An Online Learning Approach to Interpolation and Extrapolation in Domain Generalization

Elan Rosenfeld, Pradeep Ravikumar, Andrej Risteski

Nicole Chiou, Olawale Salaudeen

November 17, 2021

Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Background

OOD

The success of machine learning algorithms assumes that the data are I.I.D., an assumption that is often broken in practice.

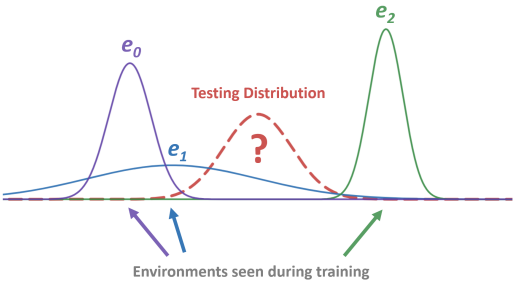
Out-of-Distribution (OOD) generalization: testing distribution is unknown and differs from the training distribution

Background

Domain Generalization

Assumption: training data comprised of distinct distributions or "environments".

Domain Generalization: given several domains, learn a model that can generalize to an unseen test data which relates to the observed domains.



Background

Group Shift

How do future test distributions relate to the groups encountered during training?

- **Assumption**: test distribution is a re-weighting of observed group likelihoods.
- Also called generalization under sub-population/group shift.

Background

Group Shift

How do future test distributions relate to the groups encountered during training?

- **Assumption**: test distribution is a re-weighting of observed group likelihoods.
- Also called generalization under sub-population/group shift.

How can we ensure good out-of-distribution performance?

- **Goal**: minimize risk over worst-case combination of groups.

Motivation

Shortcomings of Prior Works

Prior work lacks formal justification of Empirical Risk Minimization's (ERM) ability to "interpolate" the training distributions and "extrapolate" beyond them.

Motivation

Shortcomings of Prior Works

Prior work lacks formal justification of Empirical Risk Minimization's (ERM) ability to "interpolate" the training distributions and "extrapolate" beyond them.

Other shortcomings of prior work:

1. Lack of theoretical guarantees on replacements to ERM
2. Strictly consider worst-case performance, which is not representative of the real-world

Motivation

Shortcomings of Prior Works

Prior work lacks formal justification of Empirical Risk Minimization's (ERM) ability to "interpolate" the training distributions and "extrapolate" beyond them.

Other shortcomings of prior work:

1. Lack of theoretical guarantees on replacements to ERM
2. Strictly consider worst-case performance, which is not representative of the real-world

There is a need for a robust measure of OOD generalization that allows for a comparison of computational and statistical properties among domain generalization algorithms.

Contributions

Under the notion of inter- and extrapolation based on the re-weighting of sub-groups:

1. Extrapolation is computationally harder than interpolation.
2. The statistical complexity of inter- and extrapolation not too different.

Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Interpolation

Convex Hull

Given a set of E training distributions $\mathcal{E} = \{e_i\}_{i=1}^{E}$, each indexing a probability distribution p^e.

Interpolation

Convex Hull

Given a set of E training distributions $\mathcal{E} = \{e_i\}_{i=1}^{E}$, each indexing a probability distribution p^e. Then an interpolation is any distribution that can be written as

$$p^{\lambda} = \sum_{e \in \mathcal{E}} \lambda_e p^e,$$

Interpolation

Convex Hull

Given a set of E training distributions $\mathcal{E} = \{e_i\}_{i=1}^{E}$, each indexing a probability distribution p^e. Then an interpolation is any distribution that can be written as

$$p^{\lambda} = \sum_{e \in \mathcal{E}} \lambda_e p^e,$$

where $\lambda \in \nabla_E$ is a vector of convex coefficients (∇_E is the $E - a$ simplex)

Interpolation

Convex Hull

Given a set of E training distributions $\mathcal{E} = \{e_i\}_{i=1}^{E}$, each indexing a probability distribution p^e. Then an interpolation is any distribution that can be written as

$$p^{\lambda} = \sum_{e \in \mathcal{E}} \lambda_e p^e,$$

where $\lambda \in \nabla_E$ is a vector of convex coefficients (∇_E is the $E - a$ simplex) - in other words, within the convex hull of training distributions.

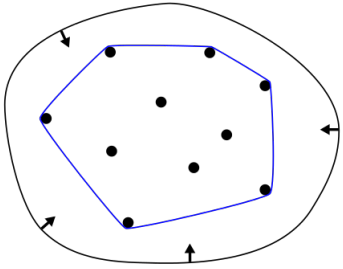


Figure: convex hull

Extrapolation

Bounded Affine Combinations

Recall that $\lambda \in \nabla_E$ is a vector of convex coefficients. The proposed test distributions can be defined as bounded affine combinations of the environments:

$$\sum_{e \in \mathcal{E}} \lambda_e = 1, \lambda_e \geq -\alpha . \forall e \in \mathcal{E}.$$

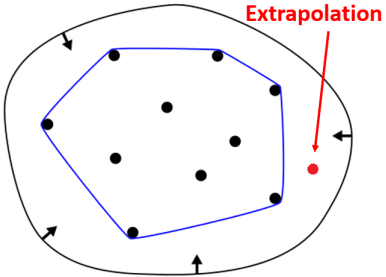
Extrapolation

Bounded Affine Combinations

Recall that $\lambda \in \nabla_E$ is a vector of convex coefficients. The proposed test distributions can be defined as bounded affine combinations of the environments:

$$\sum_{e \in \mathcal{E}} \lambda_e = 1, \lambda_e \geq -\alpha . \forall e \in \mathcal{E}.$$

Note: the resulting function is not guaranteed to be a probability distribution, but can be framed as a re-weighting of the environment *risks*.



Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Recasting Domain Generalization as a Game

Consider Domain Generalization as a continuous game of online learning - with a *player* and *adversary*.

Recasting Domain Generalization as a Game

Consider Domain Generalization as a continuous game of online learning - with a *player* and *adversary*.

Each Round:

1. Player is given a new test environment, and
2. must refine their predictor at each round.

Recasting Domain Generalization as a Game

Consider Domain Generalization as a continuous game of online learning - with a *player* and *adversary*.

Each Round:

1. Player is given a new test environment, and
2. must refine their predictor at each round.

Goal: Adapt better to unseen distributions as more distributions are seen, i.e. $\downarrow$ per-round regret.

Game Preliminaries

1. Let B be a convex set of potential predictors.
   Assume for all $\lambda \in \Delta_E$, β that minimizes p^λ is in B.
2. For any $\beta \in B$ and all $e \in \mathcal{E}$, the expected loss under p^e is finite

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$
- **for** $t = 1 \dots T$ **do**

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$
- **for** $t = 1 \dots T$ **do**
  1. Player chooses $\hat{\beta}_t \in B$

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$
- **for** $t = 1 \dots T$ **do**
  1. Player chooses $\hat{\beta}_t \in B$
  2. Adversary chooses convex coefficients

$$\lambda_t = \{\lambda_{t,e}\}_{e\in\mathcal{E}}, \lambda_{t,e} \geq 0, \forall e \in \mathcal{E}.$$

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$
- **for** $t = 1 \dots T$ **do**
  1. Player chooses $\hat{\beta}_t \in B$
  2. Adversary chooses convex coefficients

$$\lambda_t = \{\lambda_{t,e}\}_{e\in\mathcal{E}}, \lambda_{t,e} \geq 0, \forall e \in \mathcal{E}.$$

  3. Define $f_t(\beta) := \mathbb{E}_{(x,y)\sim p^{\lambda_t}}[\ell(\beta, (x, y))]$, where $p^{\lambda_t} = \sum_{e\in\mathcal{E}} \lambda_{t,e} p^e$.
- **end for**

Game

- **Input:** Convex parameter space B, distributions $\{P_{XY}\}_{e\in\mathcal{E}}$, strongly convex loss $\ell : B \times (\mathcal{X} \times \mathcal{Y})$
- **for** $t = 1 \dots T$ **do**
  1. Player chooses $\hat{\beta}_t \in B$
  2. Adversary chooses convex coefficients

$$\lambda_t = \{\lambda_{t,e}\}_{e\in\mathcal{E}}, \lambda_{t,e} \geq 0, \forall e \in \mathcal{E}.$$

  3. Define $f_t(\beta) := \mathbb{E}_{(x,y)\sim p^{\lambda_t}}[\ell(\beta, (x, y))]$, where $p^{\lambda_t} = \sum_{e\in\mathcal{E}} \lambda_{t,e} p^e$.
- **end for**
- Player suffers regret

$$R_T = \sum_{t=1}^{T} f_t(\hat{\beta}_t) - \min_{\beta\in B} \sum_{t=1}^{T} f_t(\beta).$$

Some Intuition Behind the Game

The proposed online approach allows for improvements over time, rather than overemphasizing minimax performance for single-rounds.

Some Intuition Behind the Game

The proposed online approach allows for improvements over time, rather than overemphasizing minimax performance for single-rounds.

Single-round loss guarantees good performance in the worst-case, but this is not representative of future test environments in the real-world.

Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Goal

When can we achieve sublinear regret rates?

Sublinear regret implies recovery of model with best average over all distributions in limit – minimax.

Recovers invariant predictor if it is indeed the best.

Theoretical Contribution

Regret Rates

They show that if we divide domain generalization into two tasks i) interpolation and ii) extrapolation:

Theoretical Contribution

Regret Rates

They show that if we divide domain generalization into two tasks i) interpolation and ii) extrapolation:

1. the computation complexity difference between the two is exponential, but
2. the statistical complexity (minimax) $\underbrace{\Theta(\log T), \Theta(\sqrt{T})}_{\text{unclear in what sense}}$ respectively, is not too different.

Theoretical Contribution

Regret Rates

They show that if we divide domain generalization into two tasks i) interpolation and ii) extrapolation:

1. the computation complexity difference between the two is exponential, but
2. the statistical complexity (minimax) $\underbrace{\Theta(\log T),\ \Theta(\sqrt{T})}_{\text{unclear in what sense}}$ respectively, is not too different.

Conclude that ERM is minimax for both interpolation and extrapolation.

Lemma 1

Lemma

Recall that $\mathcal{R}^\lambda(\beta)$ is defined as the risk of β on the distribution p^e. Then for all finite λ,

$$R^\lambda(\beta) = \sum_{e \in \mathcal{E}} \lambda_e \mathcal{R}^e(\beta).$$

Lemma 1

Lemma

Recall that $\mathcal{R}^\lambda(\beta)$ is defined as the risk of β on the distribution p^e. Then for all finite λ,

$$R^\lambda(\beta) = \sum_{e \in \mathcal{E}} \lambda_e \mathcal{R}^e(\beta).$$

Takeaway: When an adversary chooses convex coefficients, they are equivalently choosing a loss function f_t that is a convex combination of individual environment risks $\{f_e\}_{e=1}^E$.

Theorem 1

Kreuger et al. (2020) show that the regret rate when an adversary can play arbitrary strongly convex functions of losses is $\mathcal{O}(\log t)$.

Theorem 1

Kreuger et al. (2020) show that the regret rate when an adversary can play arbitrary strongly convex functions of losses is $\mathcal{O}(\log t)$.

Theorem

If we relax the constraint on the adversary, specifically, can choose f_t to be any convex combination of risks, then the minimax optimal regret rate is $\mathcal{O}(\log t)$.

Theorem 1

Kreuger et al. (2020) show that the regret rate when an adversary can play arbitrary strongly convex functions of losses is $\mathcal{O}(\log t)$.

Theorem

If we relax the constraint on the adversary, specifically, can choose f_t to be any convex combination of risks, then the minimax optimal regret rate is $\mathcal{O}(\log t)$.

Takeaway: We may expect that restricting the adversary to playing within the convex hull of a limited number of functions might be expected to affect the asymptotic statistical complexity.

Theorem 1

Kreuger et al. (2020) show that the regret rate when an adversary can play arbitrary strongly convex functions of losses is $\mathcal{O}(\log t)$.

Theorem

If we relax the constraint on the adversary, specifically, can choose f_t to be any convex combination of risks, then the minimax optimal regret rate is $\mathcal{O}(\log t)$.

Takeaway: We may expect that restricting the adversary to playing within the convex hull of a limited number of functions might be expected to affect the asymptotic statistical complexity.

Even with full knowledge of the adversary's limited selection, no algorithm can surpass $\mathcal{O}(\log t)$ – achieved by Follow-The-Leader (ERM).

Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Bounded Affine Combinations

Suppose, instead of only convex combinations, the adversary can play any bounded affine combinations of environments (may lie outside train convex hull, i.e. extraploate).

Bounded Affine Combinations

Suppose, instead of only convex combinations, the adversary can play any bounded affine combinations of environments (may lie outside train convex hull, i.e. extraploate).

Consider an "oblivious" adversary (sequence of loss functions selected at start of game).

For general Lipschitz functions, against an oblivious adversary:

- no deterministic strategy can guarantee sublinear regret,
- attaining sublinear regret with randomized strategy is NP-hard.

Bounded Affine Combinations

Suppose, instead of only convex combinations, the adversary can play any bounded affine combinations of environments (may lie outside train convex hull, i.e. extraploate).

Consider an "oblivious" adversary (sequence of loss functions selected at start of game).

For general Lipschitz functions, against an oblivious adversary:

- no deterministic strategy can guarantee sublinear regret,
- attaining sublinear regret with randomized strategy is NP-hard.

Follow–The–Perturbed–Leader (FTPL) can achieve Information-Theoretic regret lower bound of $\Omega(\sqrt{T})$ (Suggala & Netrapalli, 2020).

Theorem 2

Theorem

No deterministic algorithm can guarantee sublinear regret against bounded affine combinations of a finite set of strongly convex losses.

Theorem 2

Theorem

No deterministic algorithm can guarantee sublinear regret against bounded affine combinations of a finite set of strongly convex losses.

Takeaway: Surprisingly, even with relaxing the general Lipschitz function to bounded affine combinations of strongly convex losses - fully know by the player - the game remains equally hard.

Proof of Theorem 2

Proof.

Part 1/2. We will show that for any deterministic algorithm, there exists a sequence of loss functions with regret bounded by $\mathcal{O}(T)$. Assume $\lambda_e > -\alpha \, \forall e$. Define

$$f_{e_1}(x) = x^2, \qquad f_{e_2} = x^4 + \frac{1}{2\alpha}x^2.$$

Proof of Theorem 2

Proof.

Part 1/2. We will show that for any deterministic algorithm, there exists a sequence of loss functions with regret bounded by $\mathcal{O}(T)$. Assume $\lambda_e > -\alpha \, \forall e$. Define

$$f_{e_1}(x) = x^2, \qquad f_{e_2} = x^4 + \frac{1}{2\alpha}x^2.$$

On round t, the player chooses $x \in \mathbb{R}$.

The loss is as follows (adversary):

Proof of Theorem 2

Proof.

Part 1/2. We will show that for any deterministic algorithm, there exists a sequence of loss functions with regret bounded by $\mathcal{O}(T)$. Assume $\lambda_e > -\alpha \, \forall e$. Define

$$f_{e_1}(x) = x^2, \qquad f_{e_2} = x^4 + \frac{1}{2\alpha}x^2.$$

On round t, the player chooses $x \in \mathbb{R}$.

The loss is as follows (adversary):

1. if $|x| < 1$, then $f_t = (1+\alpha)f_{e_1} - \alpha f_{e_2}$,

Proof of Theorem 2

Proof.

Part 1/2. We will show that for any deterministic algorithm, there exists a sequence of loss functions with regret bounded by $\mathcal{O}(T)$. Assume $\lambda_e > -\alpha \, \forall e$. Define

$$f_{e_1}(x) = x^2, \qquad f_{e_2} = x^4 + \frac{1}{2\alpha}x^2.$$

On round t, the player chooses $x \in \mathbb{R}$.

The loss is as follows (adversary):

i. if $|x| < 1$, then $f_t = (1+\alpha)f_{e_1} - \alpha f_{e_2}$,

ii. if $|x| \geq 1$, then $f_t = f_{e_1}$.

Proof of Theorem 2

Proof.

Part 1/2. We will show that for any deterministic algorithm, there exists a sequence of loss functions with regret bounded by $\mathcal{O}(T)$. Assume $\lambda_e > -\alpha \, \forall e$. Define

$$f_{e_1}(x) = x^2, \qquad f_{e_2} = x^4 + \frac{1}{2\alpha}x^2.$$

On round t, the player chooses $x \in \mathbb{R}$.

The loss is as follows (adversary):

i. if $|x| < 1$, then $f_t = (1+\alpha)f_{e_1} - \alpha f_{e_2}$,

ii. if $|x| \geq 1$, then $f_t = f_{e_1}$.

Observe that in case i. the loss $f_t(x) \geq 0$, and in case ii. $f_t(x) \geq 1$.

Proof of Theorem 2

Proof.

Part 2/2. Suppose the player is in case i a times and case ii b times and suffers regret $> b$.

Proof of Theorem 2

Proof.

Part 2/2. Suppose the player is in case i a times and case ii b times and suffers regret $> b$.

Now consider the best actions in hindsight. If $a \leq \frac{T}{2}$, then $x^* = 0$ suffers 0 loss and the player's regret is at least $b = T - a \geq \frac{T}{2}$.

Proof of Theorem 2

Proof.

Part 2/2. Suppose the player is in case i a times and case ii b times and suffers regret $> b$.

Now consider the best actions in hindsight. If $a \leq \frac{T}{2}$, then $x^* = 0$ suffers 0 loss and the player's regret is at least $b = T - a \geq \frac{T}{2}$.

On the other hand, if $a > \frac{T}{2}$, then for any x, the player suffers a loss of

$$-a\alpha x^4 + (a/2 + a\alpha + b)x^2 \leq a\alpha(x^2 - x^4) + (a + b)x^2 = (a\alpha(1 - x^2) + T)x^2. \quad (1)$$

Proof of Theorem 2

Proof.

Part 2/2. Suppose the player is in case i a times and case ii b times and suffers regret $> b$.

Now consider the best actions in hindsight. If $a \leq \frac{T}{2}$, then $x^* = 0$ suffers 0 loss and the player's regret is at least $b = T - a \geq \frac{T}{2}$.

On the other hand, if $a > \frac{T}{2}$, then for any x, the player suffers a loss of

$$-a\alpha x^4 + (a/2 + a\alpha + b)x^2 \leq a\alpha(x^2 - x^4) + (a + b)x^2 = (a\alpha(1 - x^2) + T)x^2. \quad (1)$$

We can chose $x^* = \sqrt{1 + \frac{3}{\alpha}}$ and the player suffers regret $\geq \frac{T}{2}$. In either case, the player suffers $\mathcal{O}(T)$ regret. □

Deterministic vs. Randomized Algorithms

What about randomized algorithms?

Theorem 3

Theorem

Even with a randomized algorithm against an oblivious adversary playing bounded affine combinations, achieving sublinear regret is NP-hard.

Theorem 3

Theorem

Even with a randomized algorithm against an oblivious adversary playing bounded affine combinations, achieving sublinear regret is NP-hard.

Takeaway: Again, we cannot seem to achieve a sublinear regret rate for extrapolation.

When we consider the definitions in this paper, we realize there is a exponentially large gap, in the sense of computational complexity, between interpolation and extrapolation.

Theorem 4

Theorem

Against an oblivious adversary playing bounded affine combinations, the achievable regret is lower bounded as $\Omega(\sqrt{T})$ – with oracle access to approximate minimzation of a non-convex function.

Theorem 4

Theorem

Against an oblivious adversary playing bounded affine combinations, the achievable regret is lower bounded as $\Omega(\sqrt{T})$ – with oracle access to approximate minimzation of a non-convex function.

Takeaway: Recall that FTPL has a minimax lower bound of $\Omega(\sqrt{T})$ for an oblivious adversary playing arbitrary Lipschitz functions - FTPL *is just a noise variant of ERM*.

It turns out that that $\Omega(\sqrt{T})$ remains minimax for bounded affine combinations, that is, we cannot outperform ERM.

Outline

1 Introduction

2 Preliminaries

3 Online Domain Generalization Game

4 Interpolation Formal Results

5 Extrapolation Formal Results

6 Conclusion and Discussion

Takeaway

Divide domain generalization into two tasks i) interpolation and ii) extrapolation:

1. the computation complexity difference between the two is exponential, but
2. the statistical complexity (minimax) $\underbrace{\Theta(\log T), \Theta(\sqrt{T})}_{\text{unclear in what sense}}$ respectively, is not too different.

Discussion

1. Clarification of similarity of bounds.
2. In practice, the performance of ERM does not seem to match these results
3. Violated assumptions in practice
   - Linearity
4. Degree of extrapolation ($-\alpha$)
5. Definition of interpolation/extrapolation can (perhaps should) be reconsidered
6. Worst-case average loss is not the same as worst-case loss – the latter is still important

Questions?

Thanks for you attention!

Appendix I

Algorithm 1 Convex OFTPL

Input: Perturbation Distribution P_{PRTB}, number of samples m, number of iterations T
Denote $\nabla_0 = 0$
for $t = 1 \dots T$ **do**
    Let g_t be the guess for ∇_t
    for $j = 1 \dots m$ **do**
        Sample $\sigma_{t,j} \sim P_{\text{PRTB}}$
        $\mathbf{x}_{t,j} \in \operatorname{argmin}_{\mathbf{x} \in \mathcal{X}} \langle \nabla_{0:t-1} + g_t - \sigma_{t,j}, \mathbf{x} \rangle$
    end for
    Let $\mathbf{x}_t = \frac{1}{m} \sum_{j=1}^{m} \mathbf{x}_{t,j}$
    Play $\mathbf{x}_t$ and observe loss function f_t
end for

Appendix II

Algorithm 2 Nonconvex OFTPL

Input: Perturbation Distribution P_{PRTB}, number of samples m, number of iterations T
Denote $f_0 = 0$
for $t = 1 \dots T$ **do**
    Let g_t be the guess for f_t
    for $j = 1 \dots m$ **do**
        Sample $\sigma_{t,j} \sim P_{\text{PRTB}}$
        $\mathbf{x}_{t,j} \in \operatorname{argmin}_{\mathbf{x} \in \mathcal{X}} f_{0:t-1}(\mathbf{x}) + g_t(\mathbf{x}) - \sigma_{t,j}(\mathbf{x})$
    end for
    Let P_t be the empirical distribution over $\{\mathbf{x}_{t,1}, \mathbf{x}_{t,2} \dots \mathbf{x}_{t,m}\}$
    Play $\mathbf{x}_t$, a random sample generated from P_t
    Observe loss function f_t
end for