Invariant Risk Minimization Games

GUNA SEKHAR SAI HARSHA LAGUDU, *ASU ID: 1223118858* and YUQI WANG, *ASU ID: 1222500435*

1 ABSTRACT

When working in contexts where the test distributions differ from the training distribution owing to spurious correlations, the typical risk minimization paradigm of machine learning becomes fragile. By concentrating models on variables that have a causal link with the result, training on data from several contexts and identifying invariant predictors decreases the effect of false features. In this project, we define invariant risk minimization as determining the Nash equilibrium of an ensemble game over many environments. By doing so, we provide a simple training procedure that employs the optimal response dynamics and, in our studies, achieves comparable or greater empirical accuracy with considerably lower variance than the difficult bi-level optimization problem of [1].

Additional Key Words and Phrases: Spurious coorelations, Invariant Predictors, Nash Equilibrium, Best Response Dynamics

ACM Reference Format:
Guna Sekhar Sai Harsha Lagudu, *ASU ID: 1223118858* and Yuqi Wang, *ASU ID: 1222500435*. 2022. Invariant Risk Minimization Games. 1, 1 (May 2022), 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 INTRODUCTION

There are several humiliating examples of false correlations that fail to hold outside of a certain training (and identically distributed test) distribution in the history of machine learning. The authors of [2] built a convolutional neural network (CNN) to distinguish between camels and cows. The training dataset had one source of bias: most of the images of cows were taken in verdant meadows, but most pictures of camels were taken in deserts. CNN caught up on the erroneous association, i.e., it connected green fields with cows and failed to accurately categorize images of cows on sandy beaches. In another situation, a neural network utilized a brake light indication to continue applying brakes, despite the fact that there was a fake connection in the training data [3]; the list of similar cases is endless.

To handle the problem of models inheriting spurious correlations, the authors of [1] demonstrate how to build robust predictors by utilizing the variable degrees of spurious correlation inherently present in data acquired from multiple data sources. The authors suggest developing a representation Φ in which the optimum classifier supplied is invariant across training contexts. This approach results in a challenging bi-level optimization problem, which the authors alleviate by fixing a basic linear classifier and learning a representation Φ such that the classifier is "roughly locally optimum" in all training contexts.

We adopt a unique strategy in this project. We design a classifier ensemble, with each environment affecting one of the ensemble's components. To produce predictions, each environment utilizes the full ensemble. We made all of the environments play a game in which they had to decide on their contribution to the ensemble while minimizing their risk. Surprisingly, we show that the set of invariant predictors across training environments

Authors' address: Guna Sekhar Sai Harsha Lagudu, *ASU ID: 1223118858*, glagudu@asu.edu; Yuqi Wang, *ASU ID: 1222500435*.

equals the set of predictors that solve the ensemble game; this conclusion applies for a vast class of non-linear classifiers.

This raises the question of how to solve the game. We employ traditional best response dynamics [4], which is relatively easy to implement. Every environment takes it's turns and moves its classifier in the direction that minimizes the risk specific to its environment on a regular basis. On numerous datasets, we show that the invariant predictors discovered by our method lead to higher or equivalent performance with a smaller standard deviation than [1]. Our technique has the advantage of not requiring classifiers to be linear, which [1] underlined as an interesting topic for future research.

In general, we believe that the game-theoretic viewpoint presented here can lead to a completely new paradigm for addressing the problem of invariance.

3 INVARIANT RISK MINIMIZATION

We describe [1] invariant risk minimization (IRM). Consider datasets $\{(x_i^e, y_i^e)\}_{i=1}^{n_e}$ from various training settings $e \in \epsilon_{tr}$. The feature value $x_i^e \in X$ and the associated labels $y_i^e \in Y$, where $X \subseteq \mathbb{R}^n$ and $Y \subseteq \mathbb{R}^k$. Define a predictor for $f : X \to \mathbb{R}^k$.

The objective of IRM is to use these numerous datasets to build a predictor f that performs well over a wide range of previously unexplored environments ϵ_{all}. Define f's risk in environment e as $R^e(f) = \mathbb{E}_{X^e,Y^e}[l(f(X^e);Y^e)]$, where l is the loss when $f(X)$ is the predicted value and Y is the associated label. It is not necessary to assume that f translates to real values; for example, in a k-class classification, the output of the function f is the score for each class, which may be transformed into a hard label by picking the class with the highest score.

Invariant predictor : We claim that a data representation: $\Phi : X \to Z \subseteq \mathbb{R}^d$ triggers an invariant predictor $w \circ \Phi$ across environments $e \in \epsilon$ if there is a classifier $w : Z \to \mathbb{R}^k$ that achieves the least risk for all environments $w \in argmin_{\bar{w} \in H_w} R^e(\bar{w} \circ \Phi)$. The set of all mappings Φ is denoted by H_Φ, while the set of all classifiers is denoted by H_w. IRM may be expressed as the constrained optimization problem [1]:

$$\begin{aligned} &\min_{\Phi \in \mathcal{H}_\Phi, w \in \mathcal{H}_w} \sum_{e \in \mathcal{E}_{tr}} R^e(w \circ \Phi) \\ &\text{s.t. } w \in \arg\min_{\bar{w} \in \mathcal{H}_w} R^e(\bar{w} \circ \Phi), \forall e \in \mathcal{E}_{tr}. \end{aligned} \tag{1}$$

If (Φ, w) satisfies the conditions stated above, then $w \circ \Phi$ is an invariant predictor across the environments ϵ_{tr}. S^{IV} denotes the set of representations and accompanying classifiers (Φ, w) that meet the criteria in the above optimization problem 1, where IV stands for invariant. Define the set of invariant predictors $w \circ \Phi$ individually as $\hat{S}^{IV} = \{w \circ \Phi | (\Phi, w) \in S^{IV}\}$.

S^{IV} members are equivalently stated as solutions to

$$R^e(w \circ \Phi) \leq R^e(\bar{w} \circ \Phi), \forall \bar{w} \in \mathcal{H}_w, \forall e \in \mathcal{E}_{tr} \tag{2}$$

The primary conclusion of [1] asserts that if H_w and H_Φ are from the class of linear models, i.e., $w(z) = w^t z$, where $w \in \mathbb{R}^d$, and $\Phi(x) = \Phi_x$ with $\Phi \in \mathbb{R}^{d*n}$, then the solution to 2 stays invariant in ϵ_{all} under specific assumptions on the data generation process and training settings ϵ_{tr}.

4 SYSTEM DESIGN

4.1 Game-Theoretic Reformulation

The optimization problem 1 can be difficult to solve. To overcome it, we propose an alternative characterisation based on game theory. We give each environment its own classification $w^e \in H_w$. To create an overall classifier w^{av}, we employ a basic ensemble such that, $w^{av} : Z \to \mathbb{R}^k$ defined as $w^{av} = \frac{1}{|\varepsilon_{tr}|} \sum_{q=1}^{|\varepsilon_{tr}|} w^q$, where $w^{av}(z) =$

$\frac{1}{|\varepsilon_{tr}|}\sum_{q=1}^{|\varepsilon_{tr}|} w^q(z)$, for each $z \in Z$ (The abbreviation av stands for average). Consider binary classification with two environments $\{e_1, e_2\}$; $w^e = [w^{e_1}; w^{e_2}]$ is the classifier of environment e, where each component represents the score for each class. We define the component j of the ensemble classifier w^{av} as $w_j^{av} = w_j^{e_1} + w_j^{e_2}2$. These scores are fed into a softmax, and the overall probability given to class j for an input z equals $\frac{e^{w_j^{av}(z)}}{e^{w_1^{av}(z)}+e^{w_2^{av}(z)}}$.

This ensemble w^{av} must be used in all environments. We wish to tackle the new optimization challenge listed below.

$$\begin{aligned}&\min_{\Phi\in\mathcal{H}_\Phi, w^{av}\in\mathcal{H}_w} \sum_{e\in\mathcal{E}_{tr}} R^e\left(w^{av}\circ\Phi\right)\\ &\text{s.t. } w^e \in \arg\min_{\bar{w}^e\in\mathcal{H}_w} R^e\left(\frac{1}{|\mathcal{E}_{tr}|}\left[\bar{w}^e + \sum_{q\neq e} w^q\right]\circ\Phi\right), \forall e\in\mathcal{E}_{tr}\end{aligned} \tag{3}$$

We may rephrase the above as follows:

$$\begin{aligned}&\min_{\Phi\in\mathcal{H}_\Phi, w^{av}} \sum_{e\in\mathcal{E}_{tr}} R^e\left(w^{av}\circ\Phi\right)\\ &\text{s.t. } R^e\left(\frac{1}{|\mathcal{E}_{tr}|}\left[w^e + \sum_{q\neq e} w^q\right]\circ\Phi\right)\\ &\leq R^e\left(\frac{1}{|\mathcal{E}_{tr}|}\left[\bar{w}^e + \sum_{q\neq e} w^q\right]\circ\Phi\right) \forall \bar{w}^e\in\mathcal{H}_w \forall e\in\mathcal{E}_{tr}\end{aligned} \tag{4}$$

The benefits of this formulation 4:

(1) Using the ensemble guarantees invariance across environments automatically.
(2) Unlike in 1, when all environments' selections must be the same, each environment is allowed to choose the classifier w^e from the whole collection H_w.
(3) The limitations in 4 correspond to the set of pure NE of a game that we will define next.

The game is played amongst $|\epsilon_{tr}|$ players, with each participant assigned to a different environment e. The set of environment e actions are $w^e \in H_w$. A representation Φ is chosen from the set H_Φ at the start of the game, which is seen by all of the environments. $u_e[w^e, w^{-e}, \Phi] = -R^e(w^{av}; \Phi)$ is the utility function for an environment e, where $w^{-e} = \{w^q\}_{q\neq e}$ is the set of options for all environments except e. This game is known as Ensemble Invariant Risk Minimization (EIRM) and is written as a tuple.

$$\Gamma^{\text{EIRM}} = \left(\mathcal{E}_{tr}, \mathcal{H}_\Phi, \{\mathcal{H}_w\}_{q=1}^{|\mathcal{E}_{tr}|}, \{u_e\}_{e\in\mathcal{E}_{tr}}\right) \tag{5}$$

A pure NE is represented as a tuple: $(\Phi, \{w^q\}_{q=1}^{|\epsilon_{tr}}$). We include Φ as a component of the tuple since each pure NE relies on $\Phi.S^{EIRM}$ is defined as the set of pure NE. We create a collection of all of the NE as ensemble predictors.

$$\hat{\mathcal{S}}^{\text{EIRM}} = \left\{\left[\frac{1}{|\mathcal{E}_{tr}|}\sum_{q=1}^{|\mathcal{E}_{trr}|} w^q\right]\circ\Phi \mid \left(\Phi, \{w^q\}_{q=1}^{|\mathcal{E}_t|}\right)\in\mathcal{S}^{\text{EIRM}}\right\} \tag{6}$$

S^{EIRM} members are represented in the same way as the solution to

$$u_e\left[w^e, w^{-e}, \Phi\right] \geq u_e\left[\bar{w}^e, w^{-e}, \Phi\right], \forall w^e\in\mathcal{H}_w, \forall e\in\mathcal{E}_{tr} \tag{7}$$

If we replace $u_e\left[w^e, w^{-e}, \Phi\right]$ with $-R^e(w^{av}, \Phi)$, we obtain the inequalities in 4.

4.2 Equivalence Between NE and Invariant Predictors

Surprisingly, under relatively moderate circumstances, these two sets $\hat{S}^{IV}$ and $\hat{S}^{EIRM}$ are identical. We construct a stronger result before showing this result, and this result will follow from it.

To create a new set, we utilize the set S^{EIRM}. For each tuple $(\Phi, \{w^q\}_{q=1}^{|\epsilon_{tr}} \in S^{EIRM}$ enhance the ensemble classifier $w^{av} = \frac{1}{|\mathcal{E}_{tr}|}\sum_{q=1}^{|\mathcal{E}_{tr}|} w^q$ to get $\left(\Phi, \{w^q\}_{q=1}^{|\mathcal{E}_{tr}|}, w^{av}\right)$. The set of new tuples are represented as $\hat{S}^{EIRM}$.

To build a new set, we use the set S^{IV}. Take the component $() \in S^{IV}$. We propose the following decomposition for w in terms of environment-specific classifiers: $w = \frac{1}{|\mathcal{E}_{tr}|}\sum_{q=1}^{|\mathcal{E}_{tr}|} w^q$, where $w^q \in H_w$ and $w^q = w; \forall q \in \epsilon_{tr}$ are two simple decompositions. We utilize each of these decompositions and augment the tuple to get ; $(\Phi, \{w^q\}_{q=1}^{|\epsilon_{tr}}, w)$. $\hat{S}^{IV}$ is the name given to this new group of tuples.

Both $\hat{S}^{IV}$ and $\hat{S}^{EIRM}$ sets are made up of tuples of representation, a collection of environment specific classifiers, and an ensemble classifier. We pose an even more intriguing question than the one previously posed. Is the collection of representations, environment-specific classifiers, and ensembles discovered by playing EIRM 7 and solving IRM 2 the same? From **Theorem 1** [12], these two sets are equivalent, then the equivalence of $\hat{S}^{IV}$ and $\hat{S}^{EIRM}$ follows naturally.

Significance of Theorem 1 [12]:

(1) From a computational approach, this equivalence allows game theory methods to be used to the find NE of the *EIRM* game and, as a result, the invariant predictors.
(2) From a theoretical approach, this similarity allows us to apply game theory to study the *EIRM* game solutions and comprehend the invariant predictors.

Role of representation Φ**:** We look at what happens if we fix Φ to the identity mapping; this will inspire one of our ideas. Define $\hat{S}^{EIRM}(\Phi)$ as a collection of ensemble predictors obtained by playing the EIRM game with a fixed representation representation Φ. In the same way, we define $S^{IV}(\Phi)$ as the set of invariant predictors generated from the representation. It follows from Theorem 1 [12] that $S^{EIRM}(\Phi) = S^{IV}(\Phi)$. From Theorem 2 of [12], when we fix the representation Φ to identity and play the EIRM game, all the invariant predictors (with bounded Lp norm) that can be produced using all the representations $\Phi \in H_\Phi$ are successfully recovered. As a result, we can simply set $\Phi = I$ and learn equilibria using game-theoretic techniques.

4.3 Algorithms for Finding NE of Γ^{EIRM}

There are several techniques for computing the equilibrium in the literature, such as best response dynamics (BRD) and fictitious play [4], but none of these strategies is guaranteed to arrive to equilibria in continuous games, with the exception of certain classes of games [5–8]. Given its intuitive and natural structure, BRD is one of the most popular approaches. GANs are also trained using an approximate BRD [9]. In GANs, BRD is not known to converge to equilibrium. Instead, [10], a recent modification of it, obtains mixed NE. Unlike GANs, our game EIRM is a non-zero sum game with ongoing actions. We use this method since there are no known strategies that guarantee the computation of the equilibrium (pure or mixed) for these games, we use the traditional BRD technique.

In our first method, we employ a fixed representation. Remember how in Theorem 2 [12], we demonstrated how just fixing Φ to identity may be a very effective approach. As a result, we may either fix Φ it to be identity mapping or choose another mapping, such as approximation of the map for Gaussian kernel [11]. Once we've fixed Φ, the environments will behave in accordance with the optimal response dynamics, as seen below.

(1) Each environment takes its turn (in a cyclical fashion, with each environment going once) and minimizes its particular target.

```
Algorithm 1 Best Response Training
  Input: Data for each environment and combined data
  while iter $\leq$ iter$_{\text{max}}$ do
    if Fixed-Phi then
      $\Phi_{\text{cur}} = \text{I}$
    end if
    if Variable-Phi then
      $\Phi_{\text{nxt}} = \text{SGD}\Big[\sum_e R^e(w^{av}_{\text{cur}} \circ \Phi_{\text{cur}})\Big]$, SGD[.]: update using stochastic gradient
      descent
      $\Phi_{\text{cur}} = \Phi_{\text{nxt}}$
    end if
    for $p \in \{1, ..K\}$ do
      for $e \in \{1, .., |\mathcal{E}_{tr}|\}$ do
        $w^e_{\text{nxt}} = \text{SGD}\Big[R^e(w^{av}_{\text{cur}} \circ \Phi_{\text{cur}})\Big]$
        $w^e_{\text{cur}} = w^e_{\text{nxt}}$
      end for
      iter = iter + 1
      $w^{av}_{\text{cur}} = \frac{1}{|\mathcal{E}_{tr}|} \sum_e w^e_{\text{cur}}$
    end for
  end while
```

Fig. 1. Algorithm 1

(2) Repeat this approach until a certain condition is met, such as the maximum number of epochs or the required level of training accuracy.

The approach described above leaves little room for improvement. We return to 4 and use the upper level optimization objective to guide our search for Φ. The representation learner updates the environments regularly using the objective in 4 and between two updates of Φ the environments play according to the optimal response dynamics as stated above in this new technique.

Now we'll make assumptions about H_w and $_\Phi$ and present a comprehensive approach (see Algorithm 1) that we'll employ in tests. We suppose that we are parametrized by the w^e family of neural networks $\theta_w \in \Theta_w$. In Algorithm 1, one of the variables Fixed-Phi (for our first method) or Variable-Phi is set to true, and then accordingly Φ remains fixed or is updated periodically. We also show an example of the best response training when there are two settings and one representation learner in Figure 2.

5 RESULTS

5.1 Benchmarks

The most essential comparison point is [1], which we refer to as IRM, in the comparisons We employ the architecture defined in their work . We also copare with Empirical risk minimization variants. For EIRM games, we have two approaches: one that utilizes a xed to the identity and the other that uses a variable, which we refer to as the F-IRM and V-IRM games, respectively. We have used same architectures, hyperparameters, and optimizers used in [12].

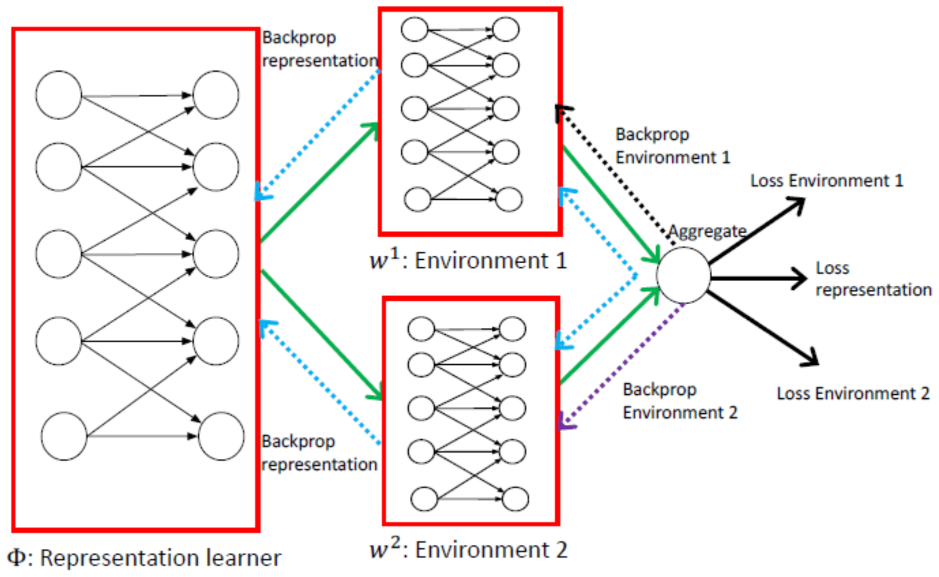

Fig. 2. Illustration of best response training with 2 environments and representation learner. Dotted lines for back propagation and solid lines for forward pass.

5.2 Dataset

We have used colored MNIST digits and colored Fashion MINST dataset in our experiment.

5.2.1 Colored MNIST Digits: We have created a training environment for classifying digits in MNIST digits data, in which the pictures in MNIST are now colored in such a way that the colors spuriously correlate with the labels. The aim is to determine if the digit is less than 5 (excluding 5) or greater than 5. There are 3 environments (two training with 30,000 points each, one test with 10,000 points). To construct the final label, we add noise to the preliminary label ($y = 0$ if the digit is between 0-4 and $y = 1$ if the digit is between 5-9) by flipping it with 25% probability. We sample the color id z by flipping the final labels with probability p_e, where p_e is 0.2 in the first environment, 0.1 in the second, and 0.9 in the third. The testing environment is the third environment. If z = 1, we paint the digit red, otherwise, we color it green. The results are for training and testing accuracy is shown in Table 1.

Table 1. Colored MNIST Digits with 2 Environments: Comparison of methods in terms of training, testing accuracy

Model	Training Accuracy	Testing Accuracy
FIRM	59.71 %	60.68 %
VIRM	53.48 %	59.81 %

5.2.2 Colored MNIST Fashion: We edit the fashion MNIST dataset in the same way that we did the MNIST digits dataset. Fashion MNIST data includes pictures from the following categories: "t-shirt," "trouser," "pullover," "dress," "coat," "sandal," "shirt," "sneaker," "bag," and "ankle boots." We colorize the photographs in such a manner that the colors match the labels. The aim is to determine if the image depicts footwear or apparel. There are three types of environments (two training, one test) We add noise to the preliminary label (y = 0: "t-shirt", "trouser", "pullover", "dress", "coat", "shirt" and y = 1: "sandle", "sneaker", "ankle boots") by ipping it with 25% probability to generate the final label. We sample the color id z with probability p_e by flipping the noisy label, where p_e is 0.2 in the first environment, 0.1 in the second environment, and 0.9 in the third environment, which is the test environment. If z = 1, we paint the item red, otherwise, we color it green.

The results are for training and testing accuracy is shown in Table 1. From the reults it is observed that the standard ERM based approaches has nearly 15% accuracy on testing set. The F-IRM and V-IRM game gets a testing accuracy of nearly 52 percent. This suggests that, unlike ERM-based techniques, the model does not employ false correlation and robust to change in the colors.

Table 2. Colored MNIST Fashion with 2 Environments: Comparison of methods in terms of training, testing accuracy

Model	Training Accuracy	Testing Accuracy
FIRM	59.43 %	52.08 %
VIRM	59.20 %	52.78 %
IRM	56.61 %	43.52 %
ERM	84.04 %	14.92 %

6 CONCLUSION

In this project, to discover invariant predictors, we created a framework based on game-theoretic techniques. We deal with data from a variety of sources. We set up an ensemble game in our framework; we build an ensemble of classifiers, with each environment influencing a fraction of the ensemble. Surprisingly, the collection of solutions to this game matches the set of invariant predictors across training conditions. The suggested framework outperforms the existing framework of [1] and has a reduced variance. We anticipate that this paradigm opens up new avenues for addressing additional problems involving invariance in causal inference using tools from game theory.

REFERENCES

[1] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz, risk minimization," arXiv preprint arXiv:1907.02893, 2019.
[2] S. Beery, G. Van Horn, and P. Perona, in terra incognita," in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 456-473.
[3] P. de Haan, D. Jayaraman, and S. Levine, confusion in imitation learning," in Advances in Neural Information Processing Systems, 2019, pp. 11 693-11 704.
[4] D. Fudenberg, F. Drew, D. K. Levine, and D. K. Levine, The theory of learning in games. MIT press, 1998, vol. 2.
[5] J. Hofbauer and S. Sorin, response dynamics for continuous zero-sum games," Discrete and Continuous Dynamical Systems Series B, vol. 6, no. 1, p. 215, 2006.
[6] E. Barron, R. Goebel, and R. Jensen, response dynamics for continuous games," Proceedings of the American Mathematical Society, vol. 138, no. 3, pp. 1069-1083, 2010.
[7] P. Mertikopoulos and Z. Zhou, in games with continuous action sets and unknown payo functions," Mathematical Programming, vol. 173, no. 1-2, pp. 465-507, 2019.
[8] S. Bervoets, M. Bravo, and M. Faure, and convergence to nash in games with continuous action sets," Working paper, Tech. Rep., 2016.
[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D.Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, adversarial nets," in Advances in neural information processing systems, 2014, pp. 2672-2680.
[10] Y.-P. Hsieh, C. Liu, and V. Cevher, mixed nash equilibria of generative adversarial networks," arXiv preprint arXiv:1811.02002, 2018.
[11] A. Rahimi and B. Recht, features for large-scale kernel machines," in Advances in neural information processing systems, 2008.
[12] Kartik Ahuja, Karthikeyan Shanmugam, Kush R. Varshney, and Amit Dhurandhar,"Invariant Risk Minimization Games".

7 APPENDIX

7.1 Architecture, Hyperparameter and Training Details

7.1.1 Architecture for 2 player EIRM game with fixed Φ: We utilized the following architecture for the two models in the game with fixed Φ. The model employed is a simple multilayer perceptron with the parameters shown below.

(1) Input layer: Input batch (batch; len; wid; depth) $\longrightarrow$ Flatten
(2) Layer 1: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(3) Layer 2: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(4) Output layer: Fully connected layer, output size = 2

We utilize the aforementioned design for all of our experiments. The shape of the input in the above architecture depends on the dimensions of the data that are input.

7.1.2 Architecture for 2 player EIRM game with variable Φ: We utilized the following architecture for the two models in the game with variable Φ. The architecture for the representation learner is:

(1) Input layer: Input batch (batch; len; wid; depth) $\longrightarrow$ Flatten
(2) Layer 1: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(3) Layer 2: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(4) Output layer: Fully connected layer, output size = 2

he output from the representation learner above is fed into two MLPs one for each environment (we use the same architecture for both environments).

(1) Layer 1: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(2) Layer 2: Fully connected layer, output size = 390, activation = ELU, L2-regularizer = 1.25e-3, Dropout = 0.75
(3) Output layer: Fully connected layer, output size = 2

We utilize the aforementioned design for all of our experiments. The shape of the input in the above architecture depends on the dimensions of the data that are input.

7.1.3 Optimizer and other hyperparameters : For training, we employed the Adam optimizer with a learning rate of 2.5e-4. The cross-entropy loss function is optimized. The batch size was set at 256. We end the algorithm in accordance with the rules outlined in this study. As a result, the number of training stages might differ between trials. All techniques have a warm start phase; we set the warm start phase to be equal to the number of steps in one epoch, where one epoch equals (training data size/ batch size). We set the period to 2 for the fixed Φ setup, which means that in one step, the first model trains and in the next, the second model trains, and this cycle repeats throughout the training. For the variable Φ setup, we let the two environments and representation learner take turns updating their respective models, with environment 1 training in one step, environment 2 training in the next, representation learner training, and so on.

7.2 Contribution

(1) I have done literature survey on IRM games [12], IRM [1] and ERM.
(2) Worked on MINST Fashion dataset for F-IRM, V-IRM, IRM algorithms.
(3) Worked on slides 10-18 in the final presentation.