Adaptive Regularization in Invariant Risk Minimization
3MD3020: Deep learning Project Final Report

Badr Youbi Idrissi

Abstract

Deep learning performs astonishingly well on an ever growing list of tasks. It has enabled learning on problems with very complex statistical dependences. However, one might ask what happens when those dependences change? The answer to this question contains some crucial insight into why deep neural networks fail in non intuitive ways and seem to "generalize" poorly in some cases. An elegant framework to address this question is that of causal inference. This work will thus try to introduce the notion of causality. We will talk about how one can hope to extract causal relationships from data. A focus will be made on a particular angle which tries to achieve this with the notion of ***invariance****. Some improvements on previous work will be reported through simple linear experiments. We propose an adaptive regularization optimization scheme that successfully improves empirical behaviour. This report is about ongoing work that will go beyond the scope of the deep learning class. This project's goal was to gain a better understanding of the subject and make progress on a personnal project.*

1. Introduction

'Cum hoc ergo propter hoc' or in english 'With this, therefore because of this'. This famous logical fallacy is taught in every introduction to statistics class to prevent false conclusions when analysing data. We hear time and again that 'Correlation does not imply causation'. And, although we are taught various aspects of correlation, causation is often left on the sidelines.

Humans have an intuitive understanding of causality. It is how we try to make most decisions. For instance one knows that atmospheric pressure causes rain to fall which causes the floor to be wet. The correlation between any two of these events is symmetrical, but the causal link between them isn't. Let us denote X, Y and Z three random variables that measure atmospheric pressure, how hard it rains, and how wet the floor is, respectively. Say we want to predict Y to decide whether or not to take an umbrella. If we have a high enough number of **independent, identically distributed** samples of these variables we will notice a high positive correlation between X and Y but also between Z and Y. What should we take into account when trying to decide whether or not to take an umbrella ? Intuitively, one would answer that we should only take into account X the atmospheric pressure. The reason being that X is a cause of Y, therefore we expect it to be a reliable variable to predict Y. We will see later how we can think of this more formally.

Statistical learning often makes this assumption that data samples $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ are a realization of **independent, identically distributed** random variables $(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)$. It then extracts the intricate statistical associations and dependences between the variables in the data. These statistical dependences can be summarized with $P_{X,Y}$ the joint probability of X and Y. With this joint probability we can, for example, regress Y over X by finding the conditional expectation $E[Y|X = x]$. Or, classify a point x to the most probable value of y (in case it is discrete). However, the i.i.d assumption limits us to the case where we have the same underlying distribution for training and testing. Just knowing $P_{X,Y}$ doesn't inform us on what happens when the underlying distribution changes. In practice however, data is rarely purely i.i.d. The training and testing distributions can be quite different. In that case, the i.i.d assumption can lead to poor performance. This is very often witnessed in deep learning. [5, 6, 16, 17, 14, 9] For instance, there is often a rift between the performance of models on benchmarks vs their performance in real life. Benchmarks are often made i.i.d artificially by shuffling them. Indeed, shuffling the data then splitting it into train and test yields very close distributions. In practice however, one doesn't have access to the test data yet so this can't be done. Sometimes, the performance on benchmarks suggests super human levels. But when the model is deployed, its accuracy quickly degrades. Is this just a case of lacking data? In the following, we argue that more data doesn't necessarily solve the problem and a shift in paradigm is needed.

This report will first briefly present my understanding of the causal framework in section 2. As we will see in section 3, the concept of invariance emerges naturally from this. Invariance motivates the Invariant Risk Minimization

[1] method that is presented in section 3.2. This method is the basis of this work, as we aim to improve it. The ideas we've had to improve IRM will be presented in section 4.2 along with experimental results.

2. Causal Inference

Causal Inference is a big area of research that needs many monographs for a full formal description. We refer the interested reader to the following : The book of why [11] for a less formal read on causality. Causality [10] for a more involved mathematical treatment. And finally Elements of Causal Inference [13] for a more recent text. This short paragraph in contrast, only reflects my current understanding and the necessary elements needed for the rest of this report.

Let us reconsider the example taken in the introduction. It is best represented with the graph in figure 1. The nodes are random variables and the edges represent the causal relationship between the variables. It is a directed edge since cause and effect are assymetrical. Consider the following :

$$Y := f(X, N_1) \text{ and } Z := g(Y, N_2)$$

These relations define the structural equation model (SEM) underlying the data. Where $N_1 \perp\!\!\!\perp N_2$. These equations are to be understood as assignements. So should the distribution of X, N_1 change, Y would still be $Y := f(X, N_1)$. For example if we fix X to the constant 1 then we would have $Y = f(1, N_1)$. This describes the mechanism with which the variables are generated. One key idea behind this is that the mechanism that generates each variable is independent of the distribution of the input variables.

Figure 1. Graphical model of X, Y, Z as presented in the introduction

The parents in the graph are the direct causes of each variable. Once X is known, knowing Z won't add any information about Y. We say that Y is conditionally independent of Z given X. So as long as the relation between Y and its parents doesn't change, we can reliably predict Y given X even if the rest of the graph changes. The graph is thus modular. Interventions on one part of the graph don't necessarily affect the whole graph.

To get back to our example, say we collect data in two cities. In the first city, it rains quite often. In this city we never see a wet floor except when it rains. In the other city, it rains much less, therefore people need to water their plants. It happens in this case that the floor is wet without the rain having fallen. The data in these two cities come from different but related distributions. For instance, the probability of atmospheric pressure changes from one city to another. We also have the apparition of an other variable representing the need of the citizens to keep their plants alive. (See figure 2) This in turn changes the distribution of Z. But all of these interventions don't affect the relation between X and Y. Causal relationships seem to have this **invariance** proprety with respect to interventions. Since in one of the cities making the floor wet didn't provoke rain, we can conclude that the dependence between the wetness of the floor and the rain is varying. Therefore we can eliminate it from the causes of the rain.

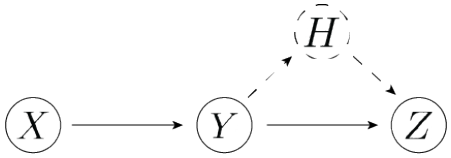

Figure 2. Graphical model of X, Y, Z as presented in the introduction

Let us consider the simple linear example presented in [1] that will be the basis of some of the experiments later on. It also serves as a motivating example. We consider the following simple structural equation model :

$$X := \mathcal{N}(0, \sigma^2) \tag{1}$$
$$Y := X + \mathcal{N}(0, \sigma^2) \tag{2}$$
$$Z := Y + \mathcal{N}(0, 1) \tag{3}$$

Where we want to predict Y when given X and Z as input. [1] consider a case where the data comes from different environements e. Each of which has a different σ standard deviation of the noise. It might help to think of these environments as the different cities from which the data comes. We have a closed form solutions for the linear regression $Y^e = \alpha_1 X^e + \alpha_2 Z^e$. [1] The derivation is detailed in Appendix A. We have three cases :

- If we regress Y over X we would have $\hat{\alpha_1} = 1$ and $\hat{\alpha_2} = 0$
- If we regress Y over Z we would have $\hat{\alpha_1} = 0$ and $\hat{\alpha_2} = \frac{\sigma^2}{\sigma^2 + 1/2}$
- If we regress Y over X and Z we would have $\hat{\alpha_1} = \frac{1}{\sigma^2+1}$ and $\hat{\alpha_2} = \frac{\sigma^2}{\sigma^2+1}$

The only case where the regression coefficents don't depend on the environement is the first one. It corresponds to the invariant feature X. Equation 3 in the structural equation model doesn't affect the relatio between Y and its parents. It is therefore conceivable that it can change. For instance, in a test environment, Z can be arbitrary. If a tsunami hits a city, the floor will be very wet without the necessary presence of rain. One can therefore have an arbitrarily large expected risk by driving Z to $+\infty$ So $\hat{\alpha_1} = 1$ and $\hat{\alpha_2} = 0$ is

optimal in a minimax sense when one allows interventions that don't affect Y with relation to its parents. Minimizing the empirical risk over many environements will generally yield solutions that depend on the environement. And that is no matter the amount of data that we put in. So we can always find a way to break the learned classifier or regressor. If we want to generalize well in these cases, we need to explicitely search for invariant solutions.

3. Causal Discovery through Statistical Invariance

The previous simple example illustrates how optimal statistical solutions (regressing over both X and Z) are only optimal for a fixed joint distribution. It also shows one aspect that could help us find solutions that will generalize to different but causally related distributions. This key aspect is invariance. As we saw, the optimal minimax solution was the one that didn't depend on the environment on which it was trained.

3.1. Invariant Causal Prediction

Invariant Causal Prediction[12] is a statistical framework that formalizes this idea in the case where there are no hidden variables. They try to select a subset of the input features that are causal with respect to the output.

Given $e \in \mathcal{E}$ an environment and $X^e = (X_1^e, \dots, X_d^e)$ the predictor variables that we observe, we want to predict Y^e. ICP introduces the following hypothesis :

$$H_{0,S}(\mathcal{E}) : \begin{array}{l} \text{There exists } g : \mathbb{R}^{|S|} \times \mathbb{R} \to \mathbb{R} \text{ and } F_\varepsilon \\ \text{such that for all } e \in \mathcal{E} Y^e = g(X_S^e, \varepsilon^e) \\ \text{with } \varepsilon^e \sim F_\varepsilon \text{ and } X_S^e \perp\!\!\!\perp \varepsilon^e \\ \text{where } X_S^e = (X_i^e)_{i \in S} \end{array}$$

This hypothesis tells us that there exists a subset of features that can be used to predict Y^e independently of e. Then it is assumed that there is a true *causal* set S^* for which $H_{0,S^*}(\mathcal{E})$ is true for all possible environments $\mathcal{E}$. If $H_{0,S}(\mathcal{E})$ is true for a set $S \subseteq \{1, \dots d\}$, Y_e is a function of these variables that doesn't depend on e. Therefore these variables in S are invariant with respect to $\mathcal{E}$. If $H_{0,S}(\mathcal{E})$ is true for a $\mathcal{E}$ that we currently have access to, maybe it is for a larger $\mathcal{E}'$. Therefore, the variables in S are good *causal* candidates.

$$S(\mathcal{E}) = \bigcap_{S : H_{0,S}(\mathcal{E}) \text{ is true}} S$$

By construction $S(\mathcal{E}) \subseteq S^*$. When $\mathcal{E}$ grows, there are less sets S for which $H_{0,S}(\mathcal{E})$ is true. $S(\mathcal{E})$ then shrinks around S^*. This reflects the fact that with more data from different environments, we can better identify the true causal predictors S^*

ICP estimates $S(\mathcal{E})$. To do that, it develops hypothesis tests for $H_{0,S}(\mathcal{E})$. Then it keeps the intersection of all the sets that the hypothesis test doesn't reject. The proposed tests are of two kinds :

1. Regress Y^e on X^e on each e. Then test if the resulting functions are the same.
2. Regress Y on X (the whole data). Then test if the residual errors on each e are the same.

This approach introduces important concepts but it remains impractical when one is faced with many input features as it scales poorly. It also has a hard time taking into acccount hidden variables. In the case of computer vision for example, there are too many input features for this methods to be practical. And there are necessarily hidden variables that we don't direclty observe such as the position of different objects.

3.2. Invariant Risk Minimization

Another notable method that is inspired from the previous one and that this work will try to improve on is Invariant Risk Minimization [1]. This method also utilizes the availability of a set of training environments $\mathcal{E}_{tr}$. It will use the subtle changes in dependence in these environements to detect spurious features that might be very correlated with the target variable but won't generalize well. For example: the wet floor to predict rain. To do this they take a different route than ICP. Instead of selecting among the input features, they try to build a invariant representation Φ (potentially non linear). In deep learning, one minimizes an emperical risk : $R(f) = E[\ell(f(X), Y)]$ where ℓ is a loss function (cross entropy or square error for example). When there are multiple environements, the risk also depends on the environement $R^e(f) = E[\ell(f(X^e), Y^e)]$. Invariant Risk Minimization splits f into two parts, a representation Φ and a classifier w such that $f = w \circ \Phi$. They argue that for Φ to be invariant, there needs to exist a classifier w that is optimal for all environments simultaneously.

$$\forall e \in \mathcal{E}_{tr} \quad w^* = \arg\min_w R^e(w \circ \Phi)$$

The optimal classifier (in the case of quadratic loss) is the conditional expectation $E[Y^e | \Phi(X^e) = h]$. If the conditional expectation is the same for all environements, it means that the representation has only extracted information from the input that doesn't depend on the environement. IRM thus proposes to solve the following bilevel optimization problem :

$$\begin{array}{ll} \min_{\Phi : \mathcal{X} \to \mathcal{H}} & \sum_{e \in \mathcal{E}_{\text{tr}}} R^e(w \circ \Phi) \\ \text{subject to} & w \in \arg\min_{\bar{w} : \mathcal{H} \to \mathcal{Y}} R^e(\bar{w} \circ \Phi), \text{ for all } e \in \mathcal{E}_{\text{tr}}. \end{array}$$

This is a very hard optimization problem that can hardly be used in practice. Therefore, they transform it into a relaxed version that is more practical to solve. Another issue is that

this problem is ill posed in the sense that there are many Φ and w that represent the same final solution. It suffices for example to take an invertible mapping Ψ and one has $\Phi' = \Psi^{-1} \circ \Phi$ and $w' = w \circ \Psi$ that are also optimal if Ψ and w are optimal. To get around this, one can fix any w and let Φ change such that w becomes optimal for all environements. This is possible since Φ is an arbitrary mapping. Since we can fix w to any function, we might as well make it a linear mapping. With the w now linear and fixed, if we have a convex loss function, we can check if w is optimal for each environement by checking that the gradient w.r.t. w is 0. The problem can thus be seen as :

$$\begin{array}{ll} \min_{\Phi:\mathcal{X}\to\mathcal{H}} & \sum_{e\in\mathcal{E}_{\text{tr}}} R^e(w\circ\Phi) \\ \text{subject to} & \nabla_w R^e(w\circ\Phi) = 0 \end{array}$$

We can further simplify the problem by making it regularized instead of constrained.

$$\min_{\Phi:\mathcal{X}\to\mathcal{H}} \sum_{e\in\mathcal{E}_{\text{tr}}} R^e(w\circ\Phi) + \lambda\|\nabla_w R^e(w\circ\Phi)\|$$

With this formulation [1] successfully train a linear representation on the problem presented in the introduction. They also successfully apply it to a colored MNIST dataset with spurrious correlations between color and number. We refer the reader to the original article for more details on the results.

3.3. Limitations and Related Work

This framework has generated a lot of interest in developping algorithms for finding invariant representations. Many papers point out some of the flaws of IRM and try to build on them [2, 4, 8, 15]. A proper review of the papers citing this work would need a standalone article. We leave this to future work. We will limit ourselves to the following article that is of particular intersest to this report.

[15] studies IRM theoretically in a case where the latent variables are gaussian. They show that in the linear case, one necessarily needs strictly more environments than there are dimensions of the spurrious latent features. In that case IRM recovers a truly invariant representation across all possible interventions. This guarrantee is quite powerful as it ensures generalization and extrapolation well beyond just the scope of the training distribution. This paper criticizes IRM however by pointing out that the number of environements needs to scale linearly with the dimension of spurrious features and is thus quite limitting. They also study IRM in the non linear case but this time using the IRMv1 loss. They show the existence of a representation Φ_ϵ that is invariant on an arbitrarily large proportion of the training data. This representation however is far from being invariant everywhere. When one changes the distribution far enough from the training distribution, the predictions of Φ_ϵ are identical to those of a representation trained only on the empirical risk. This representation also has a lower loss value than the true invariant solution which makes it a more attractive one for the optimization process. It is a major limitation since it means that what may seem like an invariant representation is actually far from being one.

4. Adaptive Regularization : Proposal and experiments

This last heavy limitation has been exposed in the case where the IRMv1 loss is used. They use the fact that one can make the IRM penalty arbitrarily small while also making the empirical risk smaller by using spurrious features. Since the regularization constant λ is fixed, there is a point at which the added penalty is smaller than the decrease in the average empirical risk.

Another problem we've witnessed with IRM is its sensitivity to initialization. The representation $\Phi = 0$ always has an IRM penalty of 0. The IRM penalty thus has a minimum at 0 that attracts gradient descent. The original paper claims that the mean expected risk term will drive the representation away from this point. However we found in our experiments that when initialized too close to 0, the representation Φ tends to gravitate to it. (See figure 3) One has to find a hard sweetspot for the regularization term λ. If it is slightly too high, the representation almost always goes to 0. If it is slightly too low, the solution isn't really invariant.

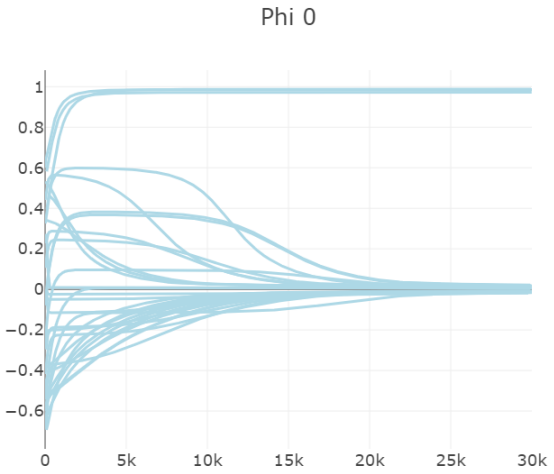

Figure 3. Launched 32 runs with IRMv1 loss with different seeds. This plot shows the evolution of the first component of Φ that corresponds to the invariant feature. 29 runs converge to the 0 representation. 3 found the invariant representation.

4.1. Experiment Setup

The experiments in this paper are based on the simple scenario described in section 2. We consider a training dataset with two environements with standard deviations $\sigma_1 = 0.1$ and $\sigma_2 = 1.5$. Each environement has 5000 samples. There are no minibatches during optimization. We train with the full 10000 samples at each epoch. The Adam optimizer is used with a learning rate of 10^{-3}. The cost function used is the squared error : $x, y \mapsto (x-y)^2$ We use a linear representation $\Phi(x, z) = \Phi_0 x + \Phi_1 z$. We will denote the application and the vector $(\Phi_0, \Phi_1)^T$ as the same quantity Φ. The linear classifier w is the fixed constant 1. Thus $w \circ \Phi(x, z) = w\Phi(x, z) = \Phi(x, z)$ Although in the forward pass the classifier doesn't change anything, it is used later to compute the IRM penalty term $\|\nabla_w R^e(w \circ \Phi)\|$. We run all the experiments below 32 times with different seeds. However, these 32 seeds stay the same between experimetns so that the initializations of Φ are the same for all experiments. Note that since the spurrious variable is of dimension 1, IRM can theoretically recover the invariant representation with just 2 different environements. [15] To test the attained solutions, similarily to [2], we create a test set from the same environements as training but we shuffle the spurrious variable across the samples. This gets rid of the correlation between this variable and the target variable. If a representation uses this spurrious variable, its mean squared error will be very high on the test set.

4.2. Proposal and Results

We propose to use an adaptive regularization scheme. Instead of tuning the λ parameter and fixing it throughout the whole training, we begin by fixing λ to a small value (for example 1). This lets gradient descent take the solution away from the null representation at first. Once the parameters' gradients have a norm below a certain threshold T ($T = 10^{-5}$ in our experiments), we update the regularization constant by multiplying it by M (in our experiments $M = 10$). This changes the optimization landscape by making the IRM penalty more prominent. See algorithm 1 for the structure of the algorithm. Figure 4 shows the evolution of solutions through the loss landscape using this adaptive regularization compared to the vanilla IRMv1 loss.

This simple modification results in much better convergence to the invariant solution, although it's still far from perfect. See figure 5.

However, this optimization procedure is limited for the following reasons. First the threshold heavily depends on the data and the architecture of the neural network. In these simple linear toy examples, it is easy to check for the convergence and tune the threshold. However, in a more complex setting, it is unrealistic to put a threshold on the gradient of all the parameters to measure convergence. Second, this imposes uniformly on the whole network an exponentially growing penalty term. This thus assumes that different parts of the network converge at the same speed.

Algorithm 1: Simple Adaptive Regularization

Input: λ_{init}, T, M
Output: Φ
Data: Training data

$\lambda := \lambda_{init}$
while *Current Iteration* $<$ *Max Iterations* **do**
  Perform gradient descent step on $R^e(w \circ \Phi) + \lambda \|\nabla_w R^e(w \circ \Phi)\|$
  if $\|\Phi\| \leq T$ **then**
    $\lambda := M\lambda$

We improve on this simple adaptive regularization scheme in two ways. First, we keep a exponential average of the gradient squares, as is done in Adam [7]. We then apply the thresholding on this quantity. The exponential average of the squares approximate the second order of the parameter gradients. [3] It is thus much more reliable to put a threshold on this.

Second, we apply a regularization constant for each parameter seperatly. This is done by regularizing on the gradients of each parameter instead of combining the two losses. See algorithm 2

Algorithm 2: Adaptive Regularization

Input: λ_{init}, T, M, β
Output: Φ
Data: Training data

$\lambda := \lambda_{init} * \text{ones\_like}(\Phi)$
while *Current Iteration* $<$ *Max Iterations* **do**
  $g_e = \nabla_\Phi R^e(w \circ \Phi)$
  $g_p = \nabla_\Phi \|\nabla_w R^e(w \circ \Phi)\|$
  $g = g_e + \lambda * g_p$
  $e = \beta e + (1 - \beta) g^2$
  for *i such that* $e[i] < T$ **do**
    $\lambda[i] := M\lambda[i]$

This decouples the network parameters and allows for more fine grained thresholding and penalization. First of all, it makes the algorithm much less sensitive to the choice of the threshold. As long as it is small enough, the training is relatively stable. Also, if parts of the network start converging to a spurrious sub representation, the penalty is exponentially increased to prevent this. The other parts of the network that are still moving heavily towards the mean empirical risk optimal point continue to do so. Figure 6 shows the drastic improvement in convergence to the invariant solution. Figure 7 shows the evolution of these decoupled regularization constants throughout training for the 32 different

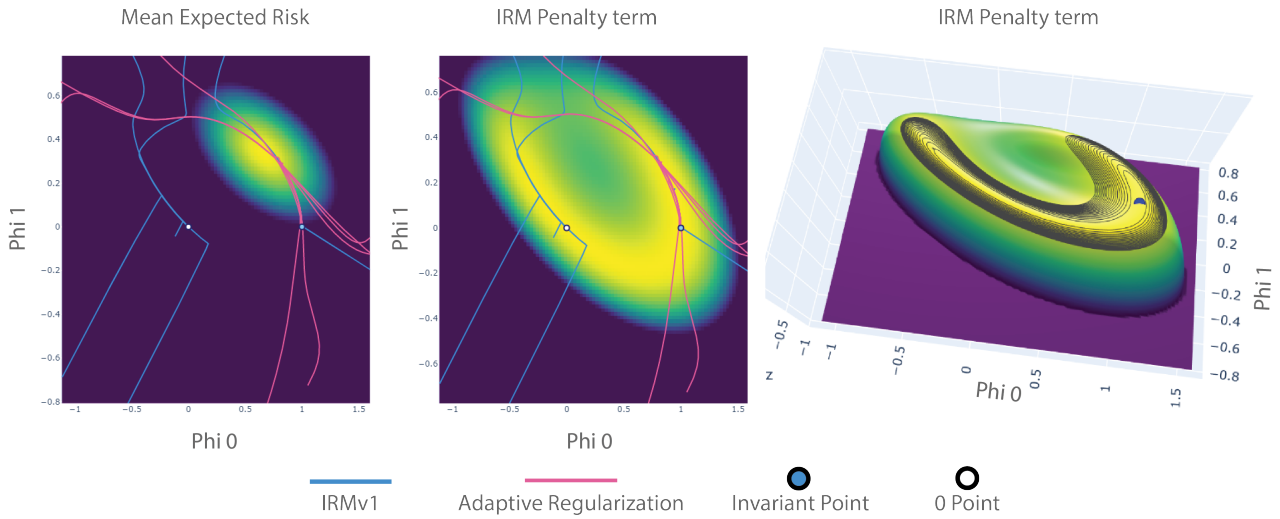

Figure 4. Visualization of the loss landscape in the simple linear example presented above. On the left is the mean emperical risk. It is a convex function with a minimum around (0.6,0.3). On the middle and right is the IRM penalty term visualized in two different ways. The right plot is to emphasize the non convexity of the penalty term. What is shown is actually clipped$(-\text{penalty})$ for visualization purposes. This penalty term is non convex even for the simple linear example presented here. It has multiple minima, the notable ones being (0,0) and (1,0). The latter corresponds to the invariant solution we wish to find. The lines in the left and middle plots present the evolution of multiple runs. The blue lines represent the evolution of Φ with the $IRMv1$ loss. When Φ is initialized on the left of (0,0), the optimization converges to (0,0). The pink lines represent optimization using the adaptive regularization scheme. Notice that they seek out the minimum mean empirical risk at first then continue on to the invariant solution. They thus avoid the null representation.

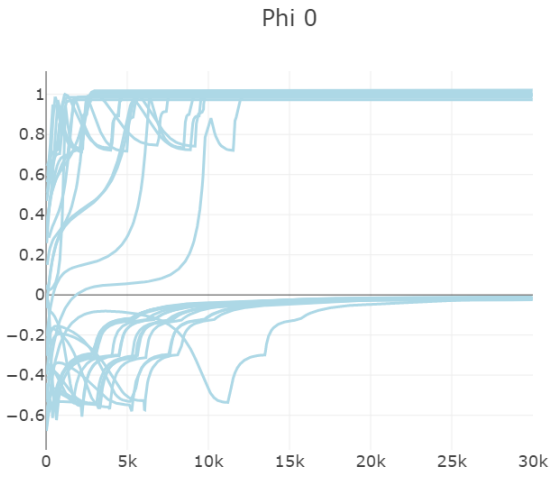

Figure 5. Launched 32 runs with IRMv1 loss with a simple adaptive regularization scheme with the same seeds as figure 3. This plot shows the evolution of the first component of Φ that corresponds to the invariant feature. Half of the runs converge to the invariant representation

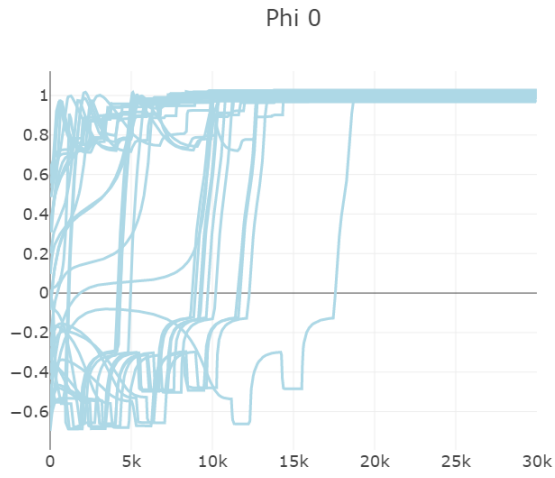

Figure 6. Launched 32 runs with IRMv1 loss with adaptive regularization with the same seeds as figure 3. This plot shows the evolution of the first component of Φ that corresponds to the invariant feature. All runs converge to the invariant representation

runs.

This adaptive regularization scheme also adresses one of the limitations of IRM mentioned in [15]. Indeed, by letting the regularization parameter grow arbitrarily, the IRM penalty will never become negligible in comparison with the empirical error. We don't know however if the solution reached with this scheme is necessarily invariant. Future work will adress this issue.

To quantify how these improvements perform in a test scenario we added two experiments. The first one is just

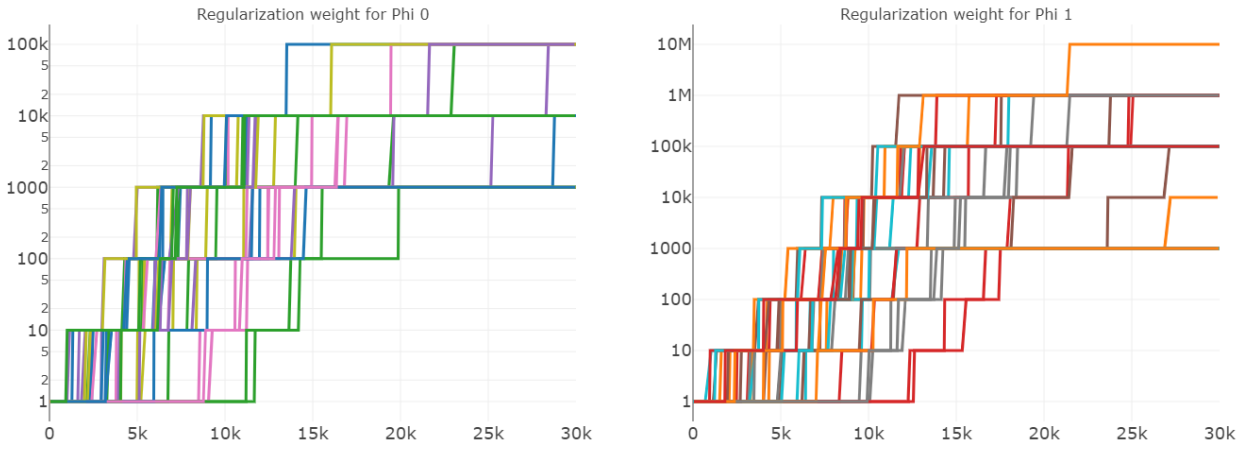

Figure 7. Same experiment as figure 6. This plot shows the evolution of the regularization constants for Φ_0 and Φ_1. We can see that the regularization constants increase in steady intervals. This optimization can be thus seen as a series of penalized problems that converge to the original constrained problem in IRM. The parameter Φ_1 that corresponds to the spurious feature in this problem is much more heavily penalized than Φ_0. This parameter is thus forced to go to 0. Decoupling the regularization of the two parameters made for a more flexible optimizaiton process.

training a linear regression using gradient descent. We refer to this experiment as ERM for Expected Risk Minimization. The second one is also a linear regression but this time, the spurrious variables are shuffled as is done in the test set. This operation is of course not possible in a real scenario where we don't know which variables are spurrious. But it serves as a kind of upper bound of the performance. We refer to this experiment as "oracle". This is similar to what is done in [2], where they expand more on these simple linear unit tests and benchmark IRM against other algorithms. Figure 8 is a box plot summarizing the test results on the 32 runs for each experiment.

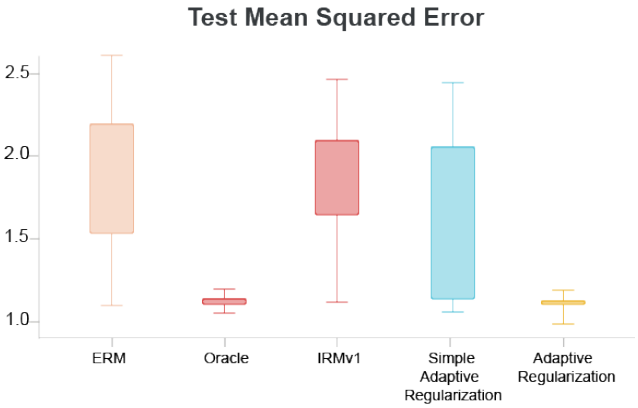

Figure 8. Test results. We see that IRMv1 here performs rather poorly in average, barely better than ERM. It should be noted that the hyperparmeters were not extensively tuned in order to simulate a real world case where a full hyperparameter search is not always possible. This result is thus more about the sensitivity of IRMv1 to initialization and the regularization hyperaparameters rather than its true potential performance. The simple adaptive regularization scheme improves on IRM by making more of the solutions converge to the invariant one. Adaptive Regularization as defined before performs the best, with a test score nearly identical to the oracle.

5. Experiment Code

One of the major contributions of this project was the design of very modular experimental code. Since this is work that will continue beyond the scope of this project, the code was made to be fully customizable with modules that can be very easily swapped out. The code was built from the ground up, using pytorch lightening and hydra. We encourage the reader to refer to the code () as more details are present there.

6. Discussion and Future Work

This project report briefly introduces some causal inference notions along with why it is an important field of research especially in deep learning. A notion that stands out in causality is that of invariance. Previous work exploits the invariance of causal mechanisms to try to infer them. We explain some of this work and analyze it critically. We then propose an optimization procedure to find the invariant representation discussed in IRM. We test this optimization procedure on a simple linear case. While these experi-

ments are toy examples with little relation to deep learning for now, they help us understand better how these methods work and how to improve on them. Although the time frame of the project hasn't allowed for a more detailed analysis of the non linear case, the code for these experiments is already implemented and some experiments have already taken place. So naturally the next steps in this project is to investigate the non linear case and see if the proposed optimization method scales well and what modifications to it will be needed. A major limitation that hasn't been addressed here (since it wasn't a problem) is numerical stability. Making the regularization constant exponentially bigger will lead to NaNs being generated at some point.

A. Derivation of linear regression solutions

We are trying to find α_1 and α_2 such that they minimize the following :

$$\min_{\alpha_1,\alpha_2} E[(Y - (\alpha_1 X + \alpha_2 Z))^2]$$

This is a classic linear regression problem that we know admits a solution. By using the structural equation model we can replace Y and Z with their expressions.

$$\begin{aligned}
&E[(Y - (\alpha_1 X + \alpha_2 Z))^2] \\
&= E[(X + N_1 - (\alpha_1 X + \alpha_2 (X + N_1 + N_2)))^2] \\
&= E[((1 - \alpha_1 - \alpha_2)X + (1 - \alpha_2)N_1 - \alpha_2 N_2)^2]
\end{aligned}$$

Since N_1, N_2 and X are centered independent gaussian random variables we know that

$$\begin{aligned}
&(1 - \alpha_1 - \alpha_2)X + (1 - \alpha_2)N_1 - \alpha_2 N_2 \\
&\sim \mathcal{N}(0, (1 - \alpha_1 - \alpha_2)^2\sigma^2 + (1 - \alpha_2)^2\sigma^2 + \alpha_2^2)
\end{aligned}$$

Since $(1 - \alpha_1 - \alpha_2)X + (1 - \alpha_2)N_1 - \alpha_2 N_2$ is centered, its variance is equal to its second moment. Therefore,

$$E[(Y-(\alpha_1 X+\alpha_2 Z))^2] = (1-\alpha_1-\alpha_2)^2\sigma^2+(1-\alpha_2)^2\sigma^2+\alpha_2^2$$

The problem here being convex (quadratic minimization problem) we just need to find a critical point. The gradient of the expression above is

$$\begin{aligned}
&\nabla E[(Y - (\alpha_1 X + \alpha_2 Z))^2] \\
&= \begin{pmatrix} 2(\alpha_1 + \alpha_2 - 1)\sigma^2 \\ 2(\alpha_1 + \alpha_2 - 1)\sigma^2 + 2(\alpha_2 - 1)\sigma^2 + 2\alpha_2 \end{pmatrix}
\end{aligned}$$

Solving for $\nabla E[(Y - (\alpha_1 X + \alpha_2 Z))^2] = 0$ is a linear system that gives us $\alpha_1 = \frac{1}{1+\sigma^2}$ and $\alpha_2 = \frac{\sigma^2}{1+\sigma^2}$

Fixing $\alpha_1 = 0$ or $\alpha_2 = 0$ leaves us with a 1D convex minimization problem. Similar (easier) arguments can be made in those cases.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization. 2, 3, 4

[2] Benjamin Aubin, Agnieszka Słowik, Martin Arjovsky, Leon Bottou, and David Lopez-Paz. Linear unit-tests for invariance discovery. page 6. 4, 5, 7

[3] Lukas Balles and Philipp Hennig. Dissecting Adam: The Sign, Magnitude and Variance of Stochastic Gradients. 5

[4] Yo Choe, Jiyeon Ham, and Kyubyong Park. An Empirical Study of Invariant Risk Minimization. 4

[5] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification Presents Challenges for Credibility in Modern Machine Learning. 1

[6] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut Learning in Deep Neural Networks. 1

[7] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. 5

[8] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Remi Le Priol, and Aaron Courville. Out-of-Distribution Generalization via Risk Extrapolation (REx). 4

[9] David Lopez-Paz. From Dependence to Causation. 1

[10] Judea Pearl. *Causality.* Cambridge University Press, 2 édition edition. 2

[11] Judea Pearl and Dana Mackenzie. *The Book of Why: The New Science of Cause and Effect*. Penguin UK. 2

[12] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: Identification and confidence intervals. 78(5):947–1012. 3

[13] Jonas Peters, Dominik Janzing, and Bernhard Scholkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press. 2

[14] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. Gradient Starvation: A Learning Proclivity in Neural Networks. 1

[15] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The Risks of Invariant Risk Minimization. 4, 5, 6

[16] Bernhard Schölkopf. Causality for Machine Learning. 1

[17] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The Pitfalls of Simplicity Bias in Neural Networks. 1