.25 (1.15)	53.61 (1.61)	90.57 (0.22)	69.58 (1.38)	67.48 (1.19)
CDANN	89.75 (0.14)	80.75 (2.97)	64.05 (1.33)	58.68 (0.22)	58.19 (0.93)	54.45 (1.40)	90.46 (0.30)	70.37 (1.44)	66.12 (0.37)
G2DM	89.40 (2.27)	79.61 (2.94)	67.75 (2.69)	59.65 (1.61)	57.62 (0.39)	53.93 (0.31)	87.57 (0.37)	66.69 (1.15)	56.98 (2.77)
CORAL	90.13 (0.52)	83.14 (1.27)	66.12 (1.48)	59.62 (1.17)	51.41 (2.63)	44.95 (3.64)	90.41 (0.20)	69.53 (2.00)	70.96 (1.06)
GROUPDRO	90.50 (1.75)	81.07 (6.12)	67.08 (1.67)	58.51 (0.12)	54.37 (2.98)	46.21 (5.69)	90.65 (0.20)	71.21 (1.51)	70.63 (0.04)
MIXUP	88.49 (0.86)	76.78 (2.49)	63.03 (1.53)	56.21 (1.20)	52.13 (2.54)	34.60 (16.81)	89.75 (0.05)	68.73 (1.36)	69.58 (0.99)
IRM	85.78 (1.11)	74.80 (1.73)	62.43 (2.70)	54.96 (1.78)	26.96 (1.11)	16.25 (1.87)	84.65 (0.31)	64.30 (2.44)	49.54 (1.08)
SELFREG	90.33 (0.14)	82.20 (0.93)	68.23 (2.47)	60.28 (0.90)	50.58 (2.35)	42.15 (4.63)	91.47 (0.12)	73.88 (0.37)	69.18 (0.68)
FISH	90.65 (0.25)	79.09 (2.46)	62.69 (0.63)	56.97 (0.49)	56.53 (1.32)	52.23 (1.47)	89.92 (0.20)	70.58 (0.90)	69.46 (0.47)
EWC	89.18 (1.72)	79.59 (4.63)	68.31 (3.31)	61.34 (2.18)	66.53 (1.26)	50.63 (5.35)	89.47 (0.17)	59.09 (7.70)	45.58 (4.92)
CIDA	87.25 (0.88)	77.91 (0.23)	65.38 (2.77)	58.15 (0.88)	53.42 (4.35)	35.21 (17.85)	91.29 (0.16)	70.19 (1.45)	65.10 (0.12)
DRAIN	86.78 (0.65)	74.57 (1.82)	67.44 (4.65)	57.76 (3.42)	67.09 (4.06)	59.49 (8.31)	89.62 (0.39)	70.36 (2.32)	64.67 (0.65)
DPNET	91.76 (0.16)	**83.35 (1.32)**	64.19 (0.95)	59.94 (0.18)	74.39 (0.23)	71.03 (0.37)	92.11 (0.26)	75.04 (1.16)	64.05 (0.64)
LSSAE	90.21 (1.95)	80.92 (3.53)	66.43 (0.81)	61.22 (0.71)	33.30 (2.14)	18.83 (3.85)	60.48 (4.99)	50.35 (4.67)	22.61 (0.25)
DDA	72.06 (4.51)	48.81 (0.97)	65.26 (3.20)	56.16 (2.45)	**78.18** (0.88)	73.70 (0.31)	86.72 (0.56)	67.60 (2.66)	70.12 (1.10)
AIRL	**92.28 (0.27)**	82.81 (2.70)	**73.50 (2.21)**	**63.29 (1.26)**	77.49 (0.86)	**74.99 (0.57)**	**93.10 (0.21)**	**78.22 (0.92)**	**73.04** (0.67)

where $\mathbf{1}$ is the column vector with all elements equal to 1, $\mathbf{X}_t^y = \{x_i : x_i \in D_t^w, y_i = y\}$ is the matrix whose columns are $\{x_i\}$, $\widehat{f}_t$ and $\widehat{g}_t$ are column-wise operations applied to $\mathbf{X}_t^y$ and $\mathbf{X}_{t+1}^y$, respectively, and n_y^t is cardinality of $\mathbf{X}_t^y$.

5.2 Inference

At the inference stage, the well-trained *representation network* and *classification network* can be used to make predictions about input x from any target domain $D_{T+K},\ K > 0$. In particular, we first map input x to representation z using the encoder Enc in the *representation network* (i.e., g^*_{T+K-1}). Then the *classification network* (i.e., LSTM) is used sequentially to generate h^*_{T+K-1} from the sequence of classifiers $\left[h^*_1, \cdots, h^*_{T+K-2}\right]$, and the prediction about z can be made by h^*_{T+K-1}. Note that at the learning stage, both g^*_{t-1} and f^*_t are used to map input x from domain D_t to the representation space while at the inference stage, only g^*_{T+K-1} is needed for target domain D_{T+K} (we do not use f^*_{T+K} because it requires access to data from all domains $\{D_t\}_{t \le T+K}$ which generally are not available during inference). The complete inference process is shown in Algorithm 2, Appendix B.

6 Experiments

In this section, we present the experimental results of the proposed AIRL and compare AIRL with a wide range of existing algorithms. We evaluate these algorithms on synthetic and real-world datasets. Next, we first introduce the experimental setup and then present the empirical results.

Experimental setup. Datasets and baselines used in the experiments are briefly introduced below. Their details are shown in Appendix C.

Datasets. We consider five datasets: **Circle** (Pesaranghader & Viktor, 2016) (a synthetic dataset containing 30 domains where each instance is sampled from 30 two-dimensional Gaussian distributions), **Circle-Hard** (a synthetic dataset adapted from **Circle** dataset such that domains do not uniformly evolve), **RMNIST** (a semi-synthetic dataset constructed from MNIST (LeCun et al., 1998) by R-degree counterclockwise rotation), **Yearbook** (Ginosar et al., 2015) (a real dataset consisting of frontal-facing American high school yearbook photos from 1930-2013), and **CLEAR** (Lin et al., 2021) (a real dataset capturing the natural temporal evolution of visual concepts that spans a decade).

Baselines. We compare the proposed AIRL with existing methods from related areas, including the followings: empirical risk minimization (ERM), last domain training (LD), fine tuning (FT), domain invariant representation learning (G2DM (Albuquerque et al., 2019), DANN (Ganin et al., 2016), CDANN (Li et al., 2018b), CORAL (Sun & Saenko, 2016), IRM (Arjovsky et al., 2019)), data augmentation (MIXUP (Zhang et al., 2018)), continual learning (EWC (Kirkpatrick et al., 2017)), continuous DA (CIDA (Wang et al., 2020)), distributionally robust optimization (GroupDRO (Sagawa

et al., 2019)), gradient-based DG (Fish (Shi et al., 2022)) contrastive learning-based DG (i.e., SelfReg (Kim et al., 2021)), non-stationary environment DG (DRAIN (Bai et al., 2022), DPNET (Wang et al., 2022), LSSAE (Qin et al., 2022), and DDA (Zeng et al., 2023)). To ensure a fair comparison, we adopt similar architectures for AIRL and baselines, including both representation mapping and classifier. The implementation details are described in Appendix B.

Evaluation method. In the experiments, models are trained on a sequence of source domains $\mathcal{D}_{src}$, and their performance is evaluated on target domains $\mathcal{D}_{tgt}$ under two different scenarios: Eval-S and Eval-D. In the scenario Eval-S, models are trained one time on the first half of domain sequence $\mathcal{D}_{src} = [D_1, D_2, \cdots, D_T]$ and are then deployed to make predictions on the second half of domain sequence $\mathcal{D}_{tgt} = [D_{T+1}, D_{T+2}, \cdots, D_{2T}]$. In the scenario Eval-D, source and target domains are not static but are updated periodically as new data/domain becomes available. For each of these two scenarios, we use two accuracy measures, OOD_{Avg} and OOD_{Wrt}, to evaluate the average- and worst-case performances. Their details are shown in Appendix C. We train each model with 5 different random seeds and report the average prediction performances.

Table 2: Ablation study for AIRL on **Circle-Hard** dataset under Eval-D scenario ($K = 5$).

LSTM	Trans	$\mathcal{L}_{inv}$	OOD_{Avg}	OOD_{Wrt}
✗	✓	✓	69.06	61.05
✓	✗	✓	65.51	58.69
✓	✗	✗	68.33	60.16
✓	✓	✓	**73.50**	**63.29**

Results. Next, we evaluate the model performance under Eval-D scenario. The results for Eval-S scenario are presented in Appendix C.

Non-stationary environment DG results. Performance of AIRL and baselines on synthetic (i.e., **Circle**, **Circle-Hard**) and real-world (i.e., **RMNIST**, **Yearbook**) data are presented in Table 1. We observe that AIRL consistently outperforms other methods over all datasets and metrics. These results indicate that AIRL can effectively capture non-stationary patterns across domains, and such patterns can be leveraged to learn the models that generalize better on target domains compared to the baselines. Among baselines, methods designed specifically for non-stationary environment DG (i.e., DRAIN, DPNET, LSSAE) and continual learning method (i.e., EWC) achieve better performance than other methods. However, such improvement is inconsistent across datasets.

Comparison with existing non-stationary environment DG methods. DPNET assumes that the evolving pattern between two consecutive domains is constant and the distances between them are small. Thus, this method does not achieve good performance for **Circle-Hard** dataset where distance between two consecutive domains is proportional to domain index. DRAIN utilizes Bayesian framework and generates the whole models at every domain. This method, however, is only capable for small neural networks and does not scale well to real-world applications. Moreover, DPNET, DRAIN, and DDA can only generalize to a single subsequent target domain. LSSAE leverages sequential variational auto-encoder (Li & Mandt, 2018) to learn non-stationary pattern. However, this model assumes the availability of aligned data across domain sequence, which may pose challenges to its performance in non-stationary DG. In contrast, AIRL is not limited to the constantly evolving pattern. It is also scalable to large neural networks and can handle multiple target domains. In particular, compared to the base model (ERM), our method has only one extra Transformer and LSTM layers. Note that these layers are used during training only. In the inference stage, predictions are made by Enc and classifier pre-generated by LSTM which then results in a similar inference time with ERM.

Ablation studies. We conduct experiments to investigate the roles of each component in AIRL. In particular, we compare AIRL with its variants; each variant is constructed by removing LSTM (i.e., use fixed classifier instead), Trans (i.e., use fixed representation instead), L_{inv} (i.e., without invariant constraint) from the model. As shown in Table 2, model performance deteriorates when removing any of them. These results validate our theorems and demonstrate the effectiveness of each component.

7 Conclusion

In this paper, we theoretically and empirically studied domain generalization under non-stationary environments. We first established the upper bounds of prediction error on target domains, and then proposed a representation learning-based method that learns adaptive invariant representations across source domains. The resulting models trained with the proposed method can generalize well to unseen target domains. Experiments on both synthetic and real data demonstrate the effectiveness of our proposed method.

REFERENCES

Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*, 2019.

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

Guangji Bai, Ling Chen, and Liang Zhao. Temporal domain generalization with drift-aware dynamic neural network. *arXiv preprint arXiv:2205.10664*, 2022.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. *Advances in neural information processing systems*, 31, 2018.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2018.

Hong-You Chen and Wei-Lun Chao. Gradual domain adaptation without indexed intermediate domains. *Advances in Neural Information Processing Systems*, 34:8201–8214, 2021.

Yining Chen, Colin Wei, Ananya Kumar, and Tengyu Ma. Self-training avoids using spurious features under domain shift. *Advances in Neural Information Processing Systems*, 33:21061–21071, 2020.

Gordon Christie, Neil Fendley, James Wilson, and Ryan Mukherjee. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6172–6180, 2018.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Shiry Ginosar, Kate Rakelly, Sarah Sachs, Brian Yin, and Alexei A Efros. A century of portraits: A visual historical record of american high school yearbooks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1–7, 2015.

Lin Lawrence Guo, Stephen R Pfohl, Jason Fries, Alistair EW Johnson, Jose Posada, Catherine Aftandilian, Nigam Shah, and Lillian Sung. Evaluation of domain generalization and adaptation on improving model robustness to temporal dataset shift in clinical medicine. *Scientific reports*, 12 (1):1–10, 2022.

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*, pp. 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Seogkyu Jeon, Kibeom Hong, Pilhyeon Lee, Jewook Lee, and Hyeran Byun. Feature stylization and domain-aware contrastive learning for domain generalization. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 22–31, 2021.

Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9619–9628, 2021.

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*, 114 (13):3521–3526, 2017.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In *High dimensional probability II*, pp. 443–457. Springer, 2000.

Ananya Kumar, Tengyu Ma, and Percy Liang. Understanding self-training for gradual domain adaptation. In *International Conference on Machine Learning*, pp. 5468–5479. PMLR, 2020.

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.

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*, volume 32, 2018a.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639, 2018b.

Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder. In *International conference on machine learning*. PMLR, 2018.

Zheren Li, Zhiming Cui, Sheng Wang, Yuji Qi, Xi Ouyang, Qitian Chen, Yuezhi Yang, Zhong Xue, Dinggang Shen, and Jie-Zhi Cheng. Domain generalization for mammography detection via multi-style and multi-view contrastive learning. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 98–108. Springer, 2021.

Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 2)*, 2021.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6568–6577, 2019.

A Tuan Nguyen, Toan Tran, Yarin Gal, and Atilim Gunes Baydin. Domain invariant representation learning with domain density transformations. *Advances in Neural Information Processing Systems*, 34, 2021.

Guillermo Ortiz-Jimenez, Mireille El Gheche, Effrosyni Simou, Hermina Petric Maretic, and Pascal Frossard. Cdot: Continuous domain adaptation using optimal transport. *arXiv preprint arXiv:1909.11448*, 2019.

Ali Pesaranghader and Herna L Viktor. Fast hoeffding drift detection method for evolving data streams. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 96–111. Springer, 2016.

Trung Phung, Trung Le, Tung-Long Vuong, Toan Tran, Anh Tran, Hung Bui, and Dinh Phung. On learning domain-invariant representations for transfer learning with multiple sources. *Advances in Neural Information Processing Systems*, 34, 2021.

Fengchun Qiao, Long Zhao, and Xi Peng. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12556–12565, 2020.

Tiexin Qin, Shiqi Wang, and Haoliang Li. Generalizing to evolving domains with latent structure-aware sequential autoencoder. *arXiv preprint arXiv:2205.07649*, 2022.

Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. *arXiv preprint arXiv:2109.02934*, 2021.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

Yuge Shi, Jeffrey Seely, Philip Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *International Conference on Learning Representations*, 2022.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Chris Xing Tian, Haoliang Li, Xiaofei Xie, Yang Liu, and Shiqi Wang. Neuron coverage-guided domain generalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Hao Wang, Hao He, and Dina Katabi. Continuously indexed domain adaptation. *arXiv preprint arXiv:2007.01807*, 2020.

Jingge Wang, Yang Li, Liyan Xie, and Yao Xie. Class-conditioned domain generalization via wasserstein distributional robust optimization. *arXiv preprint arXiv:2109.03676*, 2021.

William Wei Wang, Gezheng Xu, Ruizhi Pu, Jiaqi Li, Fan Zhou, Changjian Shui, Charles Ling, Christian Gagné, and Boyu Wang. Evolving domain generalization. *arXiv preprint arXiv:2206.00047*, 2022.

Qiuhao Zeng, Wei Wang, Fan Zhou, Charles Ling, and Boyu Wang. Foresee what you will learn: Data augmentation for domain generalization in non-stationary environments. In *Proceedings of the AAAI conference on artificial intelligence*, 2023.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *European conference on computer vision*, pp. 561–578. Springer, 2020.

A PROOFS

A.1 PROOF OF THEOREM 4.5

We first introduce the following lemmas which are used to proof Theorem 4.5.

Lemma A.1. *Suppose loss function $\mathcal{L}$ is upper bounded by C. For any classifier $\widehat{h} : \mathcal{X} \to \mathcal{Y}^{\Delta}$ and any hypothesis mechanism $\widehat{M} = \{\widehat{m}_t : \mathcal{X} \times \mathcal{Y} \to \mathcal{X} \times \mathcal{Y}\}$, the expected error of $\widehat{h}$ in an unseen target domain D_{T+1} can be upper bounded:*

$$\epsilon_{D_{T+1}}\left(\widehat{h}\right) \leq \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + CK\sqrt{\mathcal{D}\left(P^{X,Y}_{D_{T+1}}, P^{X,Y}_{\widehat{D}_{T+1}}\right)}$$

where K is a constant dependent on distance metric $\mathcal{D}(\cdot,\cdot)$, $\widehat{D}_{T+1}$ is the domain specified by the push-forward distribution $P^{X,Y}_{\widehat{D}_{T+1}} := \widehat{m}_{T+1}\sharp P^{X,Y}_{D_T}$.

Lemma A.2. *Suppose loss function $\mathcal{L}$ is upper bounded by C. Then, for any $\delta > 0$, with probability at least $1-\delta$ over samples S_n drawn i.i.d from domain D, for all $\widehat{h} \in \mathcal{H}$, the expected error of $\widehat{h}$ in domain D can be upper bounded:*

$$\epsilon_D\left(\widehat{h}\right) \leq \widehat{\epsilon}_D\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}}$$

Proof of Lemma A.1 We first show the proof for KL-divergence. Based on that, the proof for JS-divergence is given.

Let $U = (X,Y)$ and $L(U) = L(\widehat{h}(X), Y)$. We first prove $\int_{\mathcal{E}} \left| P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right| du = \frac{1}{2}\int \left| P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right| du$ where $\mathcal{E}$ is the event that $P^{U=u}_{D_{T+1}} \geq P^{U=u}_{\widehat{D}_{T+1}}$ $(*)$ as follows:

$$\begin{aligned}
\int_{\mathcal{E}} \left| P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right| du &= \int_{\mathcal{E}} \left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right) du \\
&= \int_{\mathcal{E}\cup\overline{\mathcal{E}}} \left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right) du - \int_{\overline{\mathcal{E}}} \left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right) du \\
&\overset{(1)}{=} \int_{\overline{\mathcal{E}}} \left(P^{U=u}_{\widehat{D}_{T+1}} - P^{U=u}_{D_{T+1}} \right) du \\
&= \int_{\overline{\mathcal{E}}} \left| P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right| du \\
&= \frac{1}{2}\int \left| P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right| du
\end{aligned}$$

where $\overline{\mathcal{E}}$ is the complement of $\mathcal{E}$. We have $\overset{(1)}{=}$ because $\int_{\mathcal{E}\cup\overline{\mathcal{E}}} \left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right) du = \int_{\mathcal{U}} \left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}} \right) du = 0$. Then, we have:

$$
\begin{aligned}
\epsilon_{D_{T+1}}\left(\widehat{h}\right) &= \mathbb{E}_{D_{T+1}}\left[L(U)\right] \\
&= \int_{\mathcal{U}} L(u) P_{D_{T+1}}^{U=u} du \\
&= \int_{\mathcal{U}} L(u) P_{\widehat{D}_{T+1}}^{U=u} du + \int_{\mathcal{U}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \\
&= \mathbb{E}_{\widehat{D}_{T+1}}\left[L(U)\right] + \int_{\mathcal{U}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \\
&= \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \int_{\mathcal{E}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du + \int_{\overline{\mathcal{E}}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \\
&\overset{(2)}{\leq} \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \int_{\mathcal{E}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \\
&\overset{(3)}{\leq} \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + C\int_{\mathcal{E}}\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \\
&= \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + C\int_{\mathcal{E}}\left|P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right| du \\
&\overset{(4)}{=} \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{2}\int\left|P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right| du \\
&\overset{(5)}{\leq} \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{2}\sqrt{2\min\left(\mathcal{D}_{KL}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D_{T+1}}^{U}\right), \mathcal{D}_{KL}\left(P_{D_{T+1}}^{U}, P_{\widehat{D}_{T+1}}^{U}\right)\right)} \\
&\leq \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\sqrt{\mathcal{D}_{KL}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D_{T+1}}^{U}\right)} \quad (**)
\end{aligned}
$$

We have $\overset{(2)}{\leq}$ because $\int_{\overline{\mathcal{E}}} L(u)\left(P_{D_{T+1}}^{U=u} - P_{\widehat{D}_{T+1}}^{U=u}\right) du \leq 0$; $\overset{(3)}{\leq}$ because $L(u)$ is non-negative function and is bounded by C; $\overset{(4)}{=}$ by using $(*)$; $\overset{(5)}{\leq}$ by using Pinsker's inequality between total variation norm and KL-divergence. From $(**)$, we can see that when $\mathcal{D}$ is $\mathcal{D}_{KL}$, $K = \frac{1}{\sqrt{2}}$. Next, we give the proof when $\mathcal{D}$ is $\mathcal{D}_{JS}$ (JS-divergence).

Let $P_{D'_{T+1}}^{U} = \frac{1}{2}\left(P_{D_{T+1}}^{U} + P_{\widehat{D}_{T+1}}^{U}\right)$. Apply $(**)$ for two domains D_{T+1} and D'_{T+1}, we have:

$$
\epsilon_{D_{T+1}}\left(\widehat{h}\right) \leq \epsilon_{D'_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\sqrt{\mathcal{D}_{KL}\left(P_{D_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right)} \tag{4}
$$

Apply $(**)$ again for two domains D'_{T+1} and $\widehat{D}_{T+1}$, we have:

$$
\epsilon_{D'_{T+1}}\left(\widehat{h}\right) \leq \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\sqrt{\mathcal{D}_{KL}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right)} \tag{5}
$$

Adding Eq. (4) to Eq. (5) and subtracting $\epsilon_{D'_{T+1}}$, we have:

$$
\begin{aligned}
\epsilon_{D_{T+1}}\left(\widehat{h}\right) &\leq \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\left(\sqrt{\mathcal{D}_{KL}\left(P_{D_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right)} + \sqrt{\mathcal{D}_{KL}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right)}\right) \\
&\overset{(6)}{\leq} \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\sqrt{2\left(\mathcal{D}_{KL}\left(P_{D_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right) + \mathcal{D}_{KL}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D'_{T+1}}^{U}\right)\right)} \\
&= \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \frac{C}{\sqrt{2}}\sqrt{4\mathcal{D}_{JS}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D_{T+1}}^{U}\right)} \\
&= \epsilon_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + \sqrt{2}C\sqrt{\mathcal{D}_{JS}\left(P_{\widehat{D}_{T+1}}^{U}, P_{D_{T+1}}^{U}\right)}
\end{aligned}
$$

We have $\overset{(6)}{\leq}$ by using Cauchy–Schwarz inequality. We can also see that $K = \sqrt{2}$ when $\mathcal{D}$ is $\mathcal{D}_{JS}$.

Proof of Lemma A.2 We start from the Rademacher bound Koltchinskii & Panchenko (2000) which is stated as follows.

Lemma A.3. *Rademacher Bounds. Let $\mathcal{F}$ be a family of functions mapping from Z to $[0,1]$. Then, for any $0<\delta<1$, with probability at least $1-\delta$ over sample $S_n=\{z_1,\cdots,z_n\}$, the following holds for all $f\in\mathcal{F}$:*

$$\mathbb{E}\left[f(z)\right] \leq \frac{1}{n}\sum_{i=1}^{n} f(z_i) + 2\widehat{\mathcal{R}}_n(\mathcal{F}) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}}$$

We then apply Lemma A.3 to our setting with $Z=(X,Y)$, the loss function L bounded by C, and the function class $\mathcal{L}_{\mathcal{H}} = \left\{(x,y)\to L\left(\widehat{h}\left(x\right),y\right) : \widehat{h}\in\mathcal{H}\right\}$. In particular, we scale the loss function L to $[0,1]$ by dividing by C and denote the new class of scaled loss functions as $\mathcal{L}_{\mathcal{H}}/C$. Then, for any $\delta>0$, with probability at least $1-\delta$, we have:

$$\begin{aligned}\frac{\epsilon_D\left(\widehat{h}\right)}{C} &\leq \frac{\epsilon_{\widehat{D}}\left(\widehat{h}\right)}{C} + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}/C\right) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}} \\ &\overset{(1)}{=} \frac{\epsilon_{\widehat{D}}\left(\widehat{h}\right)}{C} + \frac{2}{C}\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2n}} \end{aligned} \tag{6}$$

We have $\overset{(1)}{\leq}$ by using the property of Redamacher complexity that $\widehat{\mathcal{R}}_n(\alpha\mathcal{F}) = \alpha\widehat{\mathcal{R}}_n(\mathcal{F})$. We derive Lemma A.2 by multiplying Eq. (6) by C.

Proof of Theorem 4.5 We then apply Lemmas A.1 and A.2 for the two domains D_{T+1} and $\widehat{D}_{T+1}$. In particular, for any $0<\delta<1$ and any $\widehat{M}=\{\widehat{m}_t : \mathcal{X}\times\mathcal{Y}\to\mathcal{X}\times\mathcal{Y}\}$, with probability at least $1-\delta$ over sample S_n of domain $\widehat{D}_{T+1}$, for all $\widehat{h}:\mathcal{X}\to\mathcal{Y}^{\Delta}$, we have the following inequality :

$$\begin{aligned}\epsilon_{D_{T+1}}\left(\widehat{h}\right) &\leq \widehat{\epsilon}_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} + CK\sqrt{\mathcal{D}\left(P^U_{D_{T+1}}, P^U_{\widehat{D}_{T+1}}\right)} \\ &= \widehat{\epsilon}_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} \\ &+ CK\sqrt{\mathcal{D}\left(P^U_{D_{T+1}}, P^U_{\widehat{D}_{T+1}}\right) - \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^U_{D_t}, P^U_{\widehat{D}_t}\right) + \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^U_{D_t}, P^U_{\widehat{D}_t}\right)} \\ &\leq \widehat{\epsilon}_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} \\ &+ CK\sqrt{\left|\mathcal{D}\left(P^U_{D_{T+1}}, P^U_{\widehat{D}_{T+1}}\right) - \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^U_{D_t}, P^U_{\widehat{D}_t}\right)\right| + \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^U_{D_t}, P^U_{\widehat{D}_t}\right)} \\ &= \widehat{\epsilon}_{\widehat{D}_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} \\ &+ CK\sqrt{\Phi\left(\widehat{M}\right) + \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}_t}\right)} \end{aligned} \tag{7}$$

Applying Eq. (7) for M^*, we have the following inequality with probability at least $1-\delta$:

$$\begin{aligned}\epsilon_{D_{T+1}}\left(\widehat{h}\right) &\leq \widehat{\epsilon}_{D^*_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} \\ &+ CK\sqrt{\Phi\left(M^*\right) + \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{D^*_t}\right)} \end{aligned} \tag{8}$$

According Assumption 4.3, we also have the following inequality with probability at least $1-\delta$:

$$\Phi\left(M^*\right) \leq \epsilon \tag{9}$$

Using union bound for Eq. (8) and Eq. (9), we also have the following inequality with probability at least $1-2\delta$:

$$\epsilon_{D_{T+1}}\left(\widehat{h}\right) \leq \widehat{\epsilon}_{D^*_{T+1}}\left(\widehat{h}\right) + 2\widehat{\mathcal{R}}_n\left(\mathcal{L}_{\mathcal{H}}\right) + 3C\sqrt{\frac{\log(2/\delta)}{2n}} + CK\sqrt{\epsilon + \frac{1}{T-1}\sum_{t=2}^{T}\mathcal{D}\left(P^{X,Y}_{D_t}, P^{X,Y}_{D^*_t}\right)} \tag{10}$$

Replacing δ by $\frac{\delta}{2}$ in Eq. (10) gives us the inequality in Corollary 4.5.

A.2 Proof of Proposition 4.6

$\forall y \in \mathcal{Y}$, we have the following $(*)$:

$$\begin{aligned}
P^{Y=y}_{D^w_{t-1}} &= \int_{\mathcal{X}} P^{X=x,Y=y}_{D^w_{t-1}} dx \\
&= \int_{\mathcal{X}} w_y \times P^{X=x,Y=y}_{D_{t-1}} dx \\
&= \int_{\mathcal{X}} \frac{P^{Y=y}_{D_t}}{P^{Y=y}_{D_{t-1}}} \times P^{X=x,Y=y}_{D_{t-1}} dx \\
&= P^{Y=y}_{D_t} \int_{\mathcal{X}} P^{X=x|Y=y}_{D_{t-1}} dx \\
&= P^{Y=y}_{D_t}
\end{aligned}$$

Then we show the proof for $\mathcal{D}_{KL}$. Based on that, the proof for $\mathcal{D}_{JS}$ is given. We have:

$$\begin{aligned}
\mathcal{D}_{KL}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}^w_t}\right) &= \mathbb{E}_{P^{X,Y}_{D_t}}\left[\log P^{X,Y}_{D_t} - \log P^{X,Y}_{\widehat{D}^w_t}\right] \\
&= \mathbb{E}_{P^{X,Y}_{D_t}}\left[\log P^{Y}_{D_t} + \log P^{X|Y}_{D_t}\right] - \mathbb{E}_{P^{X,Y}_{D_t}}\left[\log P^{Y}_{\widehat{D}^w_t} + \log P^{X|Y}_{\widehat{D}^w_t}\right] \\
&= \mathbb{E}_{P^{X,Y}_{D_t}}\left[\log P^{Y}_{D_t} - \log P^{Y}_{\widehat{D}^w_t}\right] + \mathbb{E}_{P^{X,Y}_{D_t}}\left[\log P^{X|Y}_{D_t} - \log P^{X|Y}_{\widehat{D}^w_t}\right] \\
&\overset{(1)}{=} \mathbb{E}_{P^{Y}_{D_t}}\left[\mathbb{E}_{P^{X|Y}_{D_t}}\left[\log P^{X|Y}_{D_t} - \log P^{X|Y}_{\widehat{D}^w_t}\right]\right] \\
&= \mathbb{E}_{P^{Y}_{D_t}}\left[\mathcal{D}_{KL}\left(P^{X|Y}_{D_t}, P^{X|Y}_{\widehat{D}^w_t}\right)\right]
\end{aligned} \tag{11}$$

We have $\overset{(1)}{=}$ because $P^Y_{\widehat{D}^w_t} = \widehat{m}_{t-1}\sharp P^Y_{D^w_{t-1}} = P^Y_{D^w_{t-1}}$ for $m^*_{t-1}: \mathcal{X} \to \mathcal{X}$ and $P^Y_{D^w_{t-1}} = P^Y_{D_t}$ by $(*)$.
For JS-divergence $\mathcal{D}_{JS}$, let $P^{X,Y}_{D'_t} = \frac{1}{2}\left(P^{X,Y}_{D_t} + P^{X,Y}_{\widehat{D}^w_t}\right)$. Then, we have:

$$\begin{aligned}
&\mathcal{D}_{JS}\left(P^{X,Y}_{D_t}, P^{X,Y}_{\widehat{D}^w_t}\right) \\
&= \frac{1}{2}\mathcal{D}_{KL}\left(P^{X,Y}_{D_t}, P^{X,Y}_{D'_t}\right) + \frac{1}{2}\mathcal{D}_{KL}\left(P^{X,Y}_{\widehat{D}^w_t}, P^{X,Y}_{D'_t}\right) \\
&\overset{(2)}{=} \frac{1}{2}\left(\mathbb{E}_{P^{Y}_{D_t}}\left[\mathcal{D}_{KL}\left(P^{X|Y}_{D_t}, P^{X|Y}_{D'_t}\right)\right] + \mathbb{E}_{P^{Y}_{\widehat{D}^w_t}}\left[\mathcal{D}_{KL}\left(P^{X|Y}_{\widehat{D}^w_t}, P^{X|Y}_{D'_t}\right)\right]\right) \\
&= \mathbb{E}_{P^{Y}_{D_t}}\left[\frac{1}{2}\left(\mathcal{D}_{KL}\left(P^{X|Y}_{D_t}, P^{X|Y}_{D'_t}\right) + \mathcal{D}_{KL}\left(P^{X|Y}_{\widehat{D}^w_t}, P^{X|Y}_{D'_t}\right)\right)\right] \\
&= \mathbb{E}_{P^{Y}_{D_t}}\left[D_{JS}\left(P^{X|Y}_{D_t}, P^{X|Y}_{\widehat{D}^w_t}\right)\right]
\end{aligned}$$

We have $\overset{(2)}{=}$ by applying Eq. (11) for $\mathcal{D}_{KL}\left(P^{X,Y}_{D_t}, P^{X,Y}_{D'_t}\right)$ and $\mathcal{D}_{KL}\left(P^{X,Y}_{\widehat{D}^w_t}, P^{X,Y}_{D'_t}\right)$.

B MODEL DETAILS

B.1 PSEUDO CODES FOR AIRL'S LEARNING AND INFERENCE PROCESSES

Algorithm 1: Learning process for AIRL

Input: Training datasets from T source domains $\{D_t\}_{t=1}^T$, *representation network* = {Enc, Trans}, *classification network* = $\{\text{LSTM}, \widehat{h}_1\}$, α, n
Output: Trained Enc, Trans, LSTM, h_1^*

$L_{inv} = 0, L_{cls} = 0$
/* Estimate $\{w_y^t\}_{y\in\mathcal{Y},t<T}$ for important weighting */
for $t = 1 : T-1$ **do**
  for $y \in \mathcal{Y}$ **do**
    $w_y^t = P_{D_{t+1}}^{Y=y} / P_{D_t}^{Y=y}$
/* Learn weights for Enc, Trans, LSTM */
while *learning is not end* **do**
  Sample batch $\mathcal{B} = \{x_t, y_t\}_{t=1}^T \sim \{D_t\}_{t=1}^T$ where $\{x_t, y_t\} = \left\{x_t^j, y_t^j\right\}_{j=1}^n$
  $z_1 = \text{Enc}(x_1)$
  for $t = 1 : T-1$ **do**
    $z_{t+1} = \text{Enc}(x_{t+1})$
    $\widehat{z}_t = \text{Trans}(z_{\leq t})$
    $\{\widehat{z}_t(w), y_t(w)\} = \text{Reweight } \{\widehat{z}_t, y_t\} \text{ with } w^t = \{w_y^t\}_{y\in\mathcal{Y}}$
    Calculate L_{inv}^t from $\widehat{z}_t(w), z_{t+1}$ by Eq. (3)
    $L_{inv} = L_{inv} + L_{inv}^t$
    if $t > 1$ **then**
      $h_t = \text{LSTM}(h_{<t})$
    Calculate L_{cls}^t from $y_t(w), y_{t+1}, \widehat{h}_t(\widehat{z}_t(w)), \widehat{h}_t(z_{t+1})$ by Eq. (2)
    $L_{cls} = L_{cls} + L_{cls}^t$
  Update Enc, Trans, LSTM, $\widehat{h}_1$ by optimizing $L_{inv} + \alpha L_{cls}$

Algorithm 2: Inference process for AIRL

Input: Testing dataset from domain D_{T+K}, trained Enc, LSTM, h_1^*
Output: Predictions for testing dataset

for $t = 2 : (T+K-1)$ **do**
  $h_t^* = \text{LSTM}(h_{<t}^*)$
while *inference is not end* **do**
  Sample batch $\mathcal{B} = x_{T+K} \sim D_{T+K}$
  $z_{T+K} = \text{Enc}(x_{T+K})$
  Generate predictions $h_{T+K-1}^*(z_{T+K})$

B.2 DETAILS OF MODEL ARCHITECTURES

Our proposed model AIRL consists of three components: (i) encoder Enc that maps inputs to representation (i.e., equivalent to $\widehat{g}_t$ in our theoretical results), (ii) transformer layer Trans that helps to enforce the invariant representation (i.e., Enc + Trans equivalent to $\widehat{f}_t$ in our theoretical results), and (iii) classification network LSTM that generates classifiers mapping representations to the output space. At each target domain, LSTM layer is used to generate the new classifier based on the sequences of previous classifiers. The detailed architectures of these networks used in our experiment are presented in Tables 3 and 4 below.

Table 3: Detailed architecture of AIRL for **RMNIST** (**n_channel** = 1, **n_output** = 10), **Yearbook** (**n_channel** = 3, **n_output** = 1), and **CLEAR** (**n_channel** = 3, **n_output** = 10) datasets.

Networks	Layers
Representation Mapping G	Conv2d(input channel = **n_channel**, output channel = 32, kernel = 3, padding = 1)
	BatchNorm2d
	ReLU
	MaxPool2d
	Conv2d(input channel = 32, output channel = 32, kernel = 3, padding = 1)
	BatchNorm2d
	ReLU
	MaxPool2d
	Conv2d(input channel = 32, output channel = 32, kernel = 3, padding = 1)
	BatchNorm2d
	ReLU
	MaxPool2d
	Conv2d(input channel = 32, output channel = 32, kernel = 3, padding = 1)
	BatchNorm2d
	ReLU
	MaxPool2d
Transformer Trans	Q: Linear(input dim = 32, output dim = 32)
	K: Linear(input dim = 32, output dim = 32)
	V: Linear(input dim = 32, output dim = 32)
	U: Linear(input dim = 32, output dim = 32)
	Linear(input dim = 32, output dim = 32)
	Batchnorm1d
	LeakyReLU
Classification Network LSTM	Linear(input dim = (32 * 32 + 32) + (32 * **n_output** + **n_output**), output dim = 128)
	LSTM(input dim = 128, output dim = 128)
	Linear(input dim = 128, output dim = (32 * 32 + 32) + (32 * **n_output** + **n_output**))
$\widehat{h}_t$ (Output of LSTM)	Linear(input dim = 32, output dim = 32)
	ReLU
	Linear(input dim = 32, output dim = **n_output**)

Table 4: Detailed architecture of AIRL for **Circle** and **Circle-Hard** datasets.

Networks	Layers
Encoder Enc	Linear(input dim = 2, output dim = 32)
	ReLU
	Linear(input dim = 32, output dim = 32)
	ReLU
	Linear(input dim = 32, output dim = 32)
	ReLU
	Linear(input dim = 32, output dim = 32)
Transformer Trans	Q: Linear(input dim = 32, output dim = 32)
	K: Linear(input dim = 32, output dim = 32)
	V: Linear(input dim = 32, output dim = 32)
	U: Linear(input dim = 32, output dim = 32)
	Linear(input dim = 32, output dim = 32)
	Batchnorm1d
	LeakyReLU
Classification Network LSTM	Linear(input dim = (32 * 32 + 32) + (32 * 1 + 1), output dim = 128)
	LSTM(input dim = 128, output dim = 128)
	Linear(input dim = 128, output dim = (32 * 32 + 32) + (32 * 1 + 1))
$\widehat{h}_t$ (Output of LSTM)	Linear(input dim = 32, output dim = 32)
	ReLU
	Linear(input dim = 32, output dim = 1)

C Details of Experimental Setup and Additional Results

C.1 Experimental Setup

Datasets. Our experiments are conducted on two synthetic and two real-world datasets. The data statistics of these datasets are presented in Table 5. For Eval-S scenario, the first half of domains in the domain sequences are used for training and the following domains are used for testing. For Eval-D scenario, we vary the size of the training set starting from the first half of domains by sequentially adding new domains to this set. In both scenarios, we split the training set into smaller subsets with a ratio $81:9:10$; these subsets are used as training, validation, and in-distribution testing sets. The data descriptions are given as follow:

- **Circle** (Pesaranghader & Viktor, 2016): A synthetic dataset containing 30 domains. Features $X := [X_1, X_2]^T$ in domain t are two-dimensional and Gaussian distributed with mean $\bar{X}^t = [r\cos(\pi t/30), r\sin(\pi t/30)]$ where r is radius of semicircle; the distributions of different domains have the same covariance matrix but different means that uniformly evolve from right to left on a semicircle. Binary label Y are generated based on labeling function $Y = \mathbb{1}\left[(X_1 - x_1^o)^2 + (X_2 - x_2^o)^2 \leq r\right]$, where (x_1^o, x_2^o) are center of semicircle. Models trained on the right part are evaluated on the left part of the semicircle.
- **Circle-Hard**: A synthetic dataset adapted from **Circle** dataset, where mean $\bar{X}^t$ does not uniformly evolve. Instead, $\bar{X}^t = [r\cos(\theta_t), r\sin(\theta_t)]$ where $\theta_t = \theta_{t-1} + \pi(t-1)/180$ and $\theta_1 = 0\,\text{rad}$.
- **RMNIST**: A dataset constructed from MNIST (LeCun et al., 1998) by R-degree counter-clockwise rotation. We evenly select 30 rotation angles R from 0° to 180° with step size 6°; each angle corresponds to a domain. The domains with $R \leq r$ are considered source domains, those with $R > r$ are the target domains used for evaluation. In this dataset, the goal is to train a multi-class classifier on source domains that predicts the digits of images in target.
- **Yearbook** (Ginosar et al., 2015): A real dataset consisting of frontal-facing American high school yearbook photos from 1930-2013. Due to the evolution of fashion, social norms, and population demographics, the distribution of facial images changes over time. In this dataset, we aim to train a binary classifier using historical data to predict the genders of images in the future.
- **CLEAR** (Lin et al., 2021): A real dataset built from existing large-scale image collections (YFCC100M) which captures the natural temporal evolution of visual concepts in the real world that spans a decade (2004-2014). In this dataset, we aim to train a multi-class classifier using historical data to predict 10 object types in future images.

Table 5: Data statistics.

	Data type	Label type	#instance	#domain
Circle	Synthetic	Binary	30000	30
Circle-Hard	Synthetic	Binary	30000	30
RMNIST	Semi-synthetic	Multi	30000	30
Yearbook	Real-world	Binary	33431	84
CLEAR	Real-world	Multi	29747	10

Baseline methods. We compare the proposed `AIRL` with existing methods from related areas, including the followings:

- Empirical risk minimization (`ERM`): A simple method that considers all source domains as one domain.
- Last domain (`LD`): A method that only trains model using the most recent source domain.
- Fine tuning (`FT`): The baseline trained on all source domains in a sequential manner.
- Domain invariant representation learning: Methods that learn the invariant representations across source domains and train a model based on the representations. We experiment with

G2DM (Albuquerque et al., 2019), DANN (Ganin et al., 2016), CDANN (Li et al., 2018b), CORAL (Sun & Saenko, 2016), IRM (Arjovsky et al., 2019).

- Data augmentation: We experiment with MIXUP (Zhang et al., 2018) that generates new data using convex combinations of source domains to enhance the generalization capability of models.
- Continual learning: We experiment with EWC (Kirkpatrick et al., 2017), method that learns model from data streams that overcomes catastrophic forgetting issue.
- Continuous domain adaptation: We experiment with CIDA (Wang et al., 2020), an adversarial learning method designed for DA with continuous domain labels.
- Distributionally robust optimization: We experiment with GROUPDRO (Sagawa et al., 2019) that minimizes the worst-case training loss over pre-defined groups through regularization.
- Gradient-based DG: We experiment with FISH (Shi et al., 2022) that targets domain generalization by maximizing the inner product between gradients from different domains.
- Contrastive learning-based DG: We experiment with SELFREG (Kim et al., 2021) that utilizes the self-supervised contrastive losses to learn domain-invariant representation by mapping the latent representation of the same-class samples close together.
- Non-stationary environment DG: We experiment with DRAIN (Bai et al., 2022), DPNET (Wang et al., 2022), LSSAE (Qin et al., 2022). and DDA (Zeng et al., 2023). DRAIN, DPNET, and DDA focus on domain D_{T+1} only so we use the same model when making predictions for all target domains $\{D_t\}_{t>T}$.

Evaluation method. In the experiments, models are trained on a sequence of source domains $\mathcal{D}_{src}$, and their performance is evaluated on target domains $\mathcal{D}_{tgt}$ under two different scenarios: Eval-S and Eval-D.

In the scenario Eval-S, models are trained one time on the first half of domain sequence $\mathcal{D}_{src} = [D_1, D_2, \cdots, D_T]$ and are then deployed to make predictions on the next K domains in the second half of domain sequence $\mathcal{D}_{tgt} = [D_{T+1}, D_{T+2}, \cdots, D_{T+K}]$ $(T + 1 \leq K \leq 2T)$. The average and worst-case performances can be evaluated using two matrices $\mathrm{OOD}_{\mathrm{Avg}}$ and $\mathrm{OOD}_{\mathrm{Wrt}}$ defined below.

$$\mathrm{OOD}_{\mathrm{Avg}} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{acc}_{T+k}; \quad \mathrm{OOD}_{\mathrm{Wrt}} = \min_{k\in[K]} \mathrm{acc}_{T+k}$$

where acc_{T+k} denotes the accuracy of model on target domain D_{T+k}.

In the scenario Eval-D, source and target domains are not static but are updated periodically as new data/domain becomes available. This allows us to update models based on new source domains. Specifically, at time step $t \in [T, 2T - K]$, models are updated on source domains $\mathcal{D}_{src} = [D_1, D_2, \cdots, D_t]$ and are used to predict target domains $\mathcal{D}_{tgt} = [D_{t+1}, D_{t+2}, \cdots, D_{t+K}]$. The average and worst-case performances of models in this scenario can be defined as follows.

$$\mathrm{OOD}_{\mathrm{Avg}} = \tfrac{1}{(T-K+1)K} \textstyle\sum_{t=T}^{2T-K} \sum_{k=1}^{K} \mathrm{acc}_{t+k}$$
$$\mathrm{OOD}_{\mathrm{Wrt}} = \min_{t\in[T,2T-K]} \tfrac{1}{K} \textstyle\sum_{k=1}^{K} \mathrm{acc}_{t+k}$$

In our experiment, the time step t starts from the index denoting half of the domain sequence.

Implementation and training details. Data, model implementation, and training script are included in the supplementary material. We train each model on each setting with 5 different random seeds and report the average prediction performances. All experiments are conducted on a machine with 24-Core CPU, 4 RTX A4000 GPUs, and 128G RAM.

C.2 Additional experiment results

Performance gap between in-distribution and out-of-distribution predictions. This study is motivated based on the assumption that the environment changes over time and that there exist distribution shifts between training and test data. To verify this assumption in our datasets, we

Table 6: Performances of DANN on **RMNIST** dataset.

Target Domain	0°-rotated	15°-rotated	30°-rotated	45°-rotated	60°-rotated
Model Performance	51.2	59.1	70.0	69.2	53.9

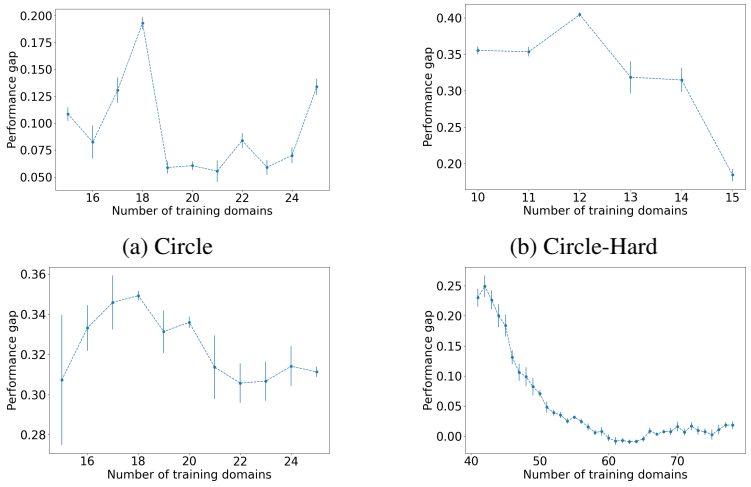
compare the performances of ERM on in-distribution and out-of-distribution testing sets. Specifically, we show the gaps between the performances of ERM measured on the in-distribution (i.e., $\mathrm{ID}_{\mathrm{Avg}}$) and out-of-distribution (i.e., $\mathrm{OOD}_{\mathrm{Avg}}$) testing sets under Eval-D scenario (i.e., $K = 5$) in Figure 4.

Performance of fixed invariant representation learning in conventional and non-stationary DG settings. A key distinction from non-stationary DG is that the model evolves over the domain sequence to capture non-stationary patterns (i.e., learn invariant representations between two consecutive domains but adaptive across domain sequence). This stands in contrast to the conventional DG (Ganin et al., 2016; Phung et al., 2021) which relies on an assumption that target domains lie on or are near the mixture of source domains, then enforcing fixed invariant representations across all source domains can help to generalize the model to target domains. We argue that this assumption may not hold in non-stationary DG where the target domains may be far from the mixture of source domains resulting in the failure of the existing methods.

To verify this argument, we conduct an experiment on rotated **RMNIST** dataset with DANN (Ganin et al., 2016) – a model that learns fixed invariant representations across all domains. Specifically, we create 5 domains by rotating images by 0, 15, 30, 45, and 60 degrees, respectively, and follow leave-one-out evaluation (i.e., one domain is target while the remaining domains are source). Clearly, the setting where the target domain are images rotated by 0 or 60 degrees can be considered as non-stationary domain generalization while other settings can be considered as conventional domain generalization. The performances of DANN with different target domains are shown in Table 6. As we can see, the accuracy drops significantly when the target domain are images rotated by 0 or 60 degrees. This result demonstrates that learning fixed invariant representations across all domains is not suitable for non-stationary DG.

Experimental results for Eval-S scenario. The prediction performances of AIRL and baselines on synthetic (i.e., Circle, Circle-Hard) and real-world (i.e., RMNIST, Yearbook) data under Eval-S scenario are presented in Figure 5 below. In this scenario, the training set is fixed as the first half of domains while the testing set is varied from the five subsequent domains to the second half of domains in the domain sequences. We report averaged results with error bars (std) for training over 5 different random seeds.

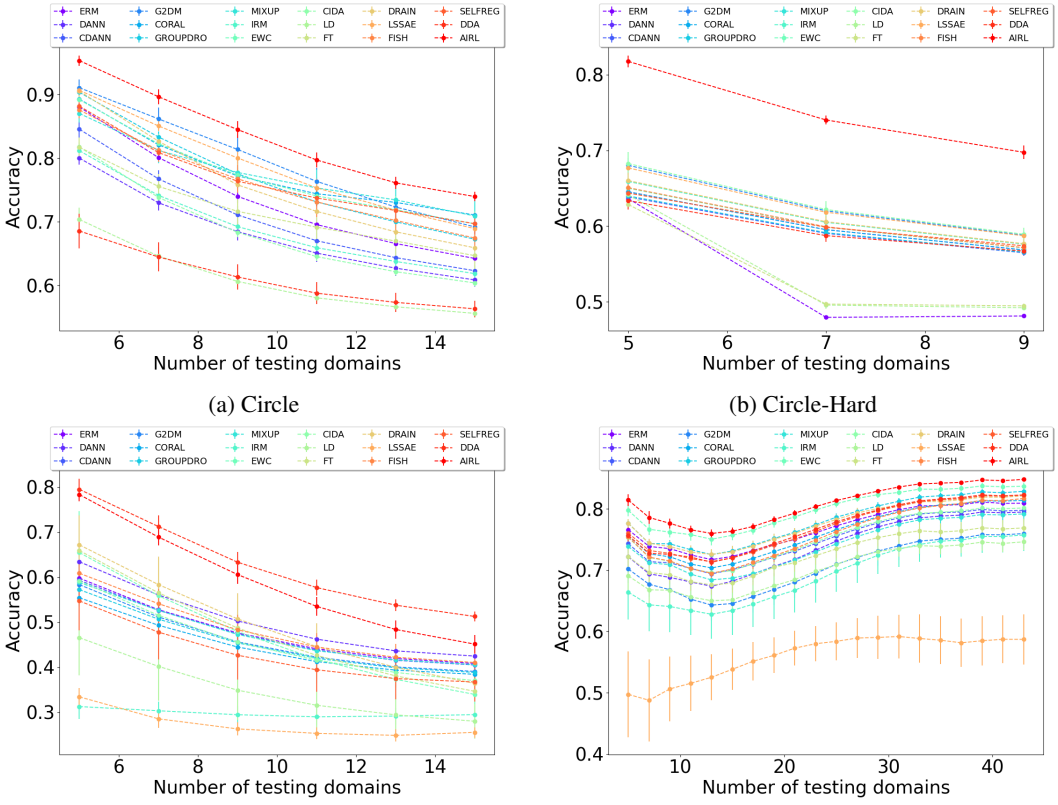
We can see that AIRL consistently outperforms baselines in most datasets. We also observe that the prediction performances decreases when the predictions are made for the distant target domains (i.e., the number of testing domain increases) for all models in **Circle**, **Circle-Hard**, and **RMNIST** datasets. This pattern is reasonable because domains in these datasets are generated monotonically. For **Yearbook** dataset, the performance curves are U-shaped that they decrease first but increase later. This dataset is from a real-world environment so we expect the shapes of the curves are more complex compared to those in the other datasets.



(a) Circle (b) Circle-Hard

(c) RMNIST (d) Yearbook

Figure 4: Gaps between the performances of ERM measured on the in-distribution and out-of-distribution testing sets (i.e., $\mathrm{ID}_{\mathrm{Avg}} - \mathrm{OOD}_{\mathrm{Avg}}$) under Eval-D scenario (i.e., $K = 5$). This experiment is conducted on **Circle**, **Circle-Hard**, **RMNIST**, and **Yearbook** datasets.



(a) Circle (b) Circle-Hard

(c) RMNIST (d) Yearbook

Figure 5: Prediction performances (i.e., $\mathrm{OOD}_{\mathrm{Avg}}$) of AIRL and baselines under Eval-S scenario. The training set is fixed as the first half of domains while the testing set is varied from the five subsequent domains to the second half of domains in the domain sequences. We report average results for training over 5 different random seeds. This experiment is conducted on **Circle**, **Circle-Hard**, **RMNIST**, and **Yearbook** datasets.