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:

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

We have $\overset{(2)}{\leq}$ because $\int_{\overline{\mathcal{E}}} L(u)\left(P^{U=u}_{D_{T+1}} - P^{U=u}_{\widehat{D}_{T+1}}\right) du \leq 0$; $\overset{(3)}{\leq}$ 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 $P^U_{D'_{T+1}} = \frac{1}{2}\left(P^U_{D_{T+1}} + P^U_{\widehat{D}_{T+1}}\right)$. Apply $(**)$ for two domains D_{T+1} and D'_{T+1}, we have:

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

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

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

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

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

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 $1-\delta$:

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

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

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

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 = {LSTM, $\widehat{h}_1$}, $\alpha$, $n$
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 $\mathrm{Enc}, \mathrm{Trans}, \mathrm{LSTM}$ */
while learning is not end do
    Sample batch $\mathcal{B} = \{x_t, y_t\}_{t=1}^{T} \sim \{D_t\}_{t=1}^{T}$ where $\{x_t, y_t\} = \left\{x_t^j, y_t^j\right\}_{j=1}^{n}$
    $z_1 = \mathrm{Enc}\left(x_1\right)$
    for $t = 1 : T-1$ do
        $z_{t+1} = \mathrm{Enc}\left(x_{t+1}\right)$
        $\widehat{z}_t = \mathrm{Trans}\left(z_{\leq t}\right)$
        $\{\widehat{z}_t(w), y_t(w)\}$ = Reweight $\{\widehat{z}_t, y_t\}$ 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
            $h_t = \mathrm{LSTM}\left(h_{<t}\right)$
        Calculate $L_{cls}^t$ from $y_t(w), y_{t+1}, \widehat{h}_t\left(\widehat{z}_t(w)\right), \widehat{h}_t\left(z_{t+1}\right)$ by Eq. (2)
        $L_{cls} = L_{cls} + L_{cls}^t$
    Update $\mathrm{Enc}, \mathrm{Trans}, \mathrm{LSTM}, \widehat{h}_1$ by optimizing $L_{inv} + \alpha L_{cls}$
```

Algorithm 2: Inference process for AIRL

```
Input: Testing dataset from domain $D_{T+K}$, trained $\mathrm{Enc}, \mathrm{LSTM}, h_1^*$
Output: Predictions for testing dataset
for $t = 2 : (T+K-1)$ do
    $h_t^* = \mathrm{LSTM}\left(h_{<t}^*\right)$
while inference is not end do
    Sample batch $\mathcal{B} = x_{T+K} \sim D_{T+K}$
    $z_{T+K} = \mathrm{Enc}\left(x_{T+K}\right)$
    Generate predictions $h_{T+K-1}^*\left(z_{T+K}\right)$
```

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 $Y = \mathbb{1}\left[(X_1 - x_1^o)^2 + (X_2 - x_2^o)^2 \le r\right]$, 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 $R \le r$ 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

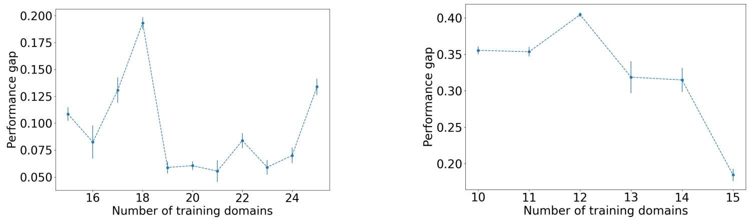
compare the performances of `ERM` on in-distribution and out-of-distribution testing sets. Specifically, we show the gaps between the performances of `ERM` measured on the in-distribution (i.e., $\mathrm{ID}_{\mathrm{Avg}}$) and out-of-distribution (i.e., $\mathrm{OOD}_{\mathrm{Avg}}$) testing sets under 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.

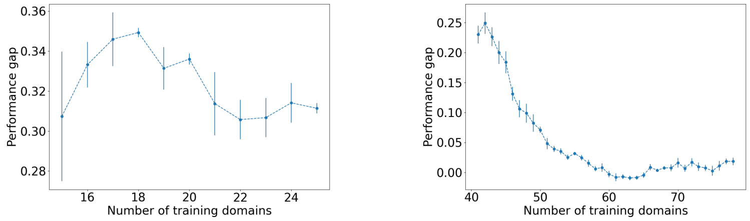
To verify this argument, we conduct an experiment on rotated **RMNIST** dataset with `DANN` (Ganin et al., 2016) – a model that learns fixed invariant representations across all domains. Specifically, we create 5 domains by rotating images by 0, 15, 30, 45, and 60 degrees, respectively, and follow leave-one-out 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.

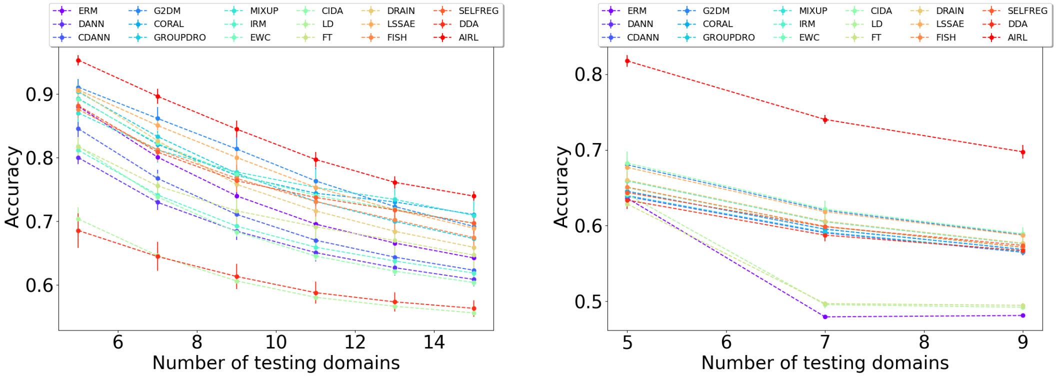


(a) Circle (b) Circle-Hard

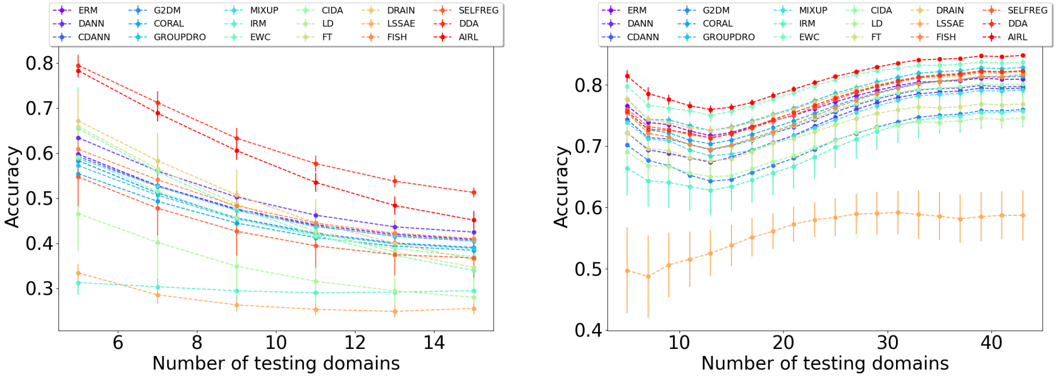


(c) RMNIST (d) Yearbook

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.