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: allow to derive guarantees no matter λ, while f may otherwise be good or very bad depending on the nuisance if trained on the mixture. E.g. to form hypothesis to be later confirmed by data.]

2 Problem statement

Let assume a probability space $(\Omega, \mathcal{F}, P)$, where Ω 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_λ depends on a nuisance parameter $\lambda \in \mathbb{R}$ whose values define a continuously parameterized family of its systematic uncertainties. That is, X_λ and Y induce together a joint probability distribution $p(X, Y|\lambda)$, where the conditional on λ denotes X_λ. 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 λ_i of the nuisance parameter. Our goal is to learn a function $f(\cdot; \theta_f) : \mathbb{R}^p \mapsto \mathcal{Y}$ of parameters θ_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 λ – which remains unknown at test time. More specifically, we aim at building f such that in the ideal case

$$f(X_{\lambda_i}(\omega); \theta_f) = f(X_{\lambda_j}(\omega); \theta_f) \tag{1}$$

for any sample $\omega \in \Omega$ and any λ_i, λ_j pair of values of the nuisance parameter.

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

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

In words, we are looking for a predictive function f such that the distribution of $f(X_\lambda; \theta_f)$ is invariant with respect to the nuisance parameter λ. 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 $\{\omega | f(X_{\lambda_i}(\omega); \theta_f) = y\}$ and $\{\omega' | f(X_{\lambda_j}(\omega'); \theta_f) = y\}$ do not need to be the same for the equality to hold.

[GL: An alternative objective, but which leads to something significantly different, is to aim at getting the best model adapted to the unknown λ, possibly using predictive information from regions of the input space affected by the nuisance parameter.]

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

3 Method

Adversarial training was first proposed by [1] 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 regularize the predictive model f in order to satisfy Eqn. 2. In particular, we pit f against an adversary regression model $r(\cdot; \theta_r) : \mathbb{R} \mapsto \mathbb{R}$ of parameters θ_r and associated loss $\mathcal{L}_r(\theta_r)$. This regressor takes as input realizations of $f(X_\lambda; \theta_f)$ and produces as output predictions $\hat{\lambda} = r(f(X_\lambda; \theta_f))$ of the nuisance parameter. If $p(f(X_\lambda; \theta_f))$ varies with λ, then the corresponding correlation can be captured by r. By contrast, if $p(f(X_\lambda; \theta_f))$ is invariant with λ as we require, then r should perform poorly. Training f such that it additionally minimizes the performance of r therefore acts as a regularization towards Eqn. 2.

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

$$E(\theta_f, \theta_r) = \mathcal{L}_f(\theta_f) - \mathcal{L}_r(\theta_r) \tag{3}$$

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

$$\hat{\theta}_f = \arg\min_{\theta_f} E(\theta_f, \hat{\theta}_r), \tag{4}$$

$$\hat{\theta}_r = \arg\max_{\theta_r} E(\hat{\theta}_f, \theta_r). \tag{5}$$

[GL: Explain this can be done simply using SGD.] [GL: What are the necessary conditions on $\mathcal{L}_r$ to imply Eqn. 2 at the saddle point? We should clarify and prove that formally! There may also be assumptions required on the learning problem itself.]

4 Experiments

5 Related work

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

6 Conclusions

Acknowledgments

References

[1] 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.