The Deep Bootstrap

Rethinking Generalization to Understand Deep Learning

Preetum Nakkiran  
Harvard

Behnam Neyshabur  
Google

Hanie Sedghi  
Google Brain

“Optimization is all you need!”

Motivation

**Goal:** “Understand” why DL methods used in practice work (small test error / test loss).

**Hope:** Predict how design choices affect test error.

**This Work:** *Framework/roadmap* for achieving goal (for supervised classification)
Setting (briefly)

**Setup:** Supervised classification.

Distribution \((x, y) \sim D\)

Want: classifier \(f(x)\) with small *test error* \(\Pr_{x,y \sim D} [f(x) \neq y]\)

Do: SGD on NN to minimize *train error*
Our Framework (high-level)

**Classical Framework:** Finite train set.

“Good models are those with small generalization gap”

**Our Framework:** Models trained on finite train set $\approx$ infinite train set

“Good models are those which optimize quickly, on infinite data”
Our Framework

**Main Idea:** compare Real World vs. Ideal World

Fix distribution $D$, architecture $\mathcal{F}$, num samples $n$.

Then, for all steps $t \in \mathbb{N}$ define:

- **Real World**($n$, $t$)
  
  SGD on **empirical loss**
  
  (Train Error $\leq$ Test Error)

  
  $f_t \leftarrow \text{Train}_{\mathcal{F},D}(n, t)$

- **Ideal World**($t$)

  SGD on **population loss**

  (Train Error $\equiv$ Test Error)

  $f_t^{\text{iid}} \leftarrow \text{Train}_{\mathcal{F},D}(\infty, t)$

Approximate Test Error

\[ \approx \text{Test Error} \]
Example

Real World: 50K samples, 100 epochs.  
Ideal World: 5M samples, 1 epoch.

Models which optimize faster in Ideal World, generalize better in Real World.
Figure 8: The corresponding train soft-errors for Figure 1.
(More) Precise Claim

**SGD on deep nets produces similar models whether trained on** re-used samples (Real) or fresh samples (Ideal)

...as measured by Test SoftError
...for as long as the Real World optimizer is still moving (e.g. TrainError ≥ 1%)
(More) Precise Claim

**ERM decomposition:**  
\[ \text{TestError}(f_t) = \text{TrainError}(f_t) + \left[ \text{TestError}(f_t) - \text{TrainError}(f_t) \right] \]  
\text{Generalization gap}

**Our decomposition:**  
\[ \text{TestError}(f_t) = \underbrace{\text{TestError}(f_{t}^{iid})}_{A: \text{Online Learning}} + \left[ \text{TestError}(f_t) - \text{TestError}(f_{t}^{iid}) \right] \]  
\[ \underbrace{\varepsilon(n, \mathcal{D}, \mathcal{F}, t)}_{B: \text{Bootstrap error}} \]

**Main Claim:** Bootstrap error \( \varepsilon(n, \mathcal{D}, \mathcal{F}, t) \) is small for realistic \((n, \mathcal{D}, \mathcal{F})\), and all \( t \leq T(n) \)

Where “stopping time” \( T(n) := \text{time when Real World reaches TrainError \leq 1\%} \).
Real World vs. Ideal World: Varying Train Size

Test Soft-Error vs. SGD Iterations for different train sizes:
- Real (n=1000)
- Real (n=2000)
- Real (n=5000)
- Real (n=10000)
- Real (n=25000)
- Real (n=50000)

Comparison with Ideal World line.
$L(n)$: Test error on $n$ samples (Real World, trained to convergence)
$T(n)$: Time to converge on $n$ samples (Real World SGD steps)
$\tilde{L}(t)$: Test error after $t$ online SGD steps (Ideal World)

Deep Bootstrap: \[ L(n) \approx \tilde{L}(T(n)) \] 

NB: Scaling exponents multiply

Thus, good training procedures:

1. **Optimize quickly** on infinite samples [$\tilde{L}$ small]
   (high-capacity models, skip-connections, BN, ...)

2. **Don’t optimize too** quickly on finite samples [$T$ large]
   (regularization, data-aug, ...)
To understand generalization, sufficient to understand:

1. **Online optimization:** how fast Ideal World learns.
   [long history, but not in DL]

2. **Empirical optimization:** how fast Real World converges.
   [recent progress: Arora, Allen-Zhu,...]

3. **Bootstrap Error:** $|\text{Real} - \text{Ideal}|$
   [long history in stats, but not in DL]

Assume/prove/believe bootstrap error small ⇒ generalization reduced to **optimization!**
Validation: Summary of Experiments

- **CIFAR-5m**: 5-million synthetic samples from a generative model trained on CIFAR-10.


- **Varying settings**: \{archs, opt, LR,...\} convnets, ResNets, MLPs, Image-GPT, Vision-Transformer.

![Samples from CIFAR-5m]

![Real vs. Ideal Worlds]

Figure 2: **Real vs Ideal World**: CIFAR-5m. SGD w 0.1 (●), 0.01 (■), 0.001 (▲). (b): Random architecture.
Implications:
Deep Learning through the Bootstrap Lens
Effect of Pretraining

Pretrained models generalize better (Real) “because” they optimize faster (Ideal)

Figure 13: Real vs. Ideal Worlds for Vision Transformer on CIFAR-5m, with and w/o pretraining.
Effect of Data Aug

Data-aug in the Ideal World = Augment each sample once

Two potential effects:
1. Ideal World Optimization Speed
2. Real World Convergence Speed

Good data-augs:
1. Don't hurt learning in Ideal World
2. Decelerate optimization in Real World (train for longer)

see “Affinity and Diversity” of [Gontijo-Lopes et al.]
Implicit Bias $\rightarrow$ Explicit Optimization

Two archs from [Neyshabur 2020]:
D-CONV (convnet) $\subset$ D-FC (mlp)

Both train to 0 Train Error, but convnet generalizes better.

Traditionally: due to “implicit bias” of SGD on the convnet.

Our view: due to better optimization in the Ideal World
Effect of Learning Rate
Random Labels (Thought Experiment)

“Understanding deep learning requires rethinking generalization” [Zhang et al. 2016]
- Train on randomly-labeled inputs.
- 0% train error, 90%/trivial test error.

Here:
- Real World: Test Error >> Train Error
- Real World Test ≈ Ideal World Test
Choice of Metric Matters!

Figure 6: **SoftError vs. Error vs. Loss: ResNet-18.**
Conclusions 1

- Reduced: one hard problem (generalization) \(\Rightarrow\) two hard problems (on/offline optimization)

- In future: Forget generalization. Focus on **optimization**.
  - Largest models trained for less than one epoch (= Ideal World)
  - Many mysteries of ML remain in Ideal World
    (no “generalization problem”, but: arch, repr. learning, robustness...)
  - Every new advance in DL: “How does it affect online opt? Offline opt?”
Conclusions 2

- Connection between over/under parameterized regimes:
  - “Overparam models behave like underparam ones...in certain sense (test soft-error)”
    “Deep Bootstrap” [N, Neyshabur, Sedghi 2020]
  - “Overparam models DO NOT behave like underparam ones in general”
    “Distributional Generalization” [Nakkiran, Bansal 2020]

- Many arbitrary choices in deep learning (arch, loss, optimizer, activation..)
  - Q: Which ones work for generalization?
  - A: Anything that works well for online optimization

Speculation: Holds much more generically (not just SGD/deep nets/etc..)
Extras
What about Non-Deep Learning?

• Not true for well-specified linear regression!
• Can be contrived to be true for misspecified regression

\[
x \sim \mathcal{N}(0, V)
\]
\[
y := \sigma(\langle \beta^*, x \rangle)
\]
\[
f_\beta(x) := \langle \beta, x \rangle
\]

Figure 7: Toy Example. Examples of settings with large and small bootstrap error.

• Setting A. Linear activation \(\sigma(x) = x\). With \(n = 20\) train samples.
• Setting B. Sign activation \(\sigma(x) = \text{sgn}(x)\). With \(n = 100\) train samples.
When Bootstrap Fails

1. Near Double-Descent region (Real World has pathology)
   - Or any setting with non-monotonic Soft-Error

2. Very small number of samples

3. Potentially: weird distributions / architectures / optimizers?
Why Soft-Error?

**Want:** RealWorld $\rightarrow$ IdealWorld as (model, data) $\rightarrow \infty$.
- This doesn’t always happen w.r.t Test Error.

**Claim:** In an overparameterized limit of (model, data) $\rightarrow \infty$,
interpolating classifiers converge to *optimal samplers*: $f(x) \sim p(y|x)$

“Distributional Generalization” [Nakkiran, Bansal 2020]

...**NOT** to Bayes-optimal classifiers: $f^*(x) = \arg\max_{y} p(y|x)$
Scaling Laws in Ideal World

$L(t) :$ Ideal-world learning curve

Empirically: power law

$L(t) \sim t^{-\alpha}$
ImageNet Experiments

(a) Standard architectures.  

(b) ResNet-18s of varying width.

Figure 3: **ImageNet-DogBird**. Real World models trained on 10K samples.
Effect of Pretraining

(b) Pretrain: Image-GPT ($n = 2K$).
When Data-Aug Hurts

Figure 10: Effect of Data Augmentation in the Ideal World.
Figure 17: CIFAR-5m Samples. Random samples from each class (by row).

Figure 18: CIFAR-10 Samples. Random samples from each class (by row).
<table>
<thead>
<tr>
<th>Trained On</th>
<th>Test Error On</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CIFAR-10</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>0.032</td>
</tr>
<tr>
<td>CIFAR-5m</td>
<td><strong>0.088</strong></td>
</tr>
</tbody>
</table>

Table 2: WRN28-10 + cutout on CIFAR-10/5m
CIFAR-5m Experiments

Figure 2: **Real vs Ideal World: CIFAR-5m.** SGD with 50K samples. (a): Varying learning-rates 0.1 (●), 0.01 (■), 0.001 (▲). (b): Random architectures from DARTS space (Liu et al., 2019).
ImageNet Experiments

Real vs. Ideal Worlds

Real vs. Ideal Worlds (no data aug)
Validation: Summary of Experiments

• **CIFAR-5m**: 5-million synthetic samples from a generative model trained on CIFAR-10
  • Realistic: Training WRN on n=50K from CIFAR-5m yields 91.2% test acc on CIFAR-10

• **ImageNet-DogBird**: 155K images by collapsing ImageNet categories.
  • Real World: n=10K for 120 epochs
  • Ideal World: n=155K for < 8 epochs (approximation of $n = \infty$)

• **Various archs**: convnets, ResNets, MLPs, Image-GPT, Vision-Transformer
RealWorld($N, T = \infty$) $\approx$ RealWorld($N, T_N$) $\approx \epsilon$ RealWorld($\infty, T_N$)

"Deep Bootstrap"

Practice: Real World
(trained as long as possible)

Real World
(stopped at $T_N$: when $\text{Train Error} \approx 1\%$)

Ideal World
(stopped at $T_N$)
Learning curves:

$L(n)$: Test error on $n$ samples (Real-world, trained to convergence)
$T(n)$: Time to converge on $n$ samples (Real world SGD steps)
$\tilde{L}(t)$: Test error after $t$ online SGD steps (Ideal World)

Then:

$L(n) \approx \tilde{L}(T(n))$
Classical Framework (ERM)

**Classical Framework**: Finite data, need to understand *generalization gap*

\[
\text{TestError}(f_t) = \text{TrainError}(f_t) + [\text{TestError}(f_t) - \text{TrainError}(f_t)]
\]

“*Good models are those with small generalization gap*”

**Obstacles:**
2. Large models can fit train sets → trivializes framework