An Empirical Study of Invariant Risk Minimization on Deep Models

Yong Lin [*1] **Lian Qing** [*1] **Tong Zhang** [1]

Abstract

Invariant Risk Minimization(IRM) is a promising framework to prevent the model from relying on spurious features. Various variants of IRM have been proposed since the initial work (Arjovsky et al., 2019). However, can IRM and its variants work well on deep models (e.g. ResNet-18 and ResNet-50)? Except for several negative results of IRMv1 mentioned in Gulrajani & Lopez-Paz (2020); Ahmed et al., this question remains underexplored so far. In this work, we undertake an extensive empirical evaluation of IRM methods on deep models with a synthetic dataset from CIFAR-10 and MNIST. Among the tested IRM methods, InvRat is the only one that can consistently learn invariant models in all the considered settings. However, the performance of InvRat still heavily relies on the way of parameterization. Several popular IRM variants (i.e. IRMv1 and REx) perform poorly, especially on ResNet-50. We release the implementations of the dataset and models to facilitate further research.

1. Introduction

In recent decades, deep learning based approaches have achieved great success in many real-world applications, such as computer vision (Krizhevsky et al., 2012; He et al., 2016), speech recognition (Graves et al., 2013), and game playing (Silver et al., 2017). Most of the approaches are optimized by Empirical Risk Minimization (ERM) under the i.i.d assumption that the training and test data are independently and identically drawn from the same distribution. However, the i.i.d assumption does not always hold in real scenarios. Specifically, the ERM-based models face catastrophic performance drop when the test distribution is different from the training one.

To relax the i.i.d assumption, Arjovsky et al. (2019) proposes Invariant Risk Minimization (IRM) that regularizes the model to learn invariant features rather than spurious features. Specifically, IRM considers the setting that the training data is collected from several environments, where the correlation of spurious features with the target changes in different environments while the correlation of invariant features remains the same. The goal of IRM is to learn an invariant representation that merely depends on invariant features. Hopefully, the model can be robust to unseen environments based on the learned invariant features.

IRM becomes popular and inspires a line of works. Several variants are proposed in Ahuja et al. (2020); Chang et al. (2020); Krueger et al. (2020); Xie et al. (2020). The theoretical property of IRM is analyzed in Rosenfeld et al. (2020); Ahuja et al. (2020g); Kamath et al. (2021). However, despite the popularity of IRM, it remains underexplored on the question of whether IRM methods can work out well on deep models (i.e. ResNet-18 and ResNet-50), except for some negative results of IRMv1 mentioned in Ahmed et al.; Gulrajani & Lopez-Paz (2020).

In this work, we carry out extensive experiments of IRM methods on ResNet-18 and ResNet-50 on a synthetic dataset from CIFAR-10 and MNIST. Different from Aubin et al. (2021); Ahmed et al.; Gulrajani & Lopez-Paz (2020), our settings is kept as close as possible to the CMNIST (Arjovsky et al., 2019) to retain simplicity. However, even in the considered settings, we find that several popular variants of IRM, i.e. IRMv1 (Arjovsky et al., 2019) and REx (Krueger et al., 2020), fail on ResNet-50. InvRat (Chang et al., 2020) performs well on the considered settings of ResNet-18 and ResNet-50. However, we find that the parameterization methods play a non-trivial role in InvRat. In several settings, InvRat with the parameterization method adopted in Chang et al. (2020) works much less effectively than the parameterization proposed in this work in Section 3.

We summarize our contributions as follows:

- We compare IRM methods on deep models (i.e. ResNet-18 and ResNet-50) on a synthetic dataset constructed from CIFAR-10 and MNIST. InvRat (Chang et al., 2020) performs well on both ResNet-18 and ResNet-50 , whereas, several popular IRM variants, i.e. IRMv1 and REx, fail on ResNet-50.

[*]Equal contribution [1]The Hong Kong University of Science and Technology . Correspondence to: Yong Lin <ylindf@connect.ust.hk>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning.,

- We find that the parameterization methods play a non-trivial role in InvRat. The parameterization method adopted in Chang et al. (2020) works much less effectively in several settings than the one proposed in Section 3.
- We provide the code of the dataset and the implementation of several IRM variants at `https://github.com/IRMBed/IRMBed`, which serves as a baseline for future work.

2. Invariant Risk Minimization(IRM) and Its Variants

In this section, we will review IRM and its several variants.

Preliminaries. Throughout the paper, upper-cased letters, X and Y, denote random variables; lower-cased letters, x and y, denote deterministic instances. Following Arjovsky et al. (2019), we consider the datasets $D := \{D_e\}_{i=1}^{|\mathcal{E}_{tr}|} = \{(x_i^e, y_i^e)\}_{i=1}^{|\mathcal{E}_{tr}|}$, which are collected from multiple environments $\mathcal{E}_{tr}$. The instance (x^e, y^e) from D_e follows the probability distribution $\mathcal{D}^e$. We then use $h_u(\cdot)$ and $g_v(\cdot)$ to denote the representation extractor and the classifier, which are parameterized by u and v respectively. The goal of out-of-distribution generalization is to minimize the following target:

$$\sup_{e\in\mathcal{E}} R^e(g_v(f_u(X^e)), Y^e),$$

where $R^e(g_v(h_u(X)), Y) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}^e} l(g_v(h_u(x)), y)$. Here l is the loss function. We denote $R^e_{u,v} := R^e(g_v(h_u(X)), Y)$ for simplicity in the rest of the paper.

Invariant Risk Minimization(IRM). Formally, IRM aims to solve the following problem:

$$\begin{aligned} &\min_{u,v} \textstyle\sum_{e\in\mathcal{E}_{tr}} R^e_{u,v} \\ &s.t. \quad v \in \arg\min_{v^e} R^e_{u,v^e}, \forall e \in \mathcal{E}_{tr} \end{aligned} \tag{1}$$

IRM described in Equation 1 aims to learn an invariant representation $h_u(X)$ based on which the classifier g_v is simultaneously optimal for all environments. In order to achieve such constrain, $h_u(X)$ should screen out the environment-dependent (spurious) features and retain the environment-independent (invariant) features.

IRMv1. Since (1) is a challenging bi-level optimization problem, Arjovsky et al. (2019) proposes IRMv1 to approximate the solution of (1). IRMv1 is shown as following:

$$\min_{u,v} \sum_{e\in\mathcal{E}_{tr}} R^e_{u,v} + \lambda\|\nabla_v R^e_{u,v}\|^2 \tag{2}$$

InvRat (Chang et al., 2020). InvRat uses another classifier, $g'_{v'}(\cdot, E)$, to test whether $g_v(\cdot)$ is simultaneously optimal for all environment. The only difference between the two classifiers is that $g'_{v'}(\cdot, e)$ also takes the environment index(embedding) as additional input. If knowing the environment index e can help to predict Y better, it means that $g_v(\cdot)$ is not optimal for some environment e. InvRat can be formulated as a minimax game (3):

$$\begin{aligned} &\min_{u,v}\max_{v'} \sum_e R^e(g_v(h_u(X)), Y) + \\ &\lambda\sum_e R^e(g_v(f_u(X)), Y) - R^e(g'_{v'}(f_u(X), e), Y) \end{aligned} \tag{3}$$

Besides IRMv1 and InvRat, there are several other IRM variants: REx (Krueger et al., 2020) and RVP (Xie et al., 2020) propose to penalize the variance of losses of different environments; IRMGame (Ahuja et al., 2020) seeks for invariant presentation by finding the Nash equilibrium of an ensemble game.

3. Parameterizing InvRat

InvRat shown in (3) is a flexible framework. One may have several possible choices to parameterize $g'_{v'}(f_u(X), E)$. In the following part, we first introduce the parameterization adopted in Chang et al. (2020) and then discuss about another parameterization method which gains a strong empirical performance.

InvRat with **E**nvironmental em**B**edding (InvRat-EB) (Chang et al., 2020). Let's consider a sample (x, y) from environment e. Chang et al. (2020) first learns a embedding function $m(\cdot)$ that maps e to a latent space. The classifier $g'_{v'}$ consumes both the extracted feature $h_u(x)$ and environment embedding $m(e)$ to make prediction, i.e. $g'_{v'}(h_u(x), m(e))$. We name this parameterization method as InvRat with environmental embedding (InvRat-EB) because it learns an embedding for each environment.

$$\begin{aligned} &\min_{u,v}\max_{v',m} \sum_e R^e(g_v(h_u(X)), Y) + \\ &\lambda\sum_e R^e(g_v(h_u(X)), Y) - R^e(g'_{v'}(h_u(X), m(e)), Y) \end{aligned} \tag{4}$$

InvRat with **E**nvironmental **C**lassifier (InvRat-EC). Here, we present another method to parameterize InvRat. We train several environmental classifiers, $\{g_{v^e}\}_e$, one for each environment. Specifically, g_{v^e} is trained to fit environment e. Each environmental classifier g_{v^e} shares the same structure with the shared classifier g_v. Then (3) converts to the following objective:

$$\begin{aligned} &\min_{u,v}\max_{\{v^e\}_e} \sum_e R^e(g_v(h_u(X)), Y) + \\ &\lambda\sum_e R^e(g_v(h_u(X)), Y) - R^e(g_{v^e}(h_u(X)), Y) \end{aligned} \tag{5}$$

Proposition 1 *InvRat-EC defined in (5) is equivalent to InvRat defined in (3) with* $g'_{v'}(\cdot, e_1) = \sum_{e_2} g_{v^{e_2}}(\cdot) I(e_1, e_2), \forall e_1 \in \mathcal{E}_{tr}$, *where* $I(e_1, e_2)$ *is the indicator function that outputs 1 if* $e_1 = e_2$, *otherwise 0. Each element of* $\{g_{v^e}\}$ *share the same structure with* g_v.

Proposition 1 shows that InvRat-EC is contained in InvRat with $g'_{v'}$ chosen from a special function class. In the inner loop, we aim to optimize $\{g_{v^e}\}_e$ by maximizing the negative loss $-\sum_e R^e(g_{v^e}(h_u(X)), Y)$. The loss gap between the g_v and g_{v^e} indicates how much environmental (spurious) information is contained in the representation $h_u(X)$, which serves as the penalty of violating the invariance condition. In the outer loop, the representation function and shared classifier are optimized to minimize the empirical loss as well as the invariance penalty. The algorithm of InvRat-EB is shown in Algorithm 1

Algorithm 1 InvRat-EC

Input: environment number N, penalty weight λ, learning rate η , η' , function h_u, g^v, $\{g_{v^e}\}_{e=1}^N$, loss function l, the number of inner steps K.

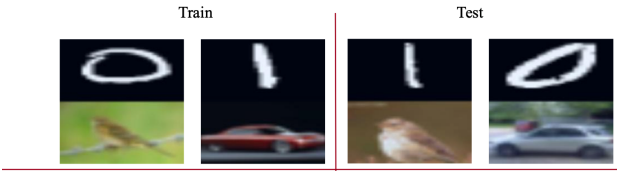
while *Training* **do**
  for e = *1:N* **do**
    for i=*1:K* **do**
      Obtain N^e samples from environment e.
      $R^e_{u,v^e} = 1/N_e \sum_{j=1}^{N_e} l(g_{v^e}(h_u(x_j)), y_j)$
      $v^e \leftarrow v^e - \eta' \nabla_{v^e} R^e_{u,v^e}$
    Obtain N^e samples from environment e.
    $R^e_{u,v} = 1/N_e \sum_{j=1}^{N_e} l(g_v(h_u(x_j)), y_j)$
  $u \leftarrow u - \eta \nabla_u \big(\sum_{e=1}^N R^e_{u,v} + \lambda (R^e_{u,v} - R^e_{u,v^e}) \big)$
  $v \leftarrow v - \eta \nabla_v \big(\sum_{e=1}^N R^e_{u,v} + \lambda (R^e_{u,v} - R^e_{u,v^e}) \big)$

4. Experiments and Results

4.1. Dataset and Experimental Setup

CIFAR-MNIST dataset: Inspired by Shah et al. (2020), we construct a CIFAR-MNIST dataset, in which each image is synthesized by concating two component images, one from CIFAR-10 and the other from MNIST. We make the CIFAR-10 and MNIST component behave as invariant and spurious features, respectively. Specifically, the label of the synthesized image is generated from the CIFAR-10 component and the MNIST component exhibit a high but unstable correlation with the label. Figure 1 shows an illustration of the dataset. Following Arjovsky et al. (2019), we construct several environments. The MNIST component's correlation with the label changes across different environments while the CIFAR-10 component's correlation remains invariant.

We consider two settings for the training sets: 1). 2 Env: the training data contains two environments, in which the spurious correlations are 99.9% and 80.0%, respectively, 2). 4 Env: the training data contains four environments, in which the spurious correlations are 99.9%, 95.0%, 90.0%, 80.0%, respectively. In both settings, we set the correlation of spurious features to 10% in testing environment to see whether the learned model relies on the spurious feature. We also add 10% label noise as Arjovsky et al. (2019) does.



Y	0	1	1	0
Spurious Feature	MNIST "0"	MNIST "1"	MNIST "1"	MNIST "0"
Invariant Feature	Cifar "Bird"	Cifar "Car"	Cifar "Bird"	Cifar "Car"

Figure 1. Illustration of the synthetic dataset from CIFAR-10 and MNIST. We first randomly select two classes ("car" and "bird") from CIFAR-10. Then each CIFAR-10 image is concatanated with an image from MNIST ("0" and "1"). The CIFAR-10 component serves as the invariant feature and the label is generated from the CIFAR-10 component. The MNIST component serves as the spurious feature. The MNIST component is highly correlated with the label in the training dataset, however, the correlation reverses in the testing dataset.

Experimental setup. In this work, we adopt the ResNet-18 and ResNet-50 (He et al., 2016) for experiments. We initialize the network with ImageNet-pretrained weight and train for 100 epochs. The networks are optimized by SGD with the learning rate as 0.01. The batch size is 128 and 32 for ResNet-18 and ResNet-50, respectively. We search for the best penalty weight in $[1, 10, 10^2, 10^3, 10^4, 10^5]$. We investigate both imposing penalty on the output(i.e. choose the scalar multiplied on the output as w for IRMv1 (Arjovsky et al., 2019)) and on the hidden feature (i.e. choose the fully connected layer as g_v for IRMv1).

We append (o) and (h) after the model name to denote imposing penalty on the output and the hidden feature, respectively. One-layer fully connected module is adopted in ResNet for models except InvRat-EB. We use a two-layer fully connected module as $g'_{v'}$ for InvRat-EB to ensure that the embedding of environment can interact with the features. The hidden dimension of the two-layer fully connected NN is 512 for ResNet-18 and 2048 for ResNet-50. We search for the best dimension of environment embedding in [64, 128, 256, 512] for InvRat-EB. Following Goodfellow et al. (2014), we use iterative gradient ascend and descend to solve the minimax game of InvRat-EC and InvRat-EB.

4.2. Experimental results

Table 1 shows the test accuracy of different IRM methods on the CIFAR-MNIST dataset. ERM and ERM(Oracle) in Table 1 stand for empirical risk minimizing on the CIFAR-MNIST dataset with and without spurious feature, respectively. ERM(Oracle) achieves over 80% accuracy on both ResNet-18 and ResNet-50, whereas, the test accuracy of ERM is less than 40%. This means that ERM heavily relies on the spurious feature. The test accuracy of ERM on ResNet-50 is even worse than that of ResNet-18. This is consistent with the finding in Sagawa et al. (2020) that over-overparameterization exacerbates spurious correlations.

Results on ResNet-18. IRMv1(o) has 70.8% test accuracy on ResNet-18(2Env). However, the test accuracy drops to 53.6% on ResNet-18 with four environments. IRMv1(h) does not work out well on both ResNet-18(2Env) and ResNet-18(4Env). This indicates that IRMv1 fails to learn invariant representations when imposing penalty on the hidden feature. Notably, REx, RVP and IRMGame do not perform well in the considered settings. InvRat-EB(h) and InvRat-EC(h) both achieve over 75% accuracy ResNet-18(2Env). Though the test accuracy of InvRat-EB(h) is averagely 68.6% on ResNet-18(4Env), the standard deviation is 9.4%, which indicates the instability of InvRat-EB(h) in this setting. InvRat-EC(h) is the only method that obtains an accuracy higher than 70% on ResNet-18(4Env).

Results on ResNet-50. Table 1 shows that IRMv1, REx, RVP and IRMGame all fail on both ResNet-50(2Env) and ResNet-50(4Env). Notably, IRMv1 works out well on ResNet-18(2Env) but fails on ResNet-50(2Env). Similar results can also be found in InvRat-EB, the performance of which deteriorates on ResNet-50. It indicates that deeper network may add more difficulties in learning invariant representation. The performance of InvRat-EC(h) is the best among the considered IRM variants on ResNet-50. Its accuracy is 81.2% on ResNet-50(2Env) and 77.9% on ResNet-50(4Env).

Regularizing on the output or the hidden feature? Comparing InvRat-EC(h) with InvRat-EC(o), we can see that regularizing on the hidden feature consistently improves the performance over regularizing on the output. This is consistent with the common practice in several related fields, i.e. Domain Adaption (Ganin et al., 2016; Tzeng et al., 2014) and Domain Generalization (Shankar et al., 2018; Piratla et al., 2020). However, IRMv1(h) performs much worse than IRMv1(o). IRMv1(h) fails even in the setting ResNet-18(2Env) where IRMv1(o) has 70.8% test accuracy. This shows that IRMv1 is less effective when function space of g_v becomes larger.

InvRat-EB vs InvRat-EC. As shown in Table 1, InvRat-EB(h) is less effective than InvRat-EC(h) in three settings, ResNet-18(4Env), ResNet-50(2Env) and ResNet-50(4Env). InvRat-EB(h) only has 55.9% test accuracy on ResNet-50(4Env), which is significantly worse than InvRat-EC(h). In Table 2, we provide more results InvRat-EB on ResNet-50(2Env) and ResNet-50(4Env). Specifically, we try different embedding size and number of hidden neurons. We find that InvRat-EB does not perform well on these combination of hyper-parameters. These results indicate that the parameterization method has a significant influence on the performance of InvRat. According to our observation, InvRat-EC(h) is more stable than InvRat-EB(h). We leave it to future work to understand the mechanism of the parameterization methods.

Table 1. Test Accuracy of Different Models on the CIFAR-MNIST Dataset. "o" and "h" denote imposing penalty on the output and the hidden feature, respectively. "Oracle" denotes that the spurious feature is removed (the mnist components are replaced by blank images).

Method	ResNet-18		ResNet-50	
	2 Env	4 Env	2 Env	4 Env
ERM(Oracle)	83.7±1.5	83.2±0.9	84.8±0.8	85.0±0.8
ERM	39.5±0.4	35.7±0.7	38.1±0.8	32.0±2.1
IRMv1(o)	70.8±0.4	53.6±3.1	43.3±12.5	51.8±4.9
REx(o)	52.5±2.2	44.5±1.7	51.1±2.2	37.6±7.3
RVP(o)	49.4±3.1	49.4±1.1	50.2±0.3	50.1±0.7
IRMv1(h)	48.6±2.1	39.1±4.8	43.3±3.2	46.7±2.2
IRMGame(h)	43.0±1.9	37.2±2.8	44.5±1.5	40.1±5.5
InvRat-EB(h)	**77.6**±2.0	68.6±9.4	68.5±3.6	55.9±14.4
InvRat-EC(o)	67.5±0.2	67.7±4.0	73.8±1.7	68.4±1.8
InvRat-EC(h)	**75.8**±1.2	**73.4**±4.0	**81.2**±1.3	**77.9**±4.2

Table 2. Experimental results of InvaRat-EB (ResNet50) on CIFAR-MNIST dataset. We try the combination of different embedding size (denoted as "Emb", and different size of hidden channel in g_v (denoted as "Hid").

Emb \ Hid	2 Env		4 Env	
	512	2048	512	2048
64	50.0±4.0	68.5±3.6	47.7±19.4	31.0±1.6
128	64.3±8.6	47.9±3.7	39.0±2.7	37.7±2.1
256	68.7±12.3	54.5±15.6	49.4±10.1	55.9±14.4
512	67.2±8.8	56.4±9.3	54.9±14.5	44.5±7.0

5. Conclusion

In this work, we construct a synthetic dataset to study the IRM methods on deep models. The performance of IRM methods is significantly different on deep models than on shallow models as reported in existing works. This work serves as a baseline for IRM methods on deep models. We make the dataset as well as the implementation of IRM methods public to facilitate further research.

References

Ahmed, F., Bengio, Y., van Seijen, H., and Courville, A. Systematic generalisation with group invari-ant predictions.

Ahuja, K., Shanmugam, K., Varshney, K., and Dhurandhar, A. Invariant risk minimization games. *arXiv preprint arXiv:2002.04692*, 2020.

Ahuja, K., Wang, J., Dhurandhar, A., Shanmugam, K., and Varshney, K. R. Empirical or invariant risk minimization? a sample complexity perspective. *arXiv preprint arXiv:2010.16412*, 2020g.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. 2019.

Aubin, B., Słowik, A., Arjovsky, M., Bottou, L., and Lopez-Paz, D. Linear unit-tests for invariance discovery. *arXiv preprint arXiv:2102.10867*, 2021.

Chang, S., Zhang, Y., Yu, M., and Jaakkola, T. S. Invariant rationalization. *arXiv preprint arXiv:2003.09772*, 2020.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. Ieee, 2013.

Gulrajani, I. and Lopez-Paz, D. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kamath, P., Tangella, A., Sutherland, D. J., and Srebro, N. Does invariant risk minimization capture invariance? *arXiv preprint arXiv:2101.01134*, 2021.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Priol, R. L., and Courville, A. Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688*, 2020.

Piratla, V., Netrapalli, P., and Sarawagi, S. Efficient domain generalization via common-specific low-rank decomposition. In *International Conference on Machine Learning*, pp. 7728–7738. PMLR, 2020.

Rosenfeld, E., Ravikumar, P., and Risteski, A. The risks of invariant risk minimization. *arXiv preprint arXiv:2010.05761*, 2020.

Sagawa, S., Raghunathan, A., Koh, P. W., and Liang, P. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pp. 8346–8356. PMLR, 2020.

Shah, H., Tamuly, K., Raghunathan, A., Jain, P., and Netrapalli, P. The pitfalls of simplicity bias in neural networks. *arXiv preprint arXiv:2006.07710*, 2020.

Shankar, S., Piratla, V., Chakrabarti, S., Chaudhuri, S., Jyothi, P., and Sarawagi, S. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

Xie, C., Chen, F., Liu, Y., and Li, Z. Risk variance penalization: From distributional robustness to causality. *arXiv preprint arXiv:2006.07544*, 2020.