

---

# Adversarial Training of Neural Networks against Systematic Uncertainty

---

Gilles Louppe  
New York University  
g.louppe@nyu.edu

## Abstract

## 1 Introduction

[GL: Distinction between statistical and systematic uncertainty.] [GL: Define nuisance parameters.] [GL: We want to build an accurate classifier whose output remains invariant with respect to systematic uncertainties.] [GL: Motivate the criterion (which may not be obvious for the ML crowd). See pivotal quantity motivation.]

## 2 Problem statement

Let assume a probability space  $(\Omega, \mathcal{F}, P)$ , where  $\Omega$  is a sample space,  $\mathcal{F}$  is a set of events and  $P$  is a probability measure. Let consider the multivariate random variables  $X_\lambda : \Omega \mapsto \mathbb{R}^p$  and  $Y : \Omega \mapsto \mathcal{Y}$ , where  $X_\lambda$  depends on a nuisance parameter  $\lambda \in \Lambda$  whose values define a parameterized family of its systematic uncertainties. That is,  $X_\lambda$  and  $Y$  induce together a joint probability distribution  $p(X, Y|\lambda)$ , where the conditional on  $\lambda$  denotes  $X_\lambda$ . For training, let further assume a finite set  $\{x_i, y_i, \lambda_i\}_{i=1}^N$  of realizations  $X_{\lambda_i}(\omega_i)$ ,  $Y(\omega_i)$ , for  $\omega_i \in \Omega$  and known values  $\lambda_i$  of the nuisance parameter. Our goal is to learn a function  $f(\cdot; \theta_f) : \mathbb{R}^p \mapsto \mathcal{Y}$  of parameters  $\theta_f$  (e.g., a neural network-based classifier if  $\mathcal{Y}$  is a finite set of classes) and minimizing a loss  $\mathcal{L}_f(\theta_f)$ . In addition, we require that  $f(X_\lambda; \theta_f)$  should be robust to the value of the nuisance parameter  $\lambda$  – which remains unknown at test time. More specifically, we aim at building  $f$  such that in the ideal case

$$f(X_\lambda(\omega); \theta_f) = f(X_{\lambda'}(\omega); \theta_f) \quad (1)$$

for all samples  $\omega \in \Omega$  and all  $\lambda, \lambda'$  pairs of values of the nuisance parameter.

Since we do not have training tuples  $(X_\lambda(\omega), X_{\lambda'}(\omega))$  (for the same unknown  $\omega$ ), we propose instead to solve the closely related problem of finding a predictive function  $f$  such that

$$P(\{\omega | f(X_\lambda(\omega); \theta_f) = y\}) = P(\{\omega' | f(X_{\lambda'}(\omega'); \theta_f) = y\}) \text{ for all } y \in \mathcal{Y}. \quad (2)$$

In words, we are looking for a predictive function  $f$  which is a pivotal quantity [1] with respect to the nuisance parameter. That is, such that the distribution of  $f(X_\lambda; \theta_f)$  is invariant with respect to the value of  $\lambda$ . Note that a function  $f$  for which Eqn. 1 is true necessarily satisfies Eqn. 2. The converse is however in general not true, since the sets of samples  $\{\omega | f(X_\lambda(\omega); \theta_f) = y\}$  and  $\{\omega' | f(X_{\lambda'}(\omega'); \theta_f) = y\}$  do not need to be the same for the equality to hold. In order to simplify notations, and as only Eqn. 2 is of direct interest in this work, we denote from here on the pivotal quantity criterion as

$$p(f(X; \theta_f) | \lambda) = p(f(X; \theta_f) | \lambda') \text{ for all } \lambda, \lambda' \in \Lambda. \quad (3)$$

### 3 Method

Adversarial training was first proposed by [2] as a way to build a generative model capable of producing samples from random noise  $z \sim p_Z$ . More specifically, the authors pit a generative model  $g : \mathcal{Z} \mapsto \mathbb{R}^p$  against an adversary classifier  $d : \mathbb{R}^p \mapsto \{0, 1\}$  whose antagonistic objective is to recognize real data  $X$  from generated data  $g(Z)$ . Both models  $g$  and  $d$  are trained simultaneously, in such a way that  $g$  learns to produce samples that are difficult to identify by  $d$ , while  $d$  incrementally adapts to changes in  $g$ . At the equilibrium,  $g$  models a distribution whose samples can be identified by  $d$  only by chance. That is, assuming enough capacity in  $d$  and  $g$ , the distribution  $p_{g(Z)}$  eventually converges towards the real distribution  $p_X$ .

In this work, we repurpose adversarial training as a means to constraint the predictive model  $f$  in order to satisfy Eqn. 3. In particular, we pit  $f$  against an adversary classifier  $r(\cdot; \theta_r) : \mathbb{R} \mapsto \Lambda$  of parameters  $\theta_r$  and associated loss  $\mathcal{L}_r(\theta_f, \theta_r)$ . Assuming that  $\Lambda$  defines a finite family of nuisance values  $\lambda_l$  (for  $l = 1, \dots, |\Lambda|$ ), this classifier takes as input realizations of  $f(X; \theta_f)$ , for the current value  $\theta_f$  of  $f$  parameters, and produces as output probability estimates  $r(f(X; \theta_f); \theta_r)_l = \hat{p}(\lambda_l | f(X; \theta_f))$  that  $f(X; \theta_f)$  is generated from the nuisance value  $\lambda_l$ . If  $p(f(X; \theta_f) | \lambda)$  varies with  $\lambda$ , then the corresponding correlation can be captured by  $r$ . By contrast, if  $p(f(X; \theta_f) | \lambda)$  is invariant with  $\lambda$ , as we require, then  $r$  should perform poorly and be close to random guessing. Training  $f$  such that it additionally minimizes the performance of  $r$  therefore acts as a regularization towards Eqn. 3.

As for generative adversarial networks, we propose to train  $f$  and  $r$  simultaneously, which we carry out by considering the value function

$$E(\theta_f, \theta_r) = \mathcal{L}_f(\theta_f) - \mathcal{L}_r(\theta_f, \theta_r) \quad (4)$$

that we optimize by finding the saddle point  $(\hat{\theta}_f, \hat{\theta}_r)$  such that

$$\hat{\theta}_f = \arg \min_{\theta_f} E(\theta_f, \hat{\theta}_r), \quad (5)$$

$$\hat{\theta}_r = \arg \max_{\theta_r} E(\hat{\theta}_f, \theta_r). \quad (6)$$

The adversarial training procedure to obtain  $(\hat{\theta}_f, \hat{\theta}_r)$  is formally presented in Algorithm 1 in the case of  $f$  being a classifier and of the cross-entropy loss for both  $\mathcal{L}_f$  and  $\mathcal{L}_r$ . The algorithm consists in using stochastic gradient descent alternatively to optimize Eqn. 5 and 6.

---

**Algorithm 1** Adversarial training of a classifier  $f$  against an adversary  $r$ .

*Inputs:* training data  $\{x_i, y_i, \lambda_i\}_{i=1}^N$

*Outputs:*  $\hat{\theta}_f, \hat{\theta}_r$

*Hyper-parameters:* Number  $T$  of training iterations, Number  $K$  of gradient steps to update  $r$ .

---

- 1: **for**  $t = 1$  to  $T$  **do**
- 2:   **for**  $k = 1$  to  $K$  **do** ▷ Update  $r$
- 3:     Sample minibatch  $\{x_m, \lambda_m\}_{m=1}^M$  of size  $M$ ;
- 4:     With  $\theta_f$  fixed, update  $r$  by ascending its stochastic gradient  $\nabla_{\theta_r} E(\theta_f, \theta_r) :=$

$$\nabla_{\theta_r} \sum_{m=1}^M \left[ \sum_{\lambda_l \in \Lambda} 1(\lambda_m = \lambda_l) \log r(f(x_m; \theta_f); \theta_r)_l \right];$$

- 5:   **end for**
- 6:     Sample minibatch  $\{x_m, y_m, \lambda_m\}_{m=1}^M$  of size  $M$ ; ▷ Update  $f$
- 7:     With  $\theta_r$  fixed, update  $f$  by descending its stochastic gradient  $\nabla_{\theta_f} E(\theta_f, \theta_r) :=$

$$\nabla_{\theta_f} \sum_{m=1}^M \left[ - \sum_{y_c \in \mathcal{Y}} 1(y_m = y_c) \log f(x_m; \theta_f)_c + \sum_{\lambda_l \in \Lambda} 1(\lambda_m = \lambda_l) \log r(f(x_m; \theta_f); \theta_r)_l \right];$$

- 8: **end for**
-

## 4 Theoretical results

In this section, we show that in the setting of Algorithm 1, the procedure converges to a classifier  $f$  which is a pivotal quantity in the sense of Eqn. 3. Results below are derived in a non-parametric setting, by assuming that both  $f$  and  $r$  have enough capacity. To simplify the presentation, we also assume the uniform prior  $p(\lambda) = \frac{1}{|\Lambda|}$  for all  $\lambda \in \Lambda$ , e.g. by having the same number of training samples for each modality  $\lambda$  of the nuisance parameter.

**Proposition 1.** *Let  $\theta_f$  be fixed and  $\hat{\theta}_r = \arg \max_{\theta_r} E(\theta_f, \theta_r)$ . If  $r(f(X; \theta_f); \hat{\theta}_r)_l = p(\lambda_l | f(X; \theta_f)) = \frac{1}{|\Lambda|}$  for all  $\lambda_l$ , then  $f$  is a pivotal quantity.*

[GL: proof 1: if  $r$  optimal and  $r(f(x)) = 1/N$  then  $f$  is pivotal. [OK]] [GL: proof 2: if  $L_f$  can be minimized under the pivotal constraint, then at the saddle point  $r$  is such that  $r(f(x)) = 1/N$ . [OK]]

## 5 Experiments

## 6 Related work

[GL: Similar to domain adaptation, but with infinitely many domains, as parameterized by  $\lambda$ , also related to transfer learning.]

## 7 Conclusions

### Acknowledgments

### References

- [1] M. H. Degroot and M. J. Schervish, *Probability and statistics*. 4 ed., 2010.
- [2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.