“Optimization is all you need!”

The Deep Bootstrap Framework

Rethinking Generalization to Understand Deep Learning

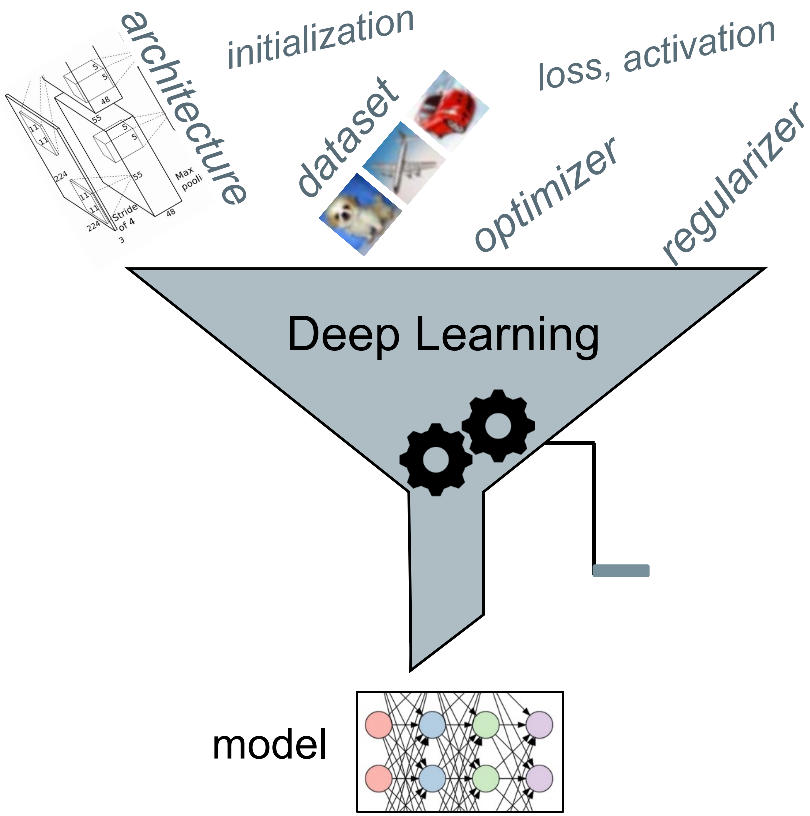
Preetum Nakkiran
Harvard

Behnam Neyshabur
Google

Hanie Sedghi
Google Brain

Appears in ICLR 2021

Motivation

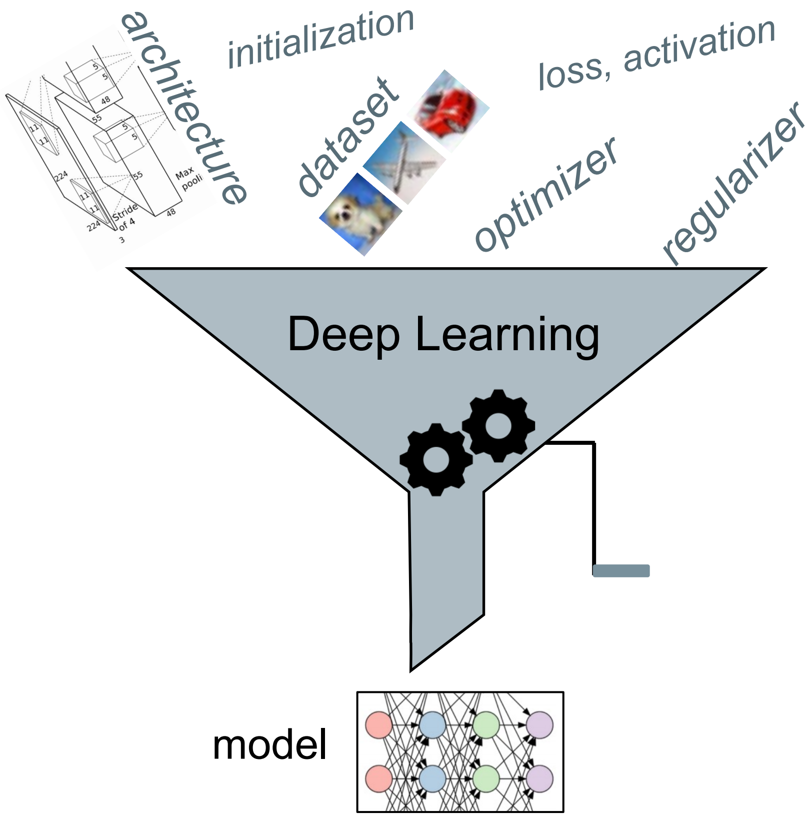


<u>Deep Learning:</u>
accepts inputs (design choices),
produces output (model)

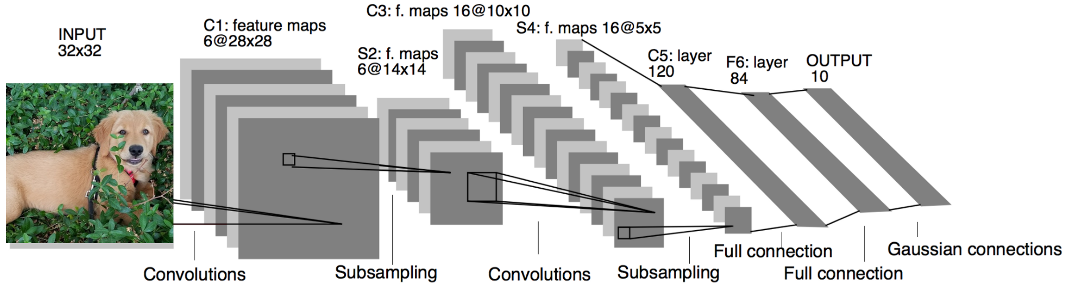
" How does what we ***do*** affect what we ***get?*** "

- Advances in DL are unpredictable.
- Every advance = new choice of inputs
- Surprised by which choices work!

Motivation

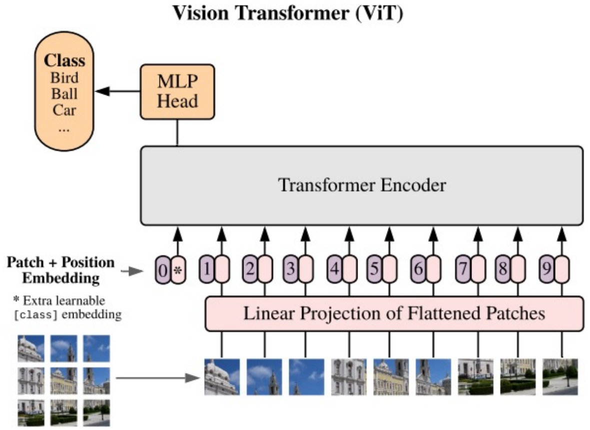


~1998-2020: ConvNets dominate vision

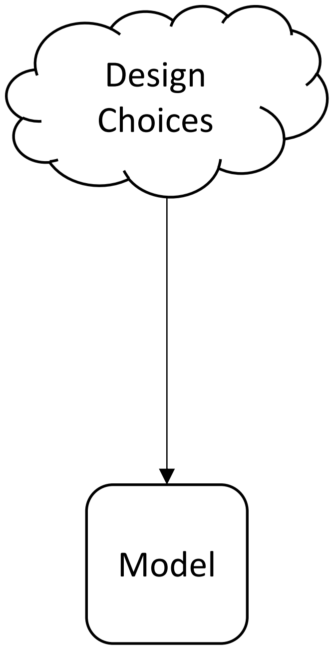


[LeCun et al 1998]

2020: *Transformers* (from NLP) dominate vision



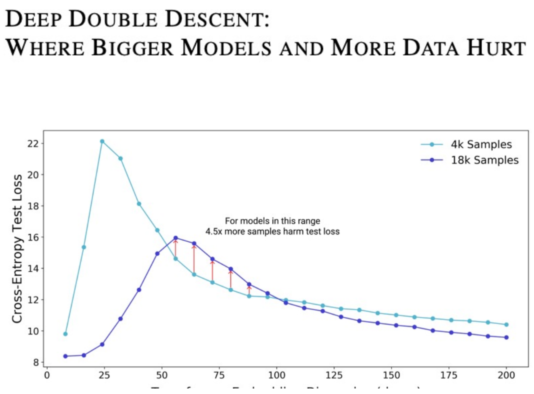
[Dosovitskiy et al 2020]

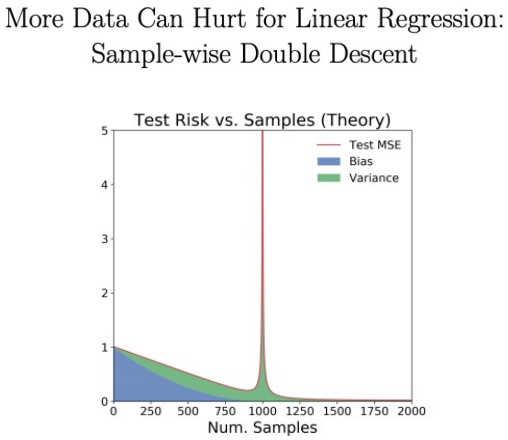


Deep Learning
= Map from {design choices} → model

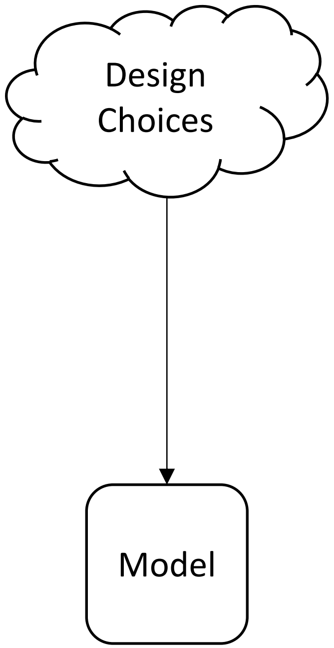
"Understanding Deep Learning"
= Identifying **structure** of map

Ex of structure: **monotonicity**
Q: "does training on more data always improve test perf?"





Supervised classification



Setup:

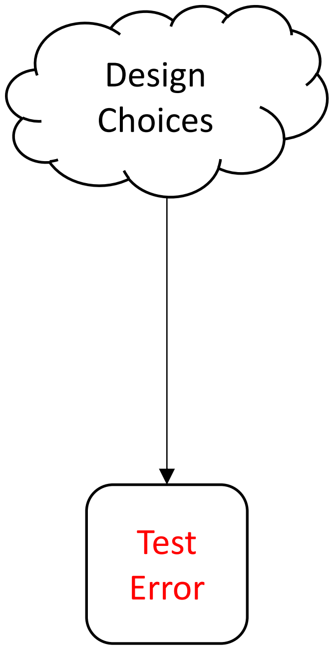
Distribution $(x, y) \sim D$

Given: iid samples from D

Do: SGD* on Neural Net to minimize *train error*

Measure: *test error* $\Pr_{x,y\sim D}[f(x) \neq y]$

Supervised classification



Setup:

Distribution $(x, y) \sim D$

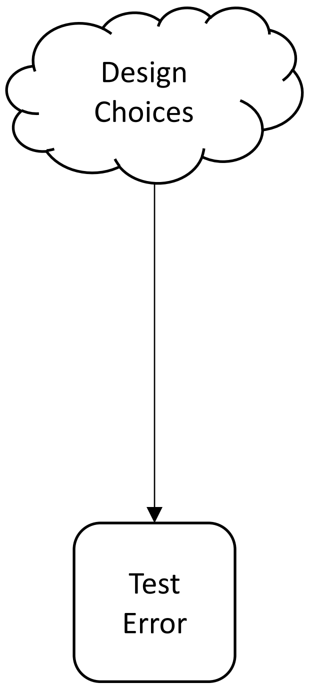
Given: iid samples from D

Do: SGD* on Neural Net to minimize *train error*

Measure: *test error* $\Pr_{x,y \sim D}[f(x) \neq y]$

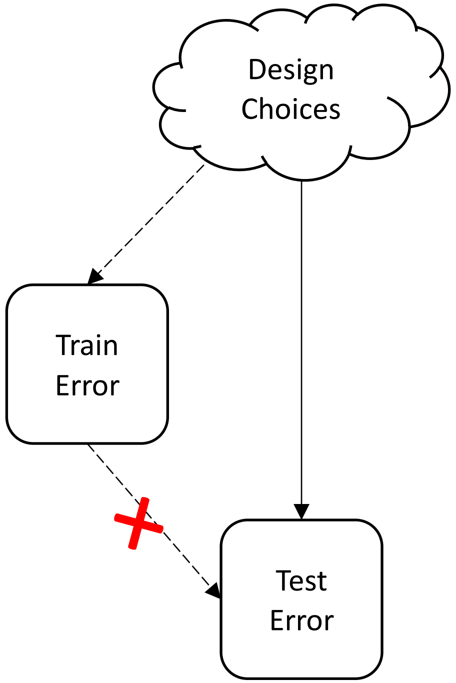
Generalization Frameworks

= **Factorization** of map



Generalization Frameworks

"Uniform Convergence Framework"



Any "big enough" network can have Train Error ≈ 0

[Zhang et al. 2016]

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)

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)
- Sample train set $S \sim D^n$
- Initialize architecture f_0 from $\mathcal{F}$
- For t steps:
  - Sample minibatch from S
  - Gradient step on minibatch
- Output f_t

Ideal World(t)

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)

- Sample train set $S \sim D^n$
- Initialize architecture f_0 from $\mathcal{F}$
- For t steps:
    - Sample minibatch from S
    - Gradient step on minibatch
- Output f_t

Ideal World(t)

- Initialize architecture f_0 from $\mathcal{F}$
- For t steps:
    - Sample minibatch from D
    - Gradient step on minibatch
- Output f_t^{iid}

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:

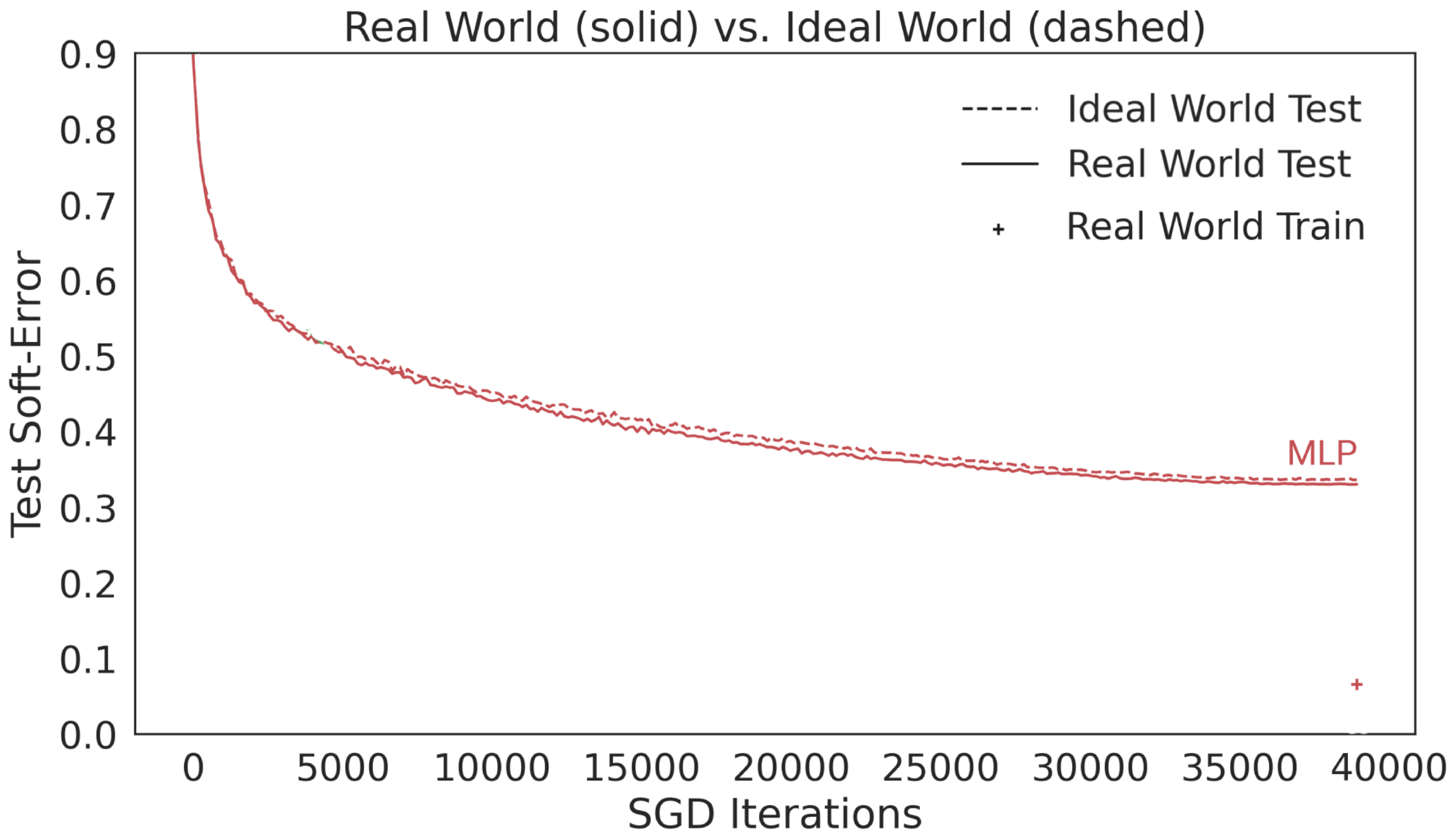
Real World(n, t)

SGD on empirical loss
(Train Error $\leq$ Test Error)

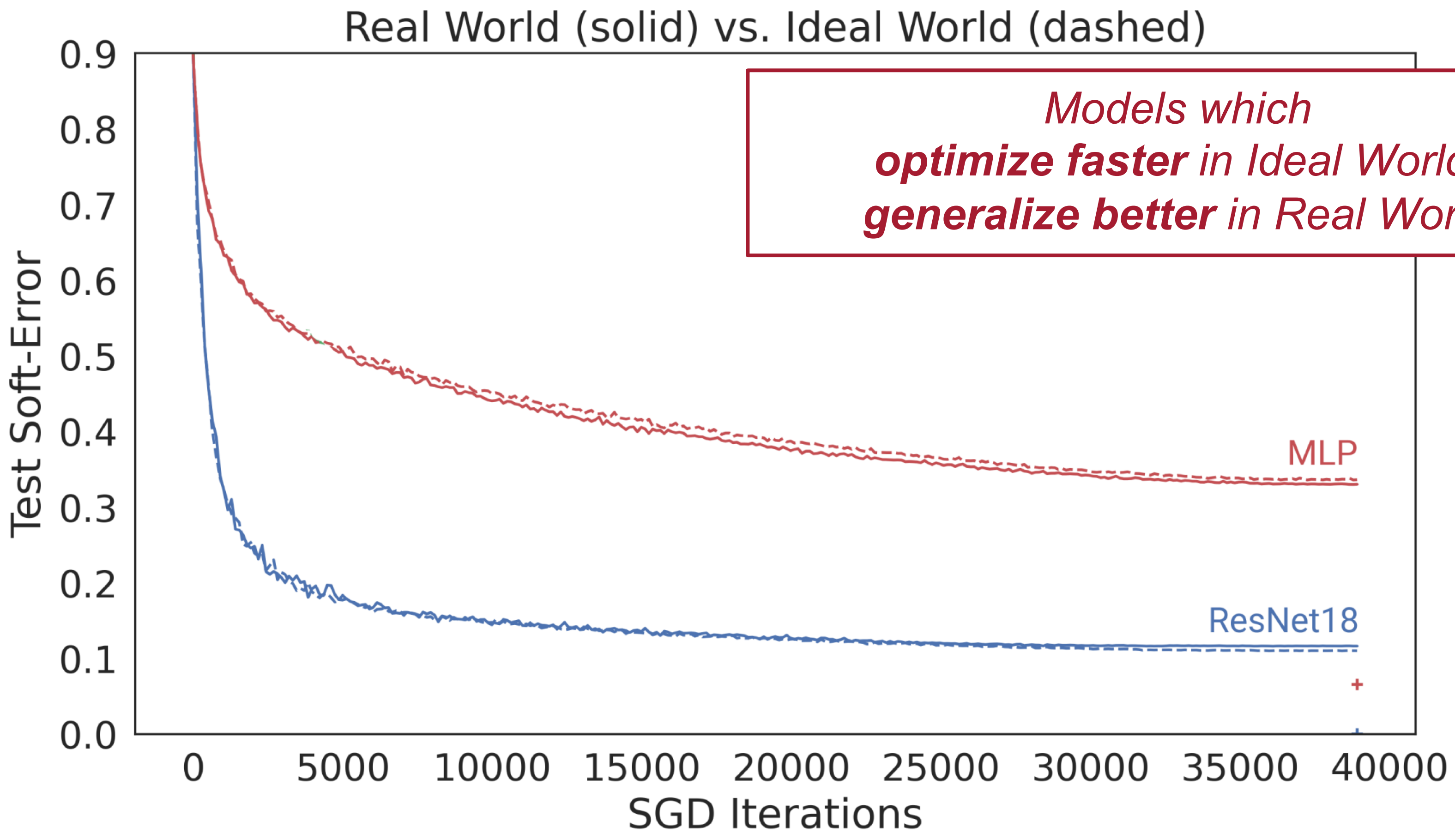
Test Error
$\approx$

Ideal World(t)

SGD on population loss
(Test Error)



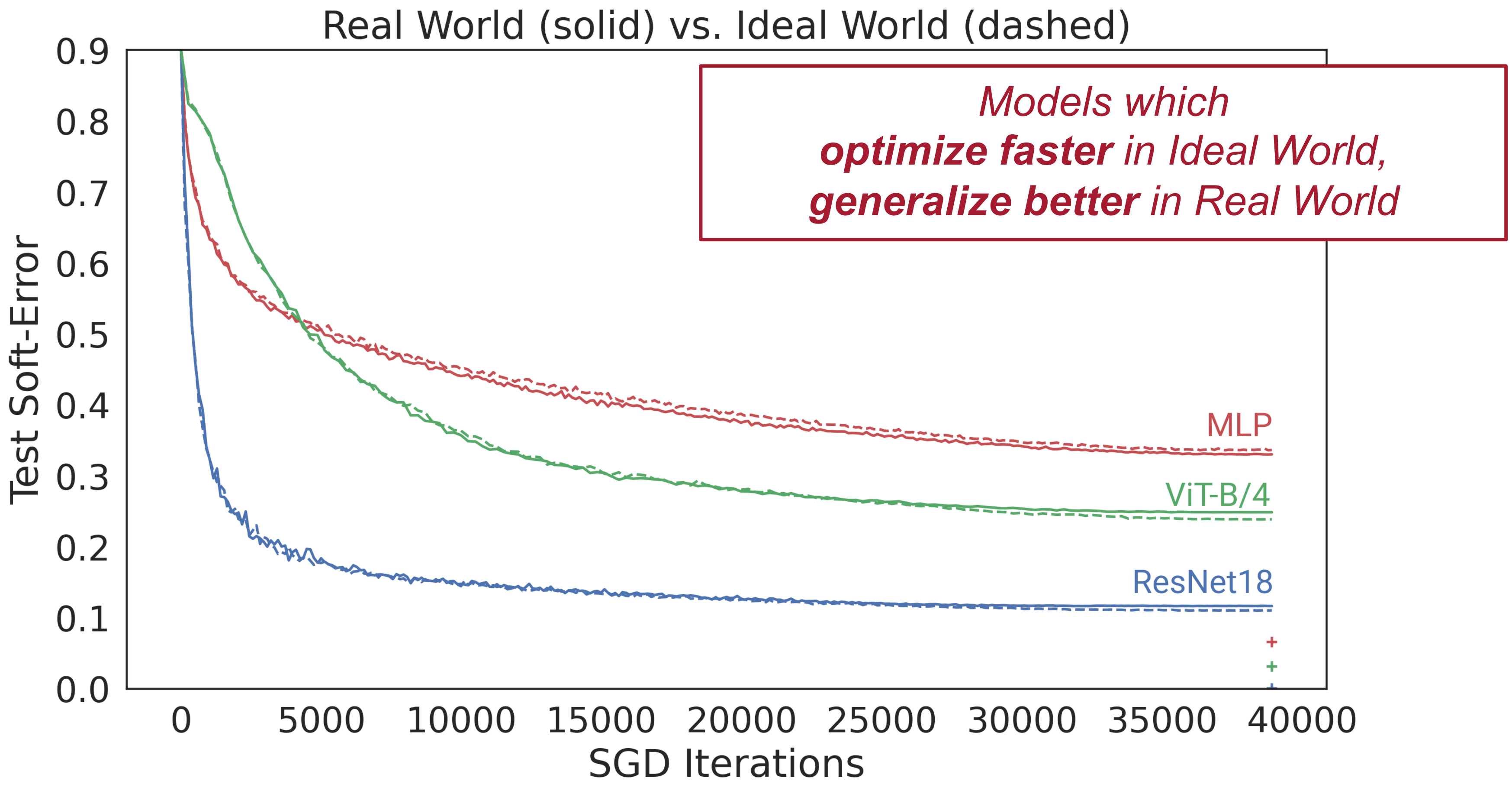
Real World: 50K samples, 100 epochs. **Ideal World:** 5M samples, 1 epoch.



Models which
***optimize faster** in Ideal World,*
***generalize better** in Real World*

Real World: 50K samples, 100 epochs.

Ideal World: 5M samples, 1 epoch.



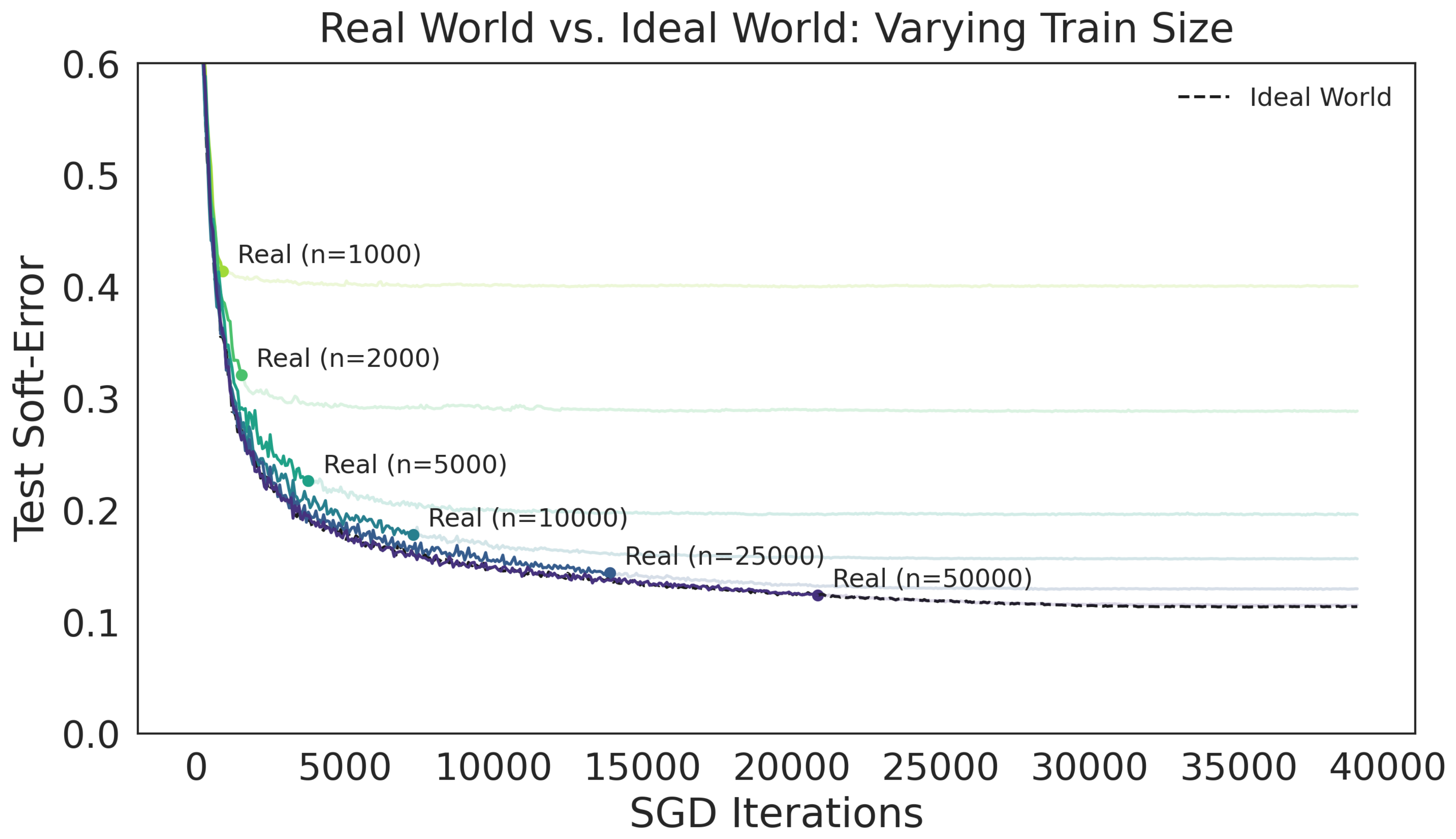
Real World: 50K samples, 100 epochs. **Ideal World:** 5M samples, 1 epoch.

$T(n)$: "Stopping time". Real World time to converge on **n** samples (< 1% train error)

Deep Bootstrap:

$$\forall t \leq T(n): \qquad \text{RealWorld}(n, t) \approx_\epsilon \text{IdealWorld}(t)$$

"SGD on deep nets behaves similarly
*whether trained on **re-used samples** or **fresh samples***
...up until the Real World has converged"


Real World vs. Ideal World: Varying Train Size
Ideal World
Real (n=1000)
Real (n=2000)
Real (n=5000)
Real (n=10000)
Real (n=25000)
Real (n=50000)
Test Soft-Error
0.0
0.1
0.2
0.3
0.4
0.5
0.6
SGD Iterations
0
5000
10000
15000
20000
25000
30000
35000
40000

Deep Bootstrap:

$$\mathrm{FinalError}(n) \approx_{\epsilon} \mathrm{IdealWorld}\big(T(n)\big)$$

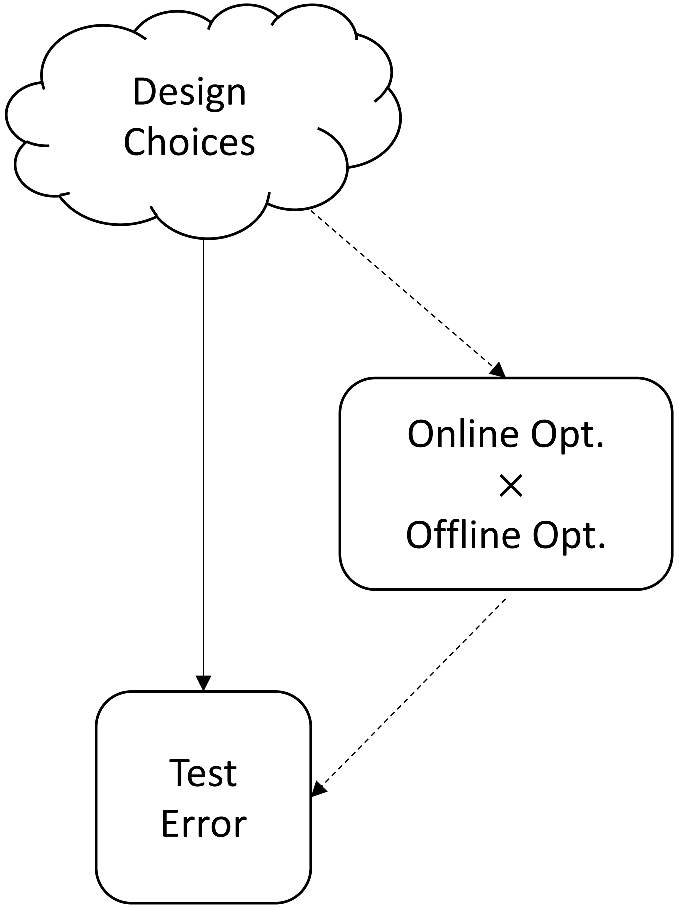
$T(n)$: Time to converge on **n** samples

LHS: Generalization

RHS: Optimization
(Online optimization & Empirical Optimization)

Deep Bootstrap:

$$\mathrm{FinalError}(n) \approx_{\epsilon} \mathrm{IdealWorld}\big(T(n)\big)$$

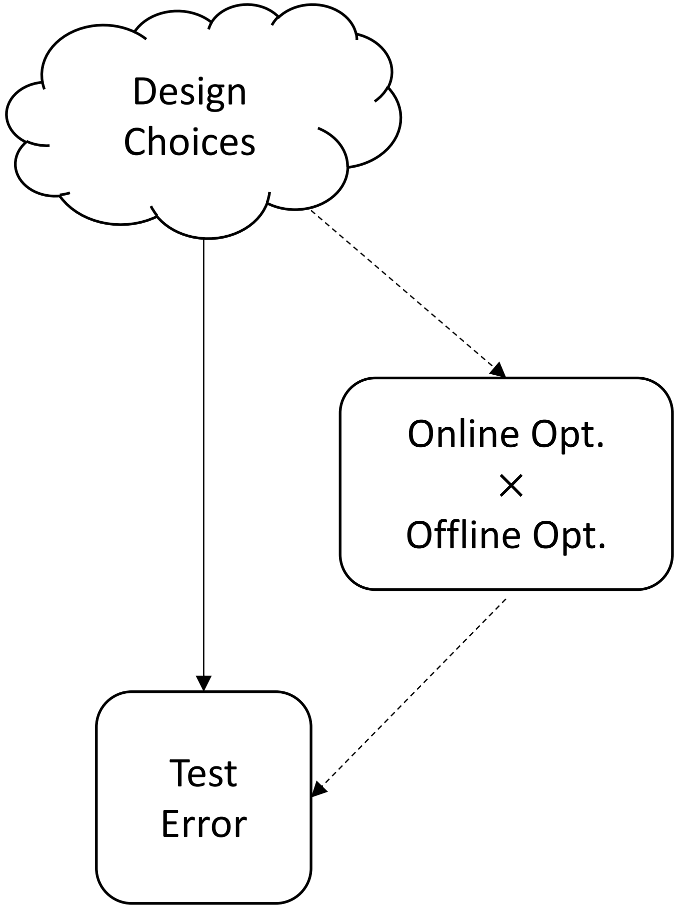


Empirically verified for varying:

- Architectures
- Model size
- Data size
- Optimizers (SGD/Adam/etc)
- Pretraining
- Data-augmentation
- Learning rate
- …

Deep Bootstrap:

$$\mathrm{FinalError}(n) \approx_{\epsilon} \mathrm{IdealWorld}\big(T(n)\big)$$



Good design choices:

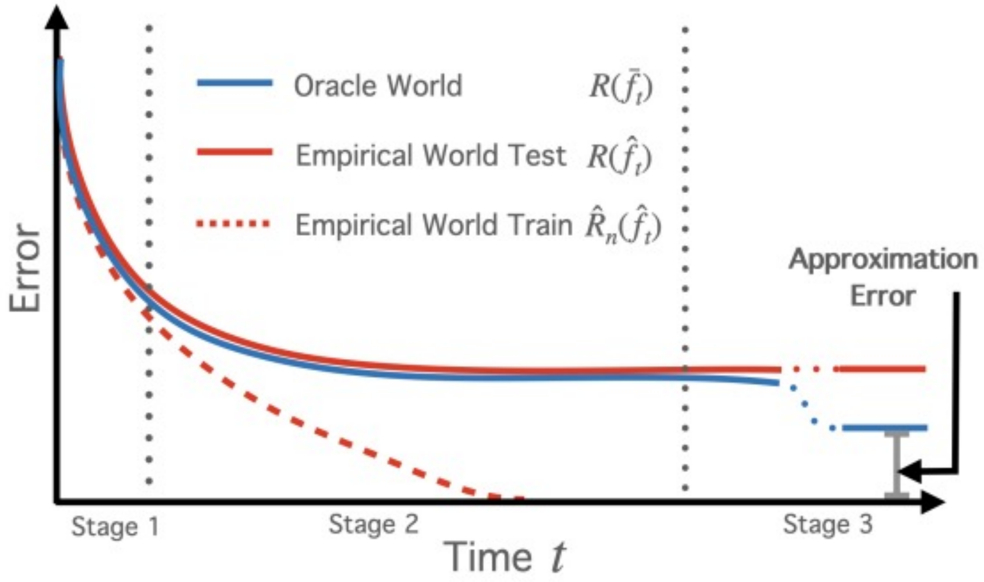
1. **Optimize quickly** in online setting
(large models, skip-connections, pretraining,…)

2. **Don't optimize too** quickly on finite samples
(regularization, data-aug,…)

Recent proof in Kernel setting:

The Three Stages of Learning Dynamics in High-dimensional Kernel Methods

Nikhil Ghosh[1], Song Mei[1], and Bin Yu[1,2,3,4]



ERM decomposition:

$$\mathrm{TestError}(f_t) = \mathrm{TrainError}(f_t) + \underbrace{[\mathrm{TestError}(f_t) - \mathrm{TrainError}(f_t)]}_{\text{Generalization gap}}$$

Our decomposition:

$$\mathrm{TestError}(f_t) = \underbrace{\mathrm{TestError}(f_t^{\mathrm{iid}})}_{\text{A: Online Learning}} + \underbrace{[\mathrm{TestError}(f_t) - \mathrm{TestError}(f_t^{\mathrm{iid}})]}_{\text{B: Bootstrap error}}$$

$$\varepsilon(n, \mathcal{D}, \mathcal{F}, t)$$

<u>**Main Claim**</u>: *Bootstrap error $\epsilon(n, \mathcal{D}, \mathcal{F}, t)$ is small for realistic $(n, \mathcal{D}, \mathcal{F})$, and all $t \leq T(n)$*

Where "stopping time" $T(n)$:= time when Real World reaches TrainError $\leq 1\%$.

$L(n)$: Test error on **n** samples (Real World, trained to convergence)

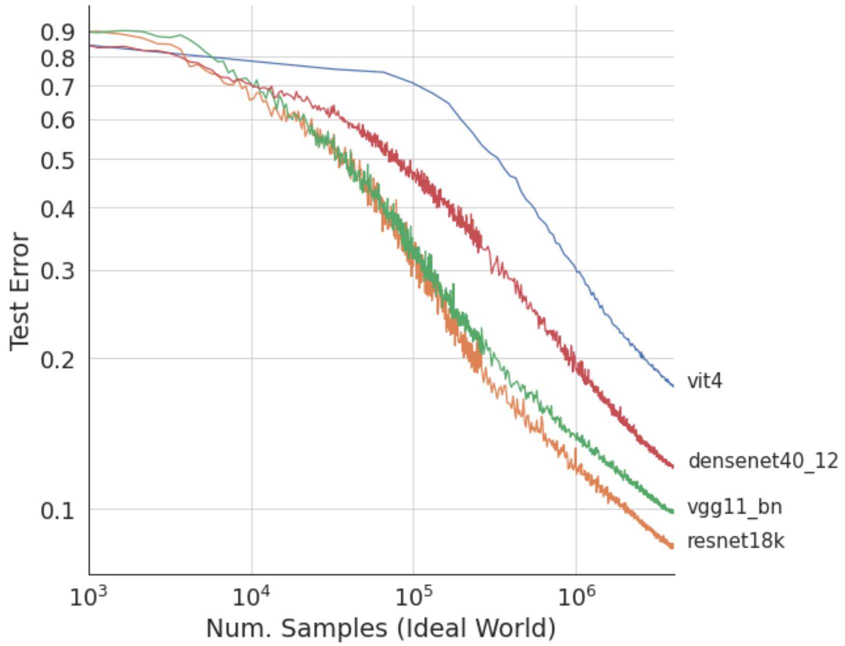
$T(n)$: Time to converge on **n** samples (Real World SGD steps)

$\tilde{L}(t)$: Test error after **t** online SGD steps (Ideal World)

<u>Deep Bootstrap:</u> $L(n) \approx \tilde{L}(T(n))$

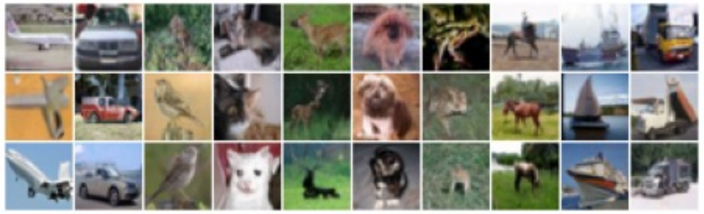
NB: Scaling exponents multiply

Assuming $T(n) \sim \Theta(n)$,
(Learning curve exponent) ≈ (Online optimization exponent)



Validation: Summary of Experiments

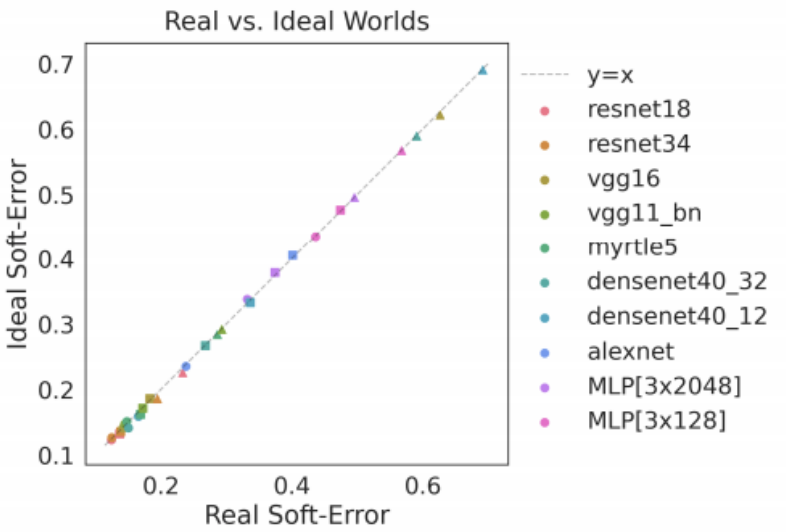
- **CIFAR-5m:** 5-million synthetic samples from a generative model trained on CIFAR-10



Samples from CIFAR-5m

- **ImageNet-DogBird:** 155K images by collapsing ImageNet catagories. Binary task.

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



(a) Standard architectures.

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

Alternate Perspectives

Generalization Perspective:		Optimization Perspective:
“ConvNets *generalize better* than MLPs”	→	“ConvNets *optimize faster* than MLPs”
“Pretraining helps *generalization*”	→	“Pretraining helps *optimization*” (a la *preconditioning*)

Effect of Pretraining

Pretrained models generalize better (Real)
and optimize correspondingly faster (Ideal)

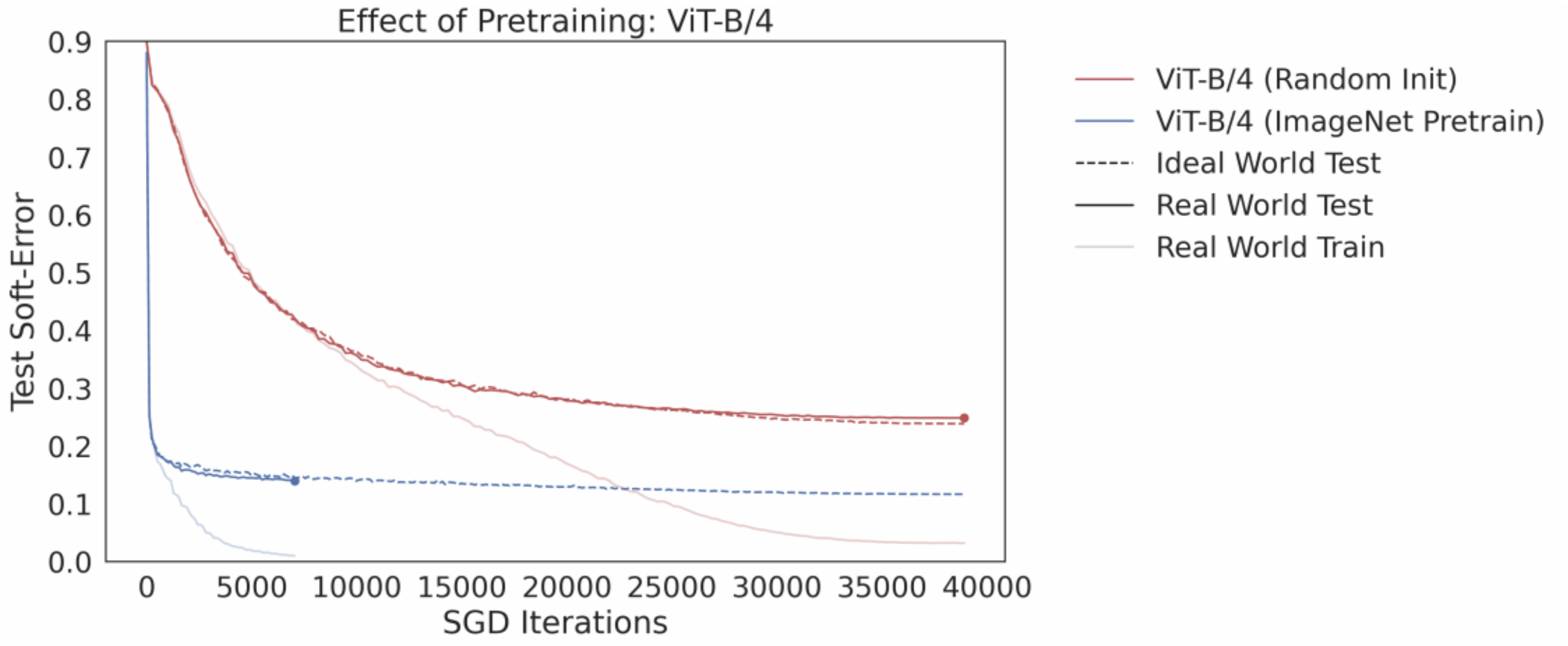


Figure 13: Real vs. Ideal Worlds for Vision Transformer on CIFAR-5m, with and w/o pretraining.

Effect of Data Augmentation

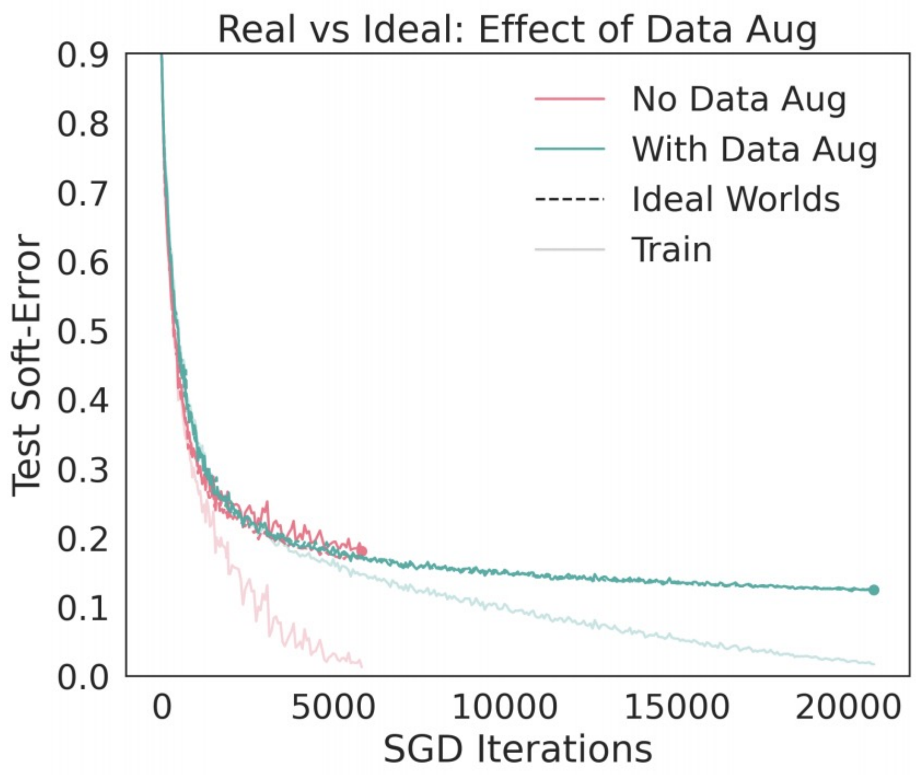
Data-aug in the Ideal World =
Augment each sample once

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.]

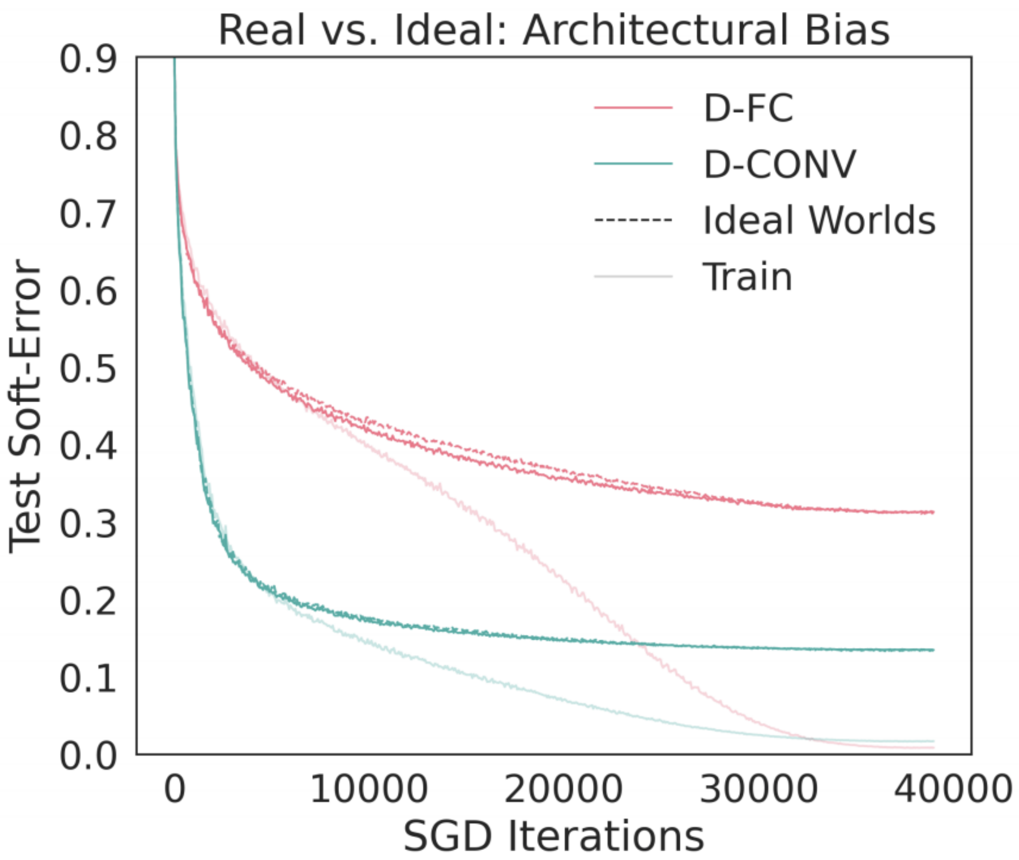
Implicit Bias → Explicit Optimization

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

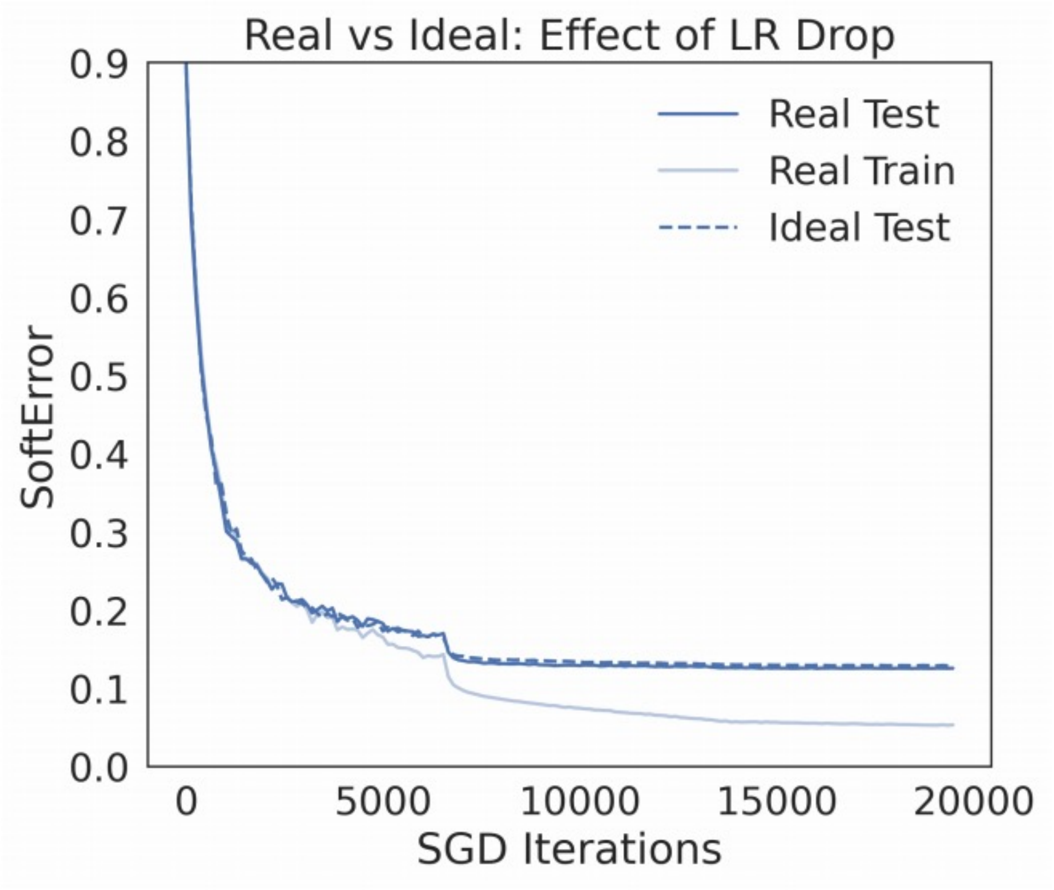
Both train to 0 Train Error, but convnet generalizes better.

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, rethinking:

- Real World: Test Error >> Train Error
- Real World Test ≈ Ideal World Test

Insights on a Practical Mystery

Two regimes in practice:

1. Effectively infinite data (e.g. train on internet, 1B+ samples)
   want architectures which optimize quickly

2. Small finite data (e.g. 50K samples)
   want architectures which generalize well

<u>Mystery:</u> Why do we use the same architectures in both regimes?

<u>Deep Bootstrap:</u> Not a coincidence…

Choice of Metric Matters!

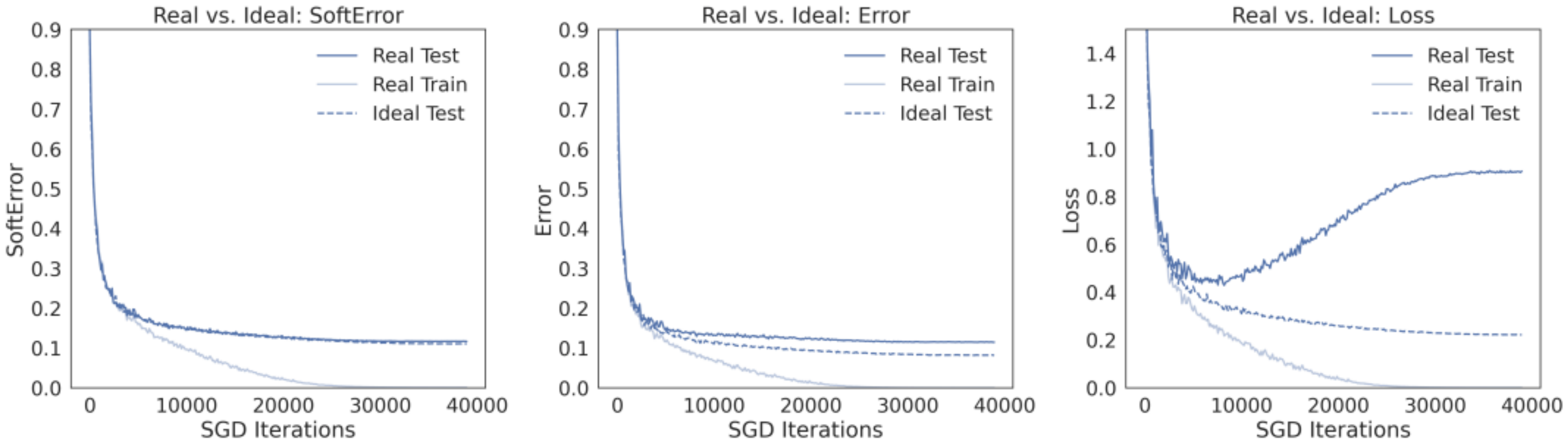


Figure 6: **SoftError vs. Error vs. Loss: ResNet-18.**

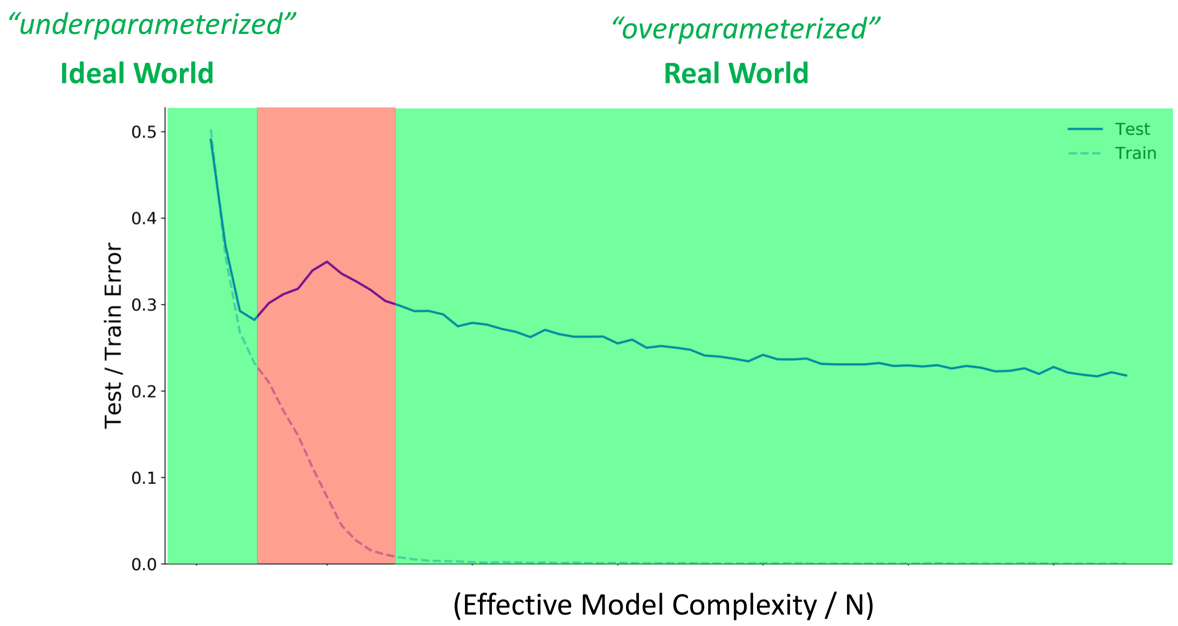
Conclusions

Assuming bootstrap claim: Reduces *generalization* to *optimization*.

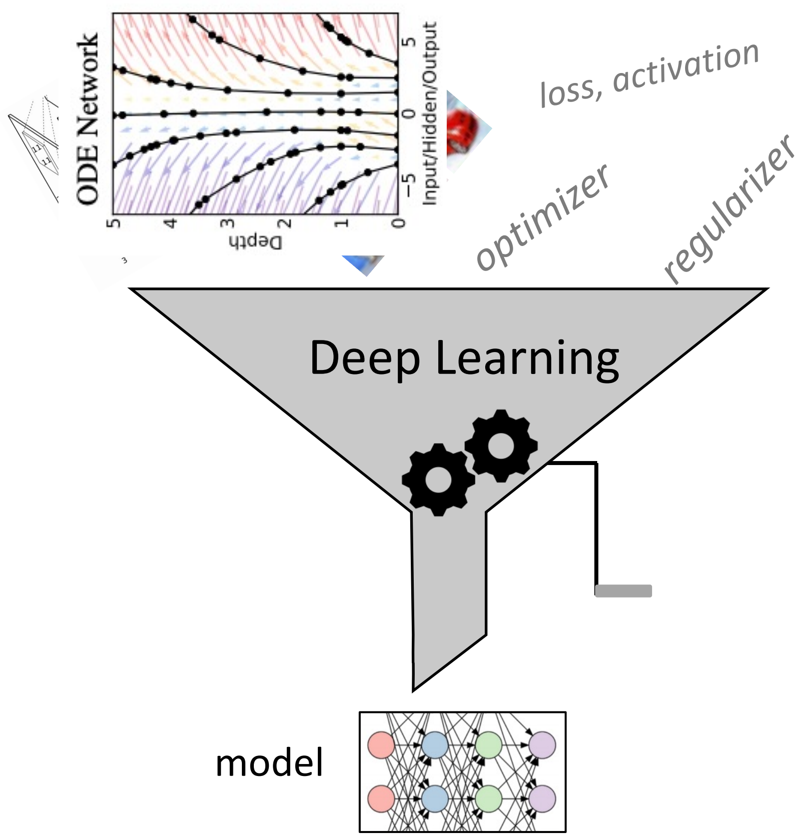
Hope: Refocus attention on online optimization aspects of deep learning
(some modern models actually in "Ideal World")

Connects *overparametrized* and *underparameterized* regimes:

Models which fit their train sets "behave like" models trained on infinite data



Conclusions



Many *diverse* choices in deep learning "work" (generalize)

Want theory of generalization that applies to all.

<u>Deep Bootstrap:</u>

"Any choice that works for online optimization will work for offline generalization."

Speculation: Holds much more generically than deep learning…

Thanks!

preetum@nakkiran.org
preetum.nakkiran.org

EXTRAS

Open Problems

Mysteries of Online Learning:

- (essentially all mysteries in ML remain)
- Why do certain architectures optimize faster on certain distributions?
- How to characterize interaction between: {architecture, optimizer, task}?
- Why does pretraining act as a preconditioner?
- Out-of-distribution robustness
- Why do we “learn representations”?
- …

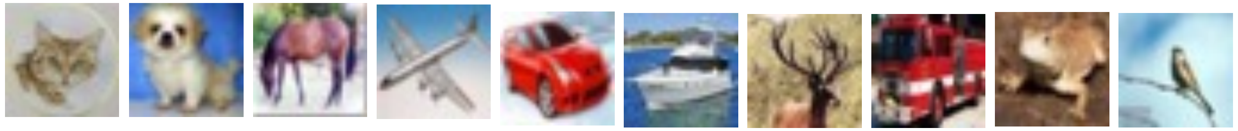
Beyond Test Error: Similarities between Real & Ideal World

- Similar behavior under distribution shift?
- Similar representations?
- Similar transfer performance

Ideal World as a Testbed:

- To compare optimizers
- Calibration/uncertainty/ensembling

Digression



Setup: Take CIFAR-10 train set, apply label noise: cats → dog w.p. 30%.
Train a ResNet to 0 train error. What happens on test samples?

Result: Cats → dog w.p. ~30% on test set! (other classes unaffected)

Surprising because:

- Not close to Bayes-optimal classifier
- Ideal World won't do this
    - (unless we consider randomized softmax instead of argmax)

"Distributional Generalization"
[Nakkiran, Bansal 2020]

When Bootstrap Fails

1. Near Double-Descent region (Real World has pathology)

   - Or any setting with non-monotonic Soft-Error

2. Potentially: weird distributions / architectures / optimizers?

   (seems to work in any setting with “real data”, regardless of model)

Why Soft-Error?

Want: F(RealWorld) – F(IdealWorld) $\rightarrow 0$ as (model, data) $\rightarrow \infty$.

This doesn't happen for F = TestError, if:

(1) Bayes risk $\neq 0$.

(2) Take overparameterized limit: (model, data) $\rightarrow \infty$, $\text{model} \gg \text{data}$

"Distributional Generalization"
[Nakkiran, Bansal 2020]

Why Soft-Error?

Want: F(RealWorld) – F(IdealWorld) $\to 0$ as (model, data) $\to \infty$.

This doesn't happen for F = TestError, if Bayes risk $\neq 0$.

Suppose Real World takes overparameterized limit: (model, data) $\to \infty$

Ideal World converges to *Bayes-optimal classifier:*

$$\lim_{T\to\infty,\ S\to\infty} \tilde{f}_{S,T}(x) = \operatorname{argmax}_y p(y \mid x)$$

Real World converges to *optimal sampler:*

$$\lim_{N\to\infty,\ S\to\infty} f_{S,N}(x) \sim p(y|x)$$

"Distributional Generalization"
[Nakkiran, Bansal 2020]

What about Non-Deep Learning?

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

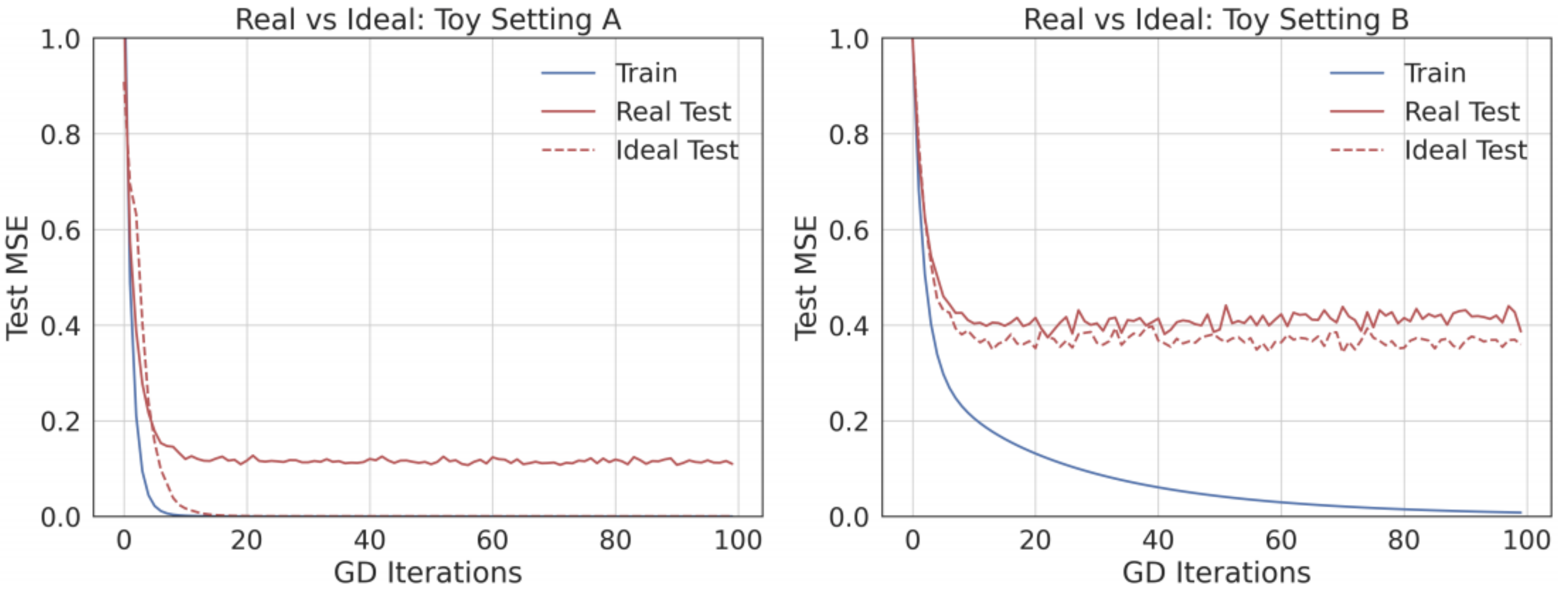


Figure 7: **Toy Example.** Examples of settings with large and small bootstrap error.

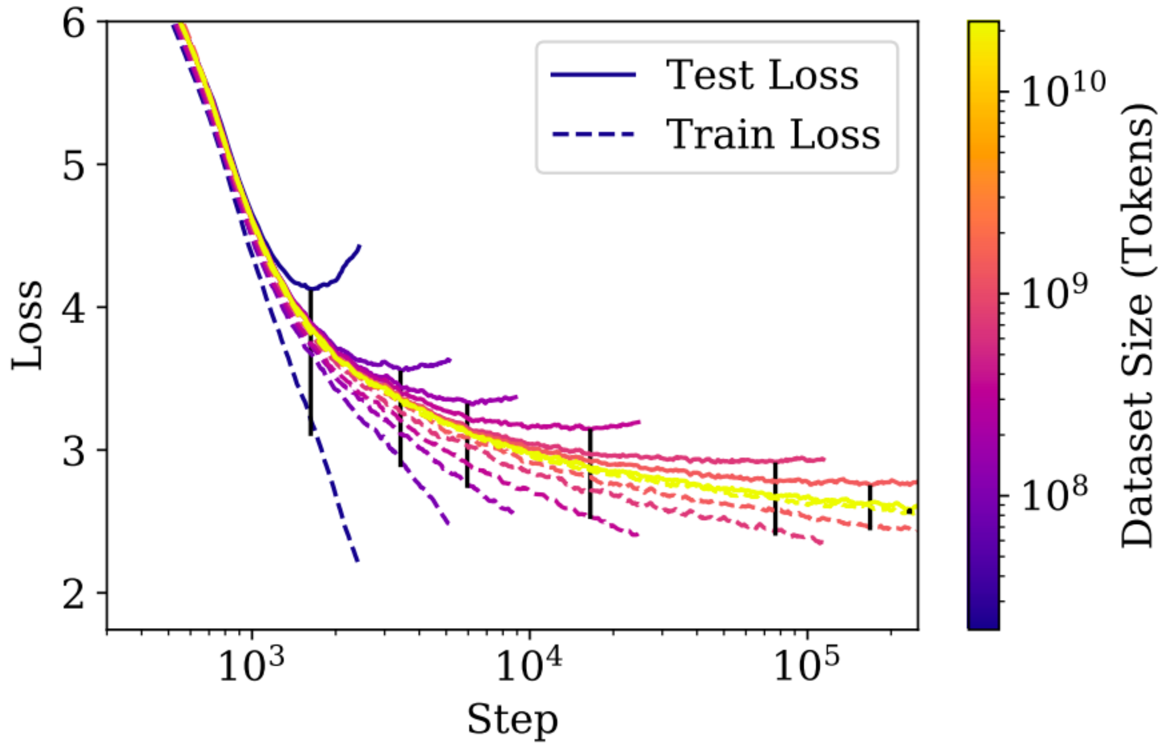
$$x \sim \mathcal{N}(0, V)$$
$$y := \sigma(\langle \beta^*, x \rangle)$$

$$f_\beta(x) := \langle \beta, x \rangle$$

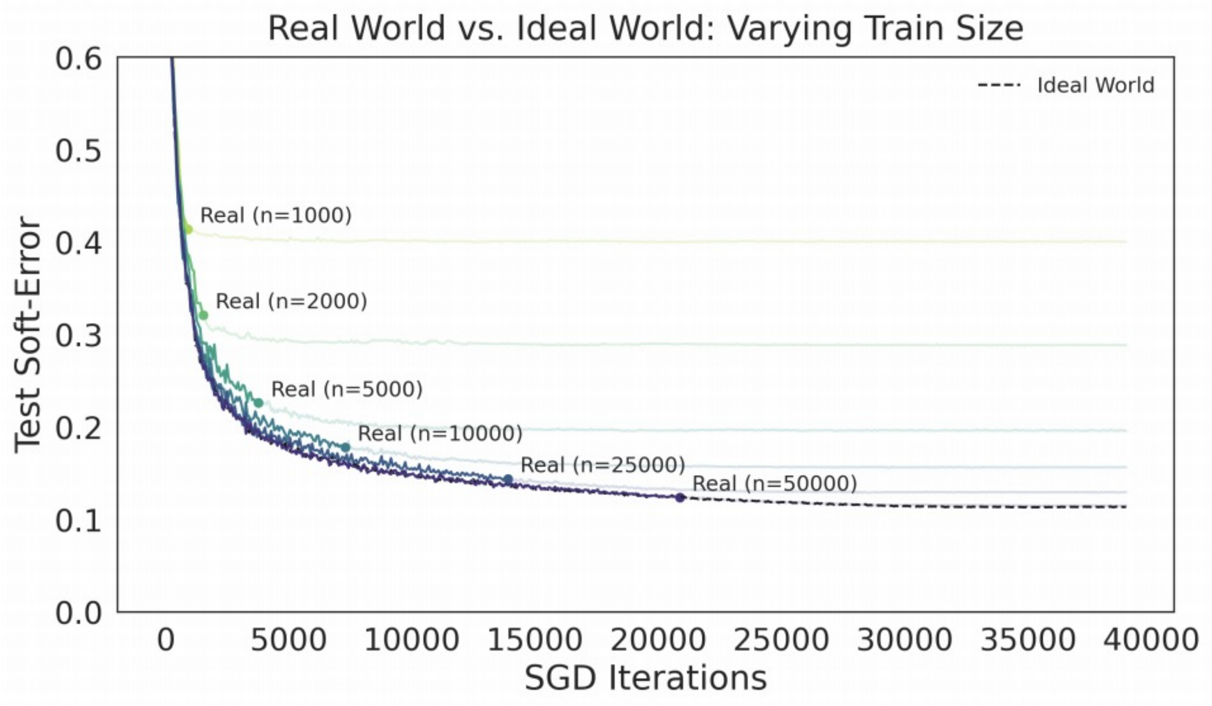
- **Setting A.** Linear activation $\sigma(x) = x$. With $n = 20$ train samples.
- **Setting B.** Sign activation $\sigma(x) = \text{sgn}(x)$. With $n = 100$ train samples.

GPT-3 Learning Curves

[Kaplan et al 2020]



ResNet18 Curves

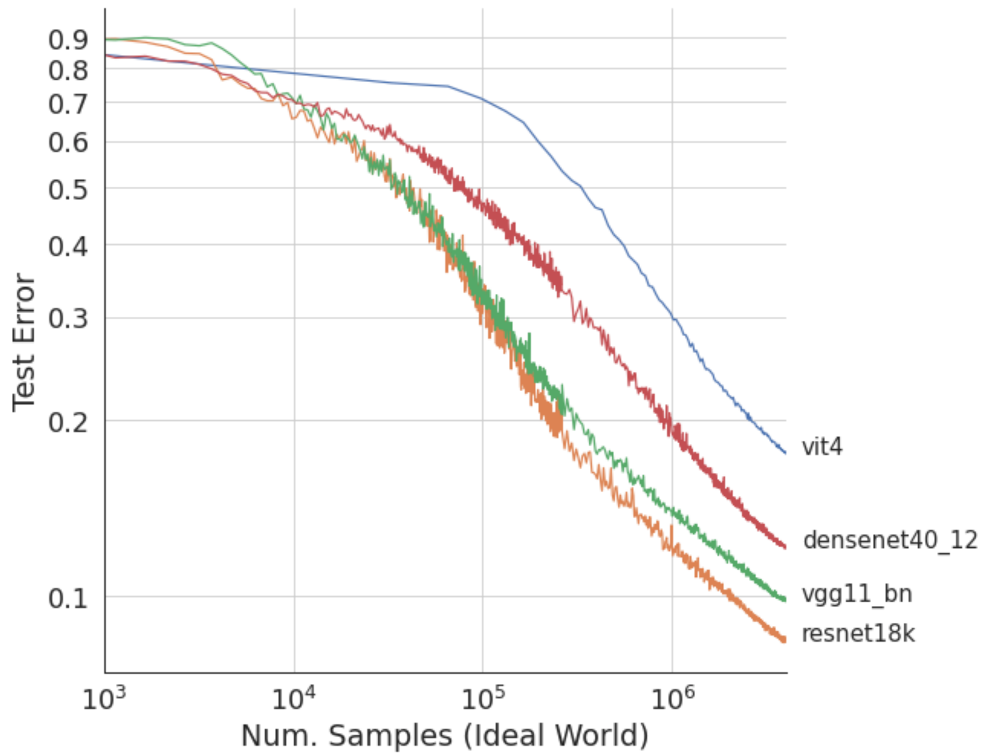


Scaling Laws in Ideal World

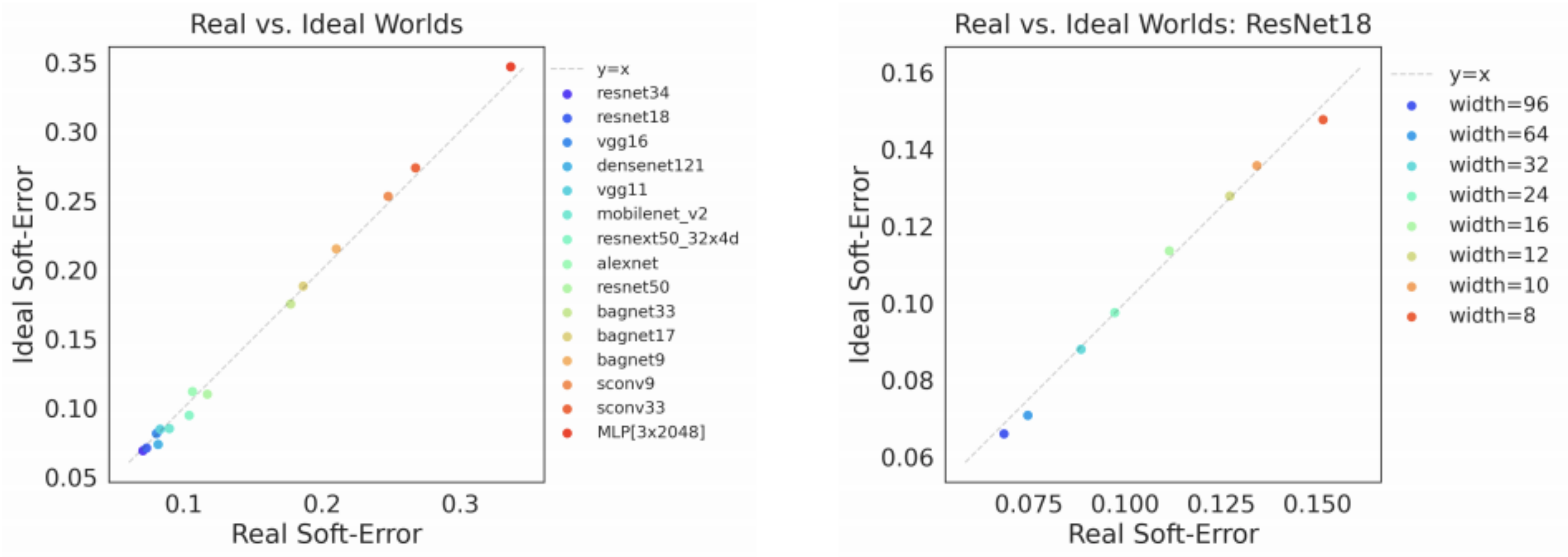
L(t) : Ideal-world learning curve

Empirically: power law

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



ImageNet Experiments

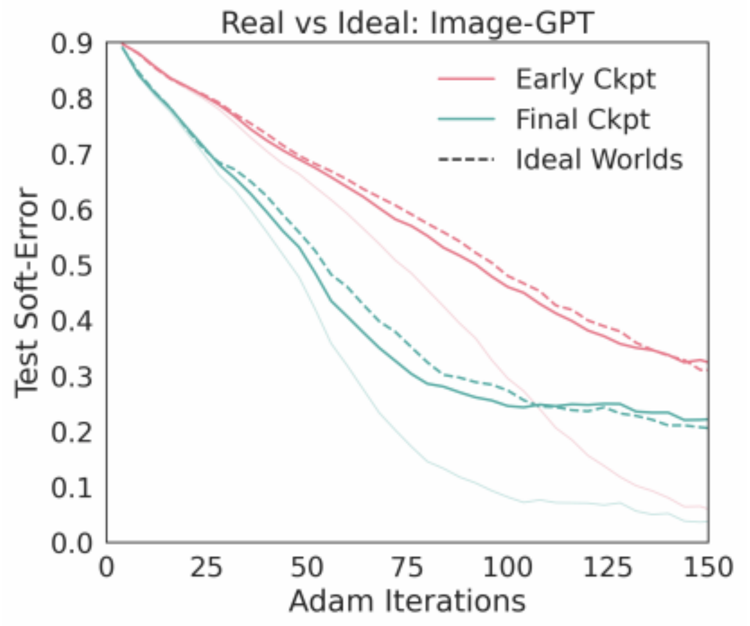


(a) Standard architectures.

(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

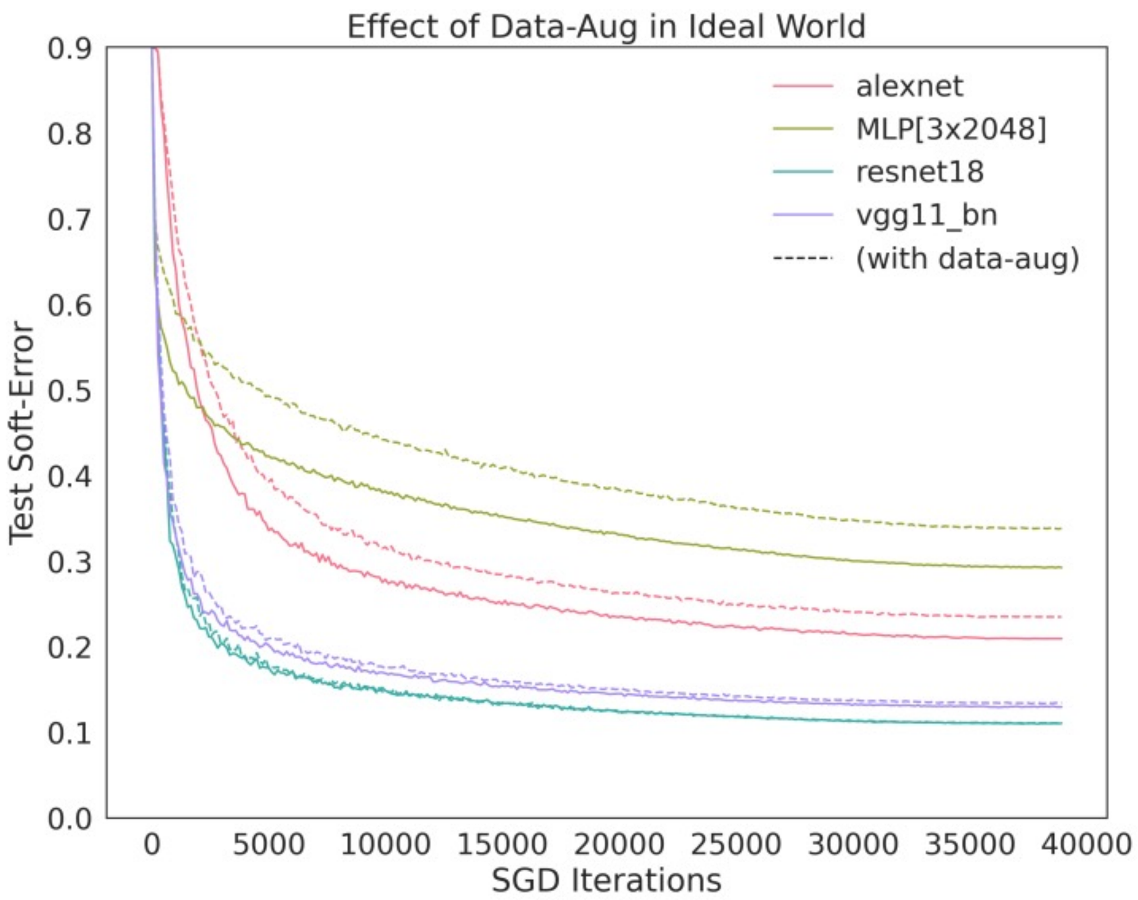


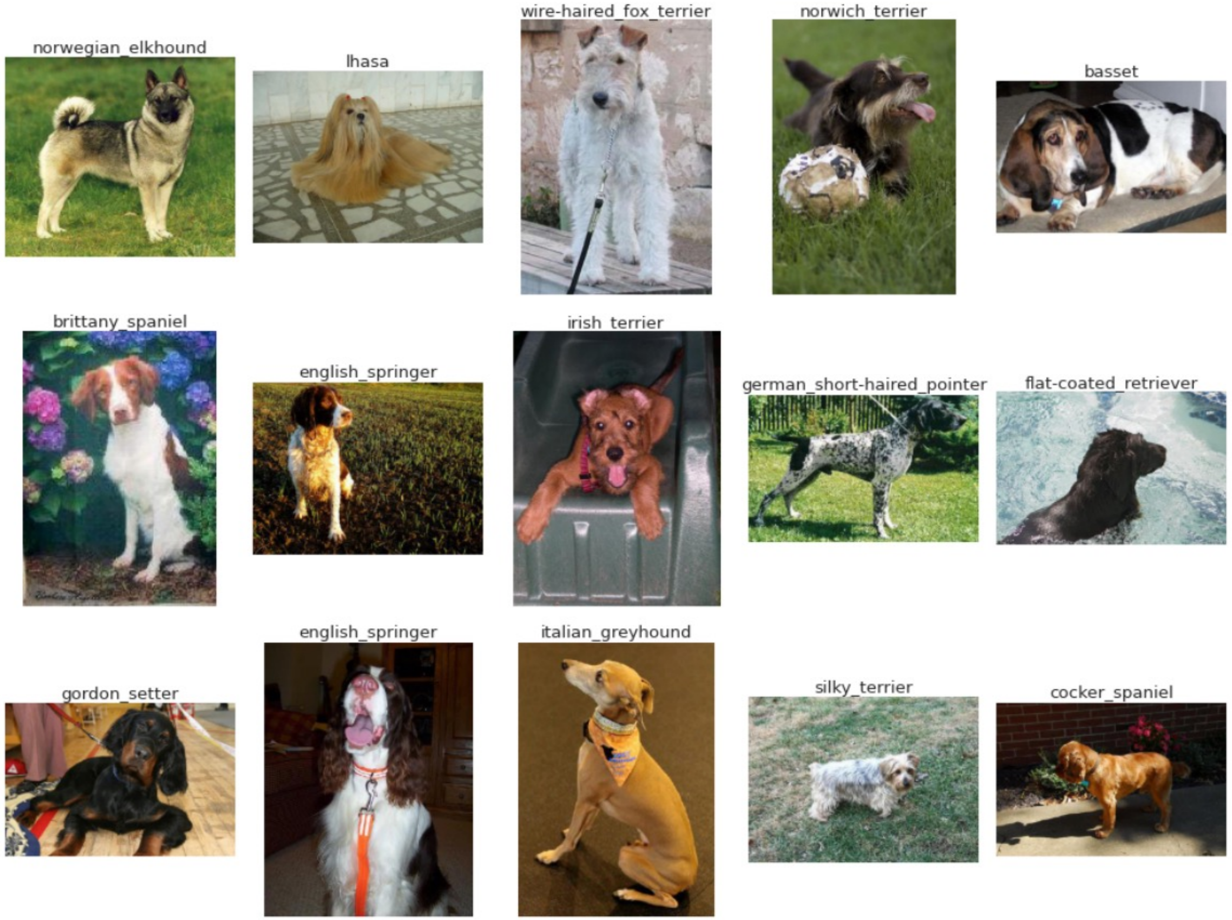
Figure 10: Effect of Data Augmentation in the Ideal World.

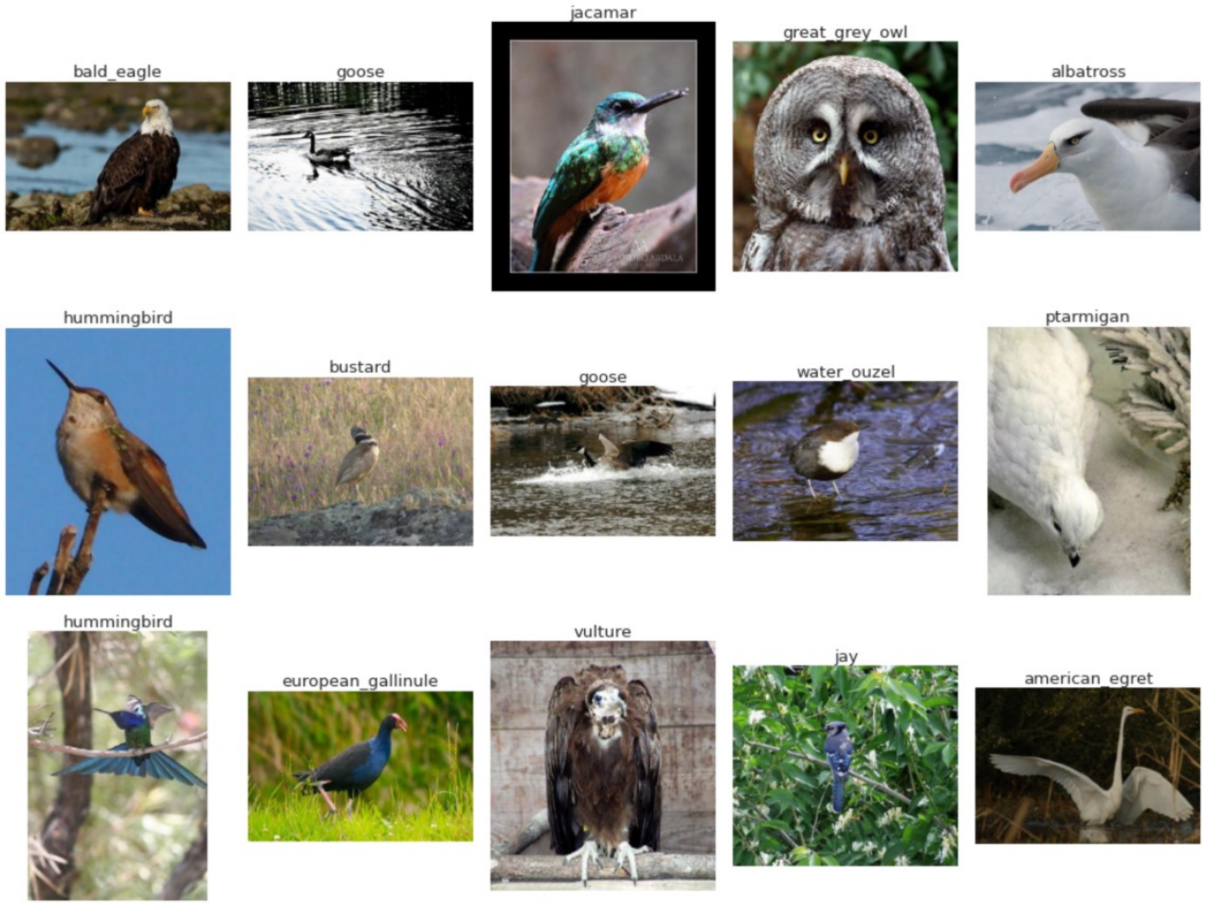
Figure 17: **CIFAR-5m Samples.** Random samples from each class (by row).

Figure 18: **CIFAR-10 Samples.** Random samples from each class (by row).

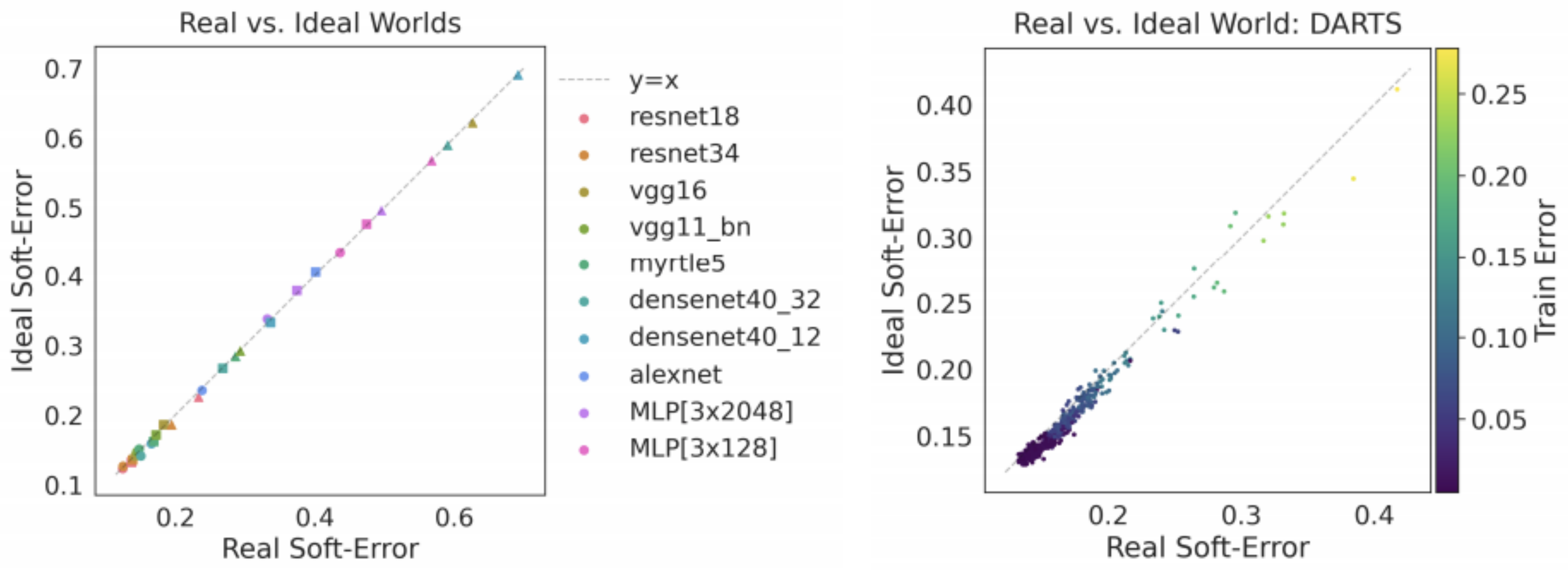
Trained On	Test Error On	
	CIFAR-10	CIFAR-5m
CIFAR-10	0.032	0.091
CIFAR-5m	$\boxed{0.088}$	0.097

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


norwegian_elkhound
lhasa
wire-haired_fox_terrier
norwich_terrier
basset
brittany_spaniel
english_springer
irish_terrier
german_short-haired_pointer
flat-coated_retriever
gordon_setter
english_springer
italian_greyhound
silky_terrier
cocker_spaniel


bald_eagle
goose
jacamar
great_grey_owl
albatross
hummingbird
bustard
goose
water_ouzel
ptarmigan
hummingbird
european_gallinule
vulture
jay
american_egret

CIFAR-5m Experiments



(a) Standard architectures.

(b) Random DARTS architectures.

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).

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 catagories.
  - 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

"Deep Bootstrap"

$$\mathrm{RealWorld}(N, T=\infty) \approx \mathrm{RealWorld}(N, T_N) \approx_\epsilon \mathrm{RealWorld}(\infty, T_N)$$

Practice: Real World
(trained as long as possible)

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

Ideal World
(stopped at T_N)

Classical Framework (ERM)

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

$$\underline{\text{TestError}(f_t)} = \underline{\cancelto{0}{\text{TrainError}(f_t)}} + \underbrace{[\text{TestError}(f_t) - \cancelto{0}{\text{TrainError}(f_t)}]}_{\text{Generalization gap}}$$

"Good models are those with small generalization gap"

Obstacles:

1. Hard: Decades of work, little progress.
2. Large models can fit train sets → trivializes framework