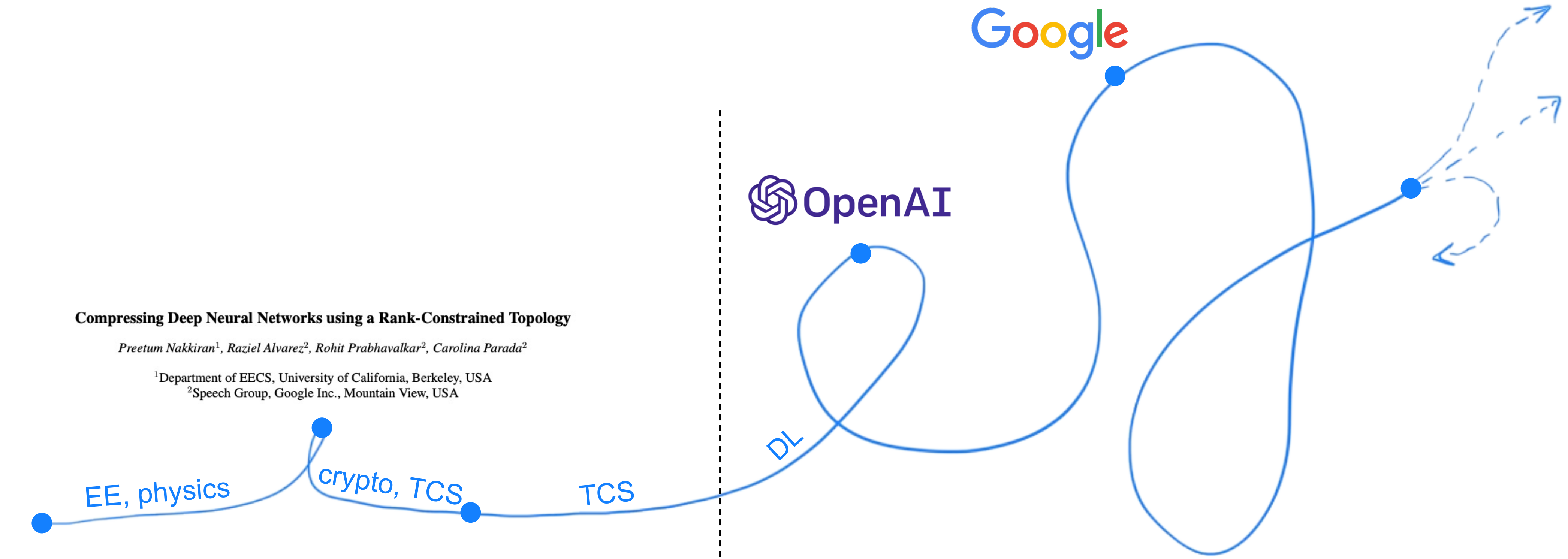
Towards Understanding Deep Learning

Preetum Nakkiran

UCSD

Mar 14, 2022 @ Apple

About Me



2012

Berkeley
BS EECS

2016

Harvard
PhD CS

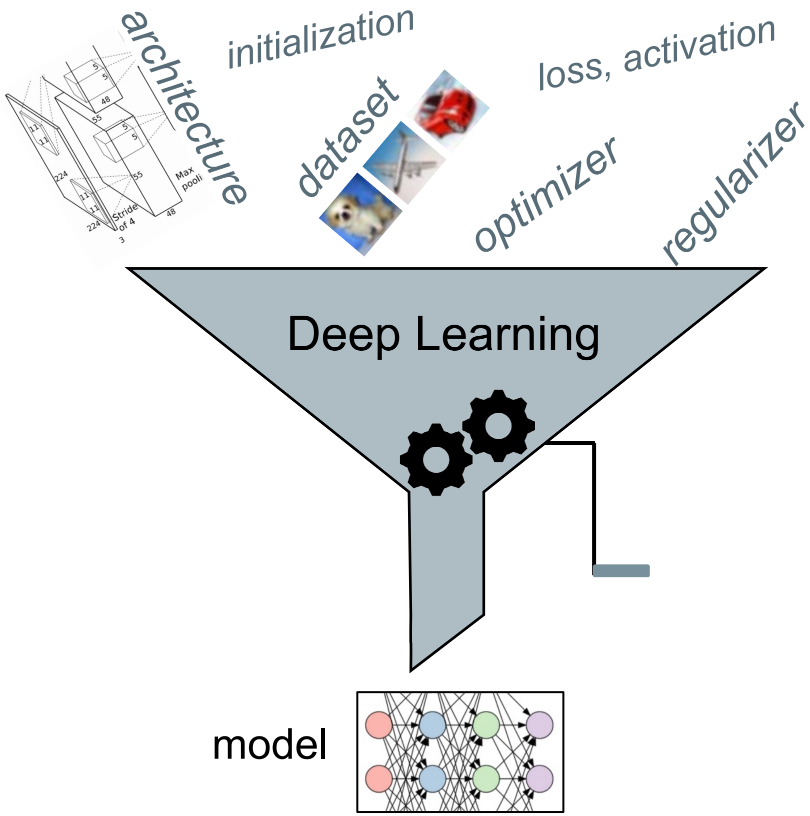
Madhu Sudan
Boaz Barak

2021

UCSD
Postdoc

Misha Belkin

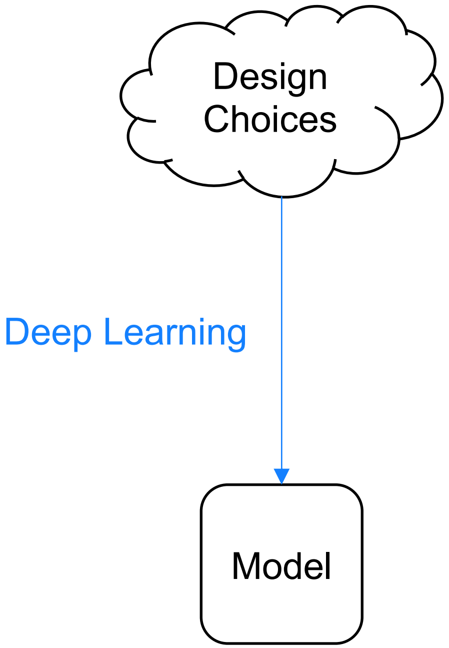
Goal: Understand Deep Learning



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

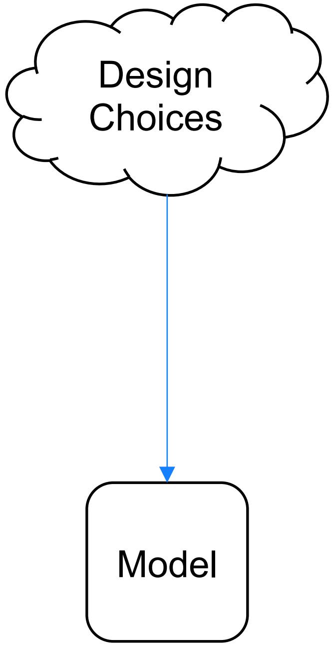
" How does what we ***do***
affect what we ***get?*** "

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



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

Methodology



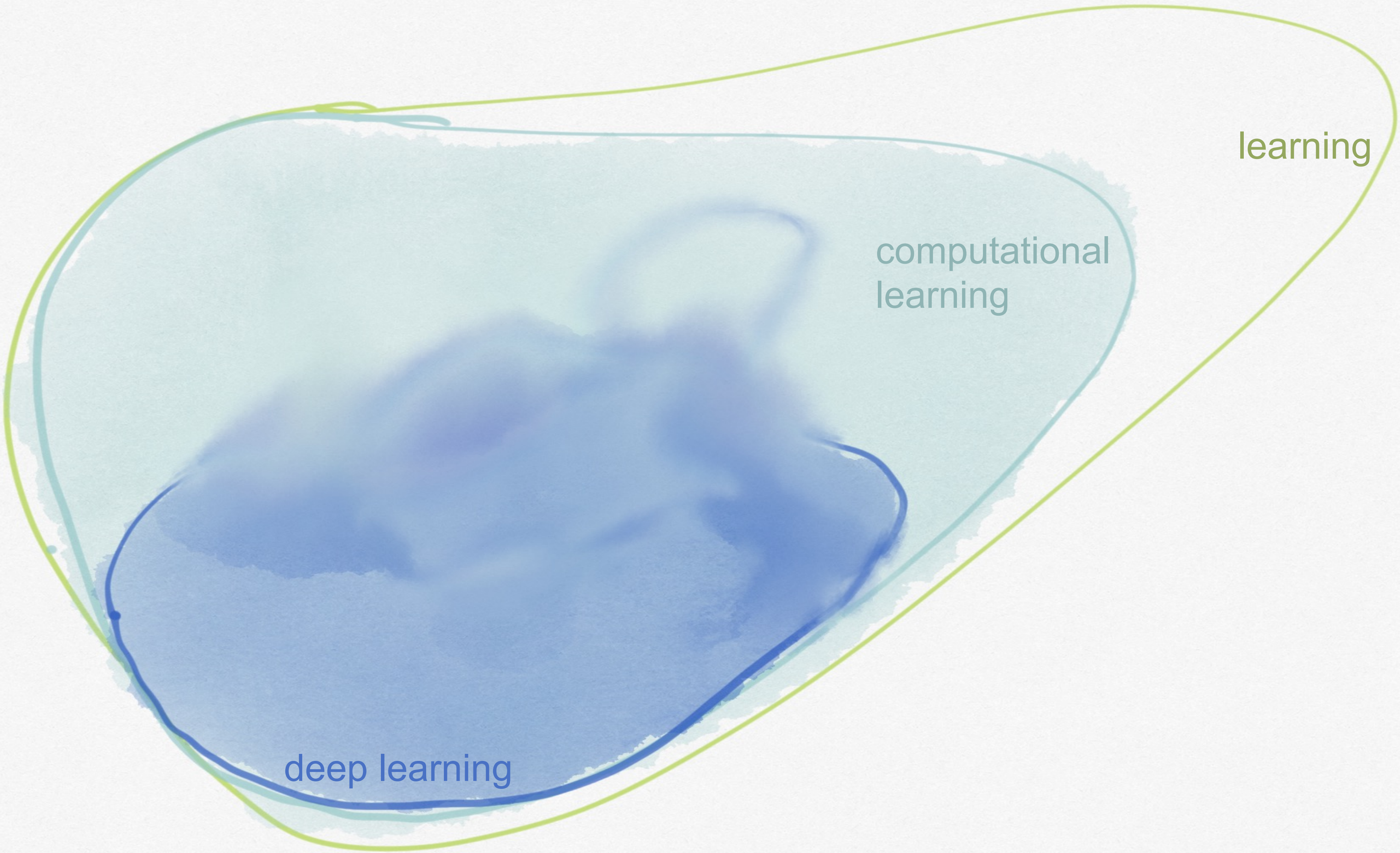
Two complementary ways to understand structure:

1. **Theorems**: Prove structure
    - "Gold-standard" when it's possible …but often not possible in DL
    - Complex system; precise understanding is difficult

2. **Experiments:** Empirically characterize structure
    - Do careful experiments, observe behavior, form quantitative conjectures
    - "Natural science" approach (e.g. Kepler's laws)

Experiments guide theory, theory informs experiments.

Conceptual tools for understanding learning systems



What can we learn, and how?

What does success of DL teach us about learning?

Selected Works

Double Descent

Deep Double Descent: Where Bigger Models and More Data Hurt
[**Nakkiran**, Kaplun*, Bansal*, Yang, Barak, Sutskever. ICLR 2020]

Optimal Regularization Can Mitigate Double Descent
[**Nakkiran**, Venkat, Kakade, Ma. ICLR 2021]

Classical Generalization

The Deep Bootstrap Framework: Good Online Learners are Good Offline Generalizers
[**Nakkiran**, Neyshabur, Sedghi. ICLR 2021]

SGD on Neural Networks Learns Functions of Increasing Complexity
[**Nakkiran**, Kaplun, Kalimeris, Yang, Edelman, Zhang, Barak. NeurIPS 2019]

Learning Rate Annealing Can Provably Help Generalization, Even for Convex Problems
[**Nakkiran**. ICML OPPO 2020]

Limitations of Neural Collapse for Understanding Generalization in Deep Learning
[Hui, Belkin, **Nakkiran.** Preprint 2022]

Selected Works

Fine-Grained Generalization

Distributional Generalization: A New Kind of Generalization
[**Nakkiran***, Bansal*. ICML OPPO 2021]

Deconstructing Distributions: A Pointwise Framework of Learning
[Kaplun*, Ghosh*, Garg, Barak, **Nakkiran.** Preprint 2022]

Distributional Generalization for Algorithm Design in Deep Learning
[Kulynych*, Yang*, Yu, Błasiok, **Nakkiran**. Preprint 2022]

Representation Learning

Revisiting Model Stitching to Compare Neural Representations
[Bansal, **Nakkiran**, Barak. NeurIPS 2021]

Adversarial Examples

Computational Limitations in Robust Classification and Win-Win Results
[Degwekar, **Nakkiran**, Vaikuntanathan. COLT 2019]

Adversarial Examples are Just Bugs, Too
[**Nakkiran**. Distill 2019]

This Talk

2020

Distributional Generalization: A New Kind of Generalization

Preetum Nakkiran* Harvard University

Yamini Bansal* Harvard University

2022

What You See is What You Get: Distributional Generalization for Algorithm Design in Deep Learning

Bogdan Kulynych [1*] Yao-Yuan Yang [2*] Yaodong Yu [3] Jarosław Błasiok [4] Preetum Nakkiran [2]

DISTRIBUTIONAL GENERALIZATION:
A New Kind of Generalization

Preetum Nakkiran* Yamini Bansal*
Harvard Harvard

ICML OPPO 2020

Supervised Classification

Setup:

Distribution D over pairs (input, label): $D \in \Delta(\mathcal{X} \times \mathcal{Y})$

Ex: Image Classification

$\mathcal{X}$ = { images of cats/dogs }

$\mathcal{Y}$ = { `'cat'`, `'dog'` }

Given:

IID samples from distribution: $S = \{(x_i, y_i)\}$

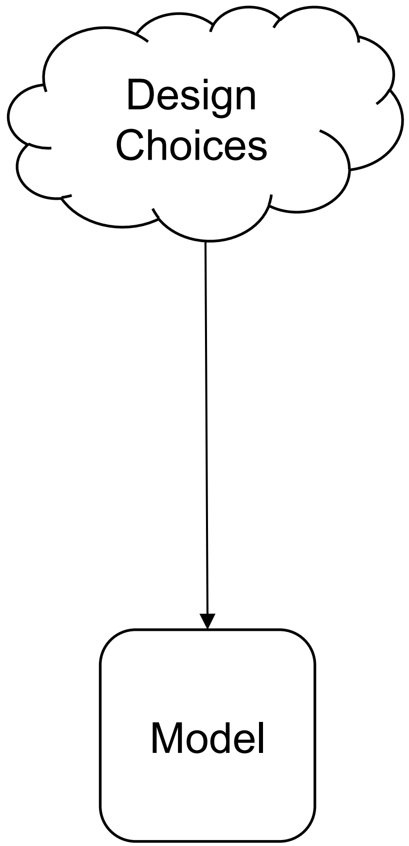
Want:

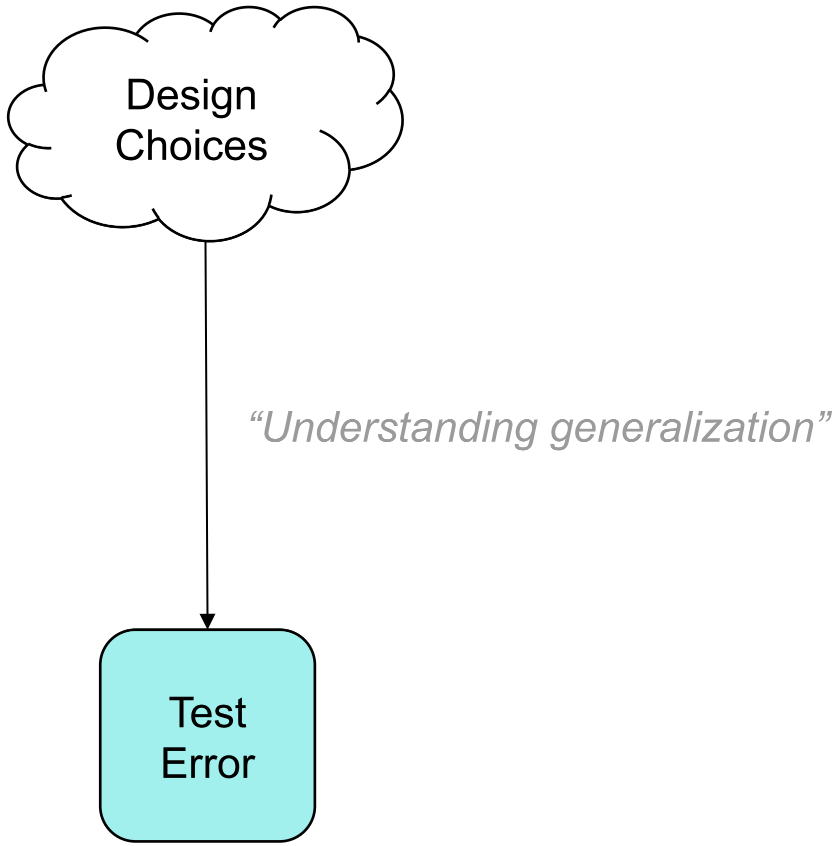
Find function $f: \mathcal{X} \to \mathcal{Y}$ with small test error:

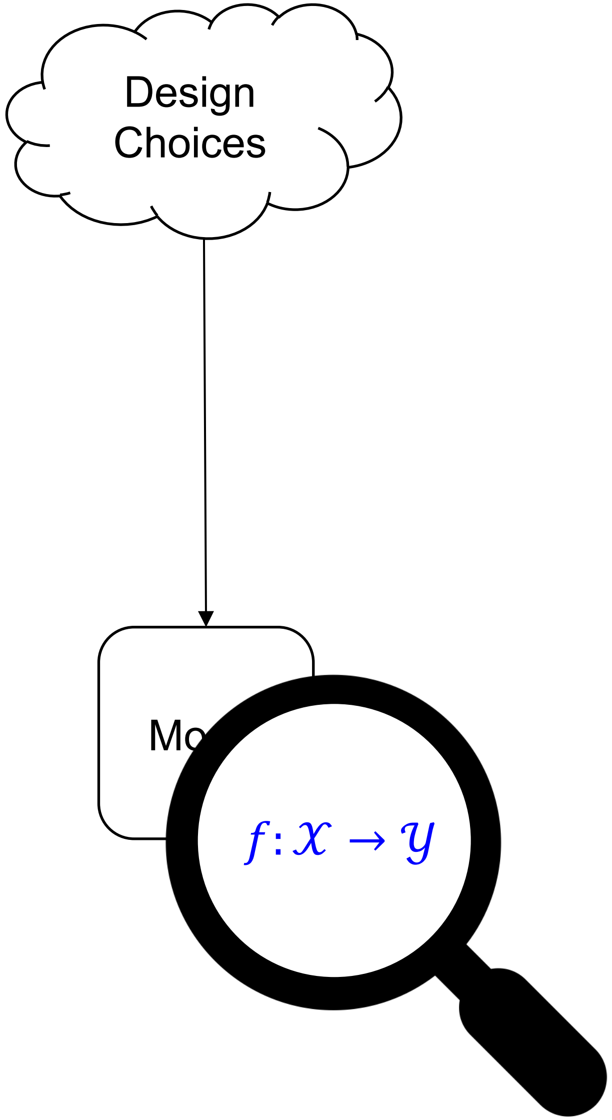
$$\text{TestError}(f) \coloneqq \Pr_{x,y \sim D}[f(x) \neq y]$$

Use learning procedure (SGD, DT,…):

$$\text{Learn} : S \mapsto f_S$$


Design
Choices
Model


Design Choices
"Understanding generalization"
Test Error



Suppose test error of f = 20%
Many such f! Which one did we get?

This work: Generalization beyond error…

This Work

1. **Mathematical language** for “fine-grained” generalization (beyond error)
2. **Experiments** showing new kinds of generalization “in the wild”
3. **Conjectural theory** unifying experiments

AlexNet Example

ImageNet: Image classification. 1000-classes, 116 dogs.

AlexNet : ~56% test accuracy

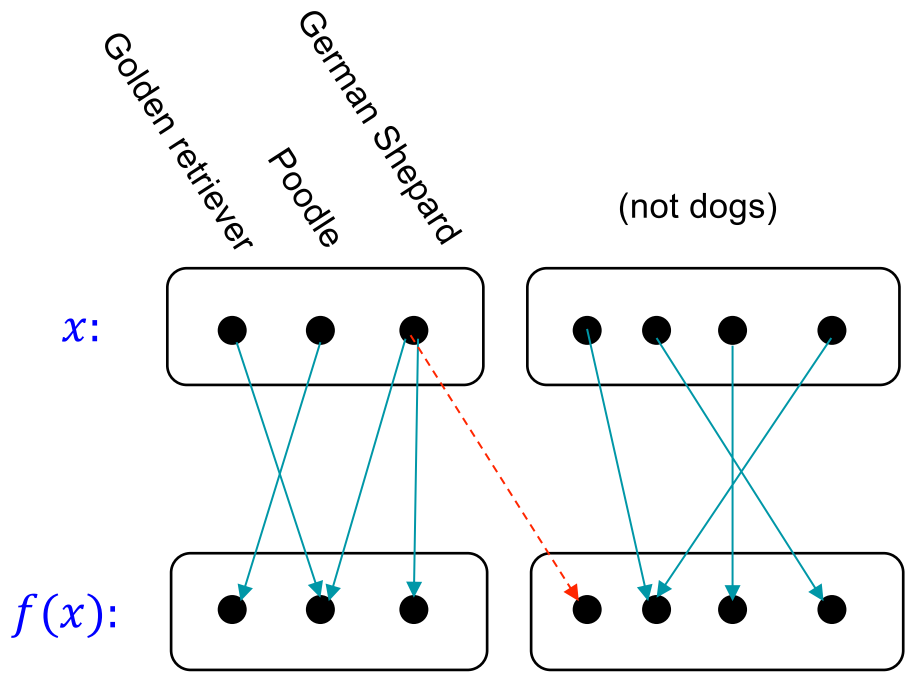
Many ways to get 56% accuracy…

Does AlexNet at least classify dogs as *some type* of dog?

- Yes! (98% acc).

- *Even "bad" classifiers (w.r.t. test error), can have "good" structure*

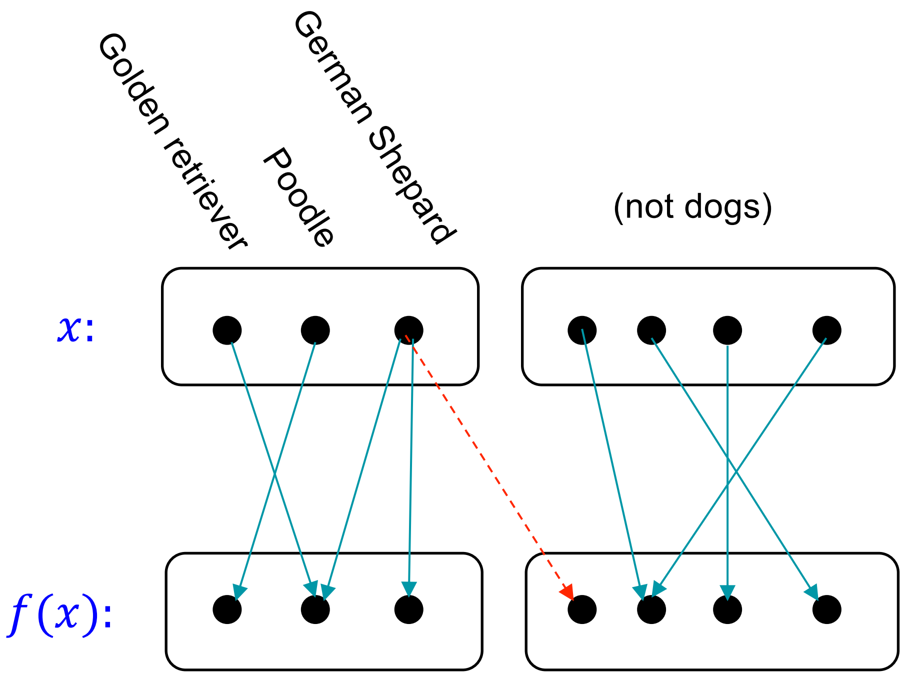
- Not captured by classical generalization



AlexNet Example

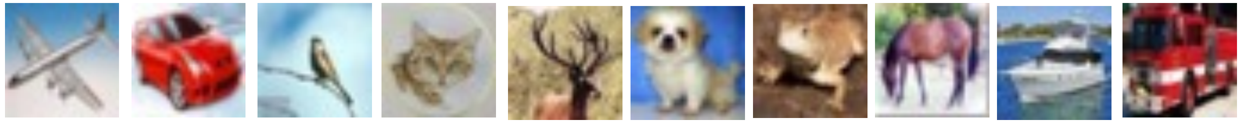
- Didn't tell AlexNet what "dogs" are.

- Want: general language to describe this type of behavior



Noisy-Binary-CIFAR

Type: 0 1 2 3 … 9

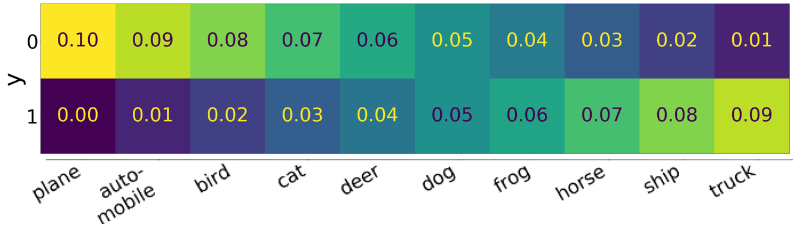


Distribution on (x, y):

$x \sim$ { random CIFAR-10 image }

$y|x \sim$ Bernoulli(type(x) / 10)

Train Set (x, y)



Sample n=50K from this distribution.

Train a ResNet to interpolation,

to predict $f : \mathcal{X} \rightarrow \mathcal{Y}$

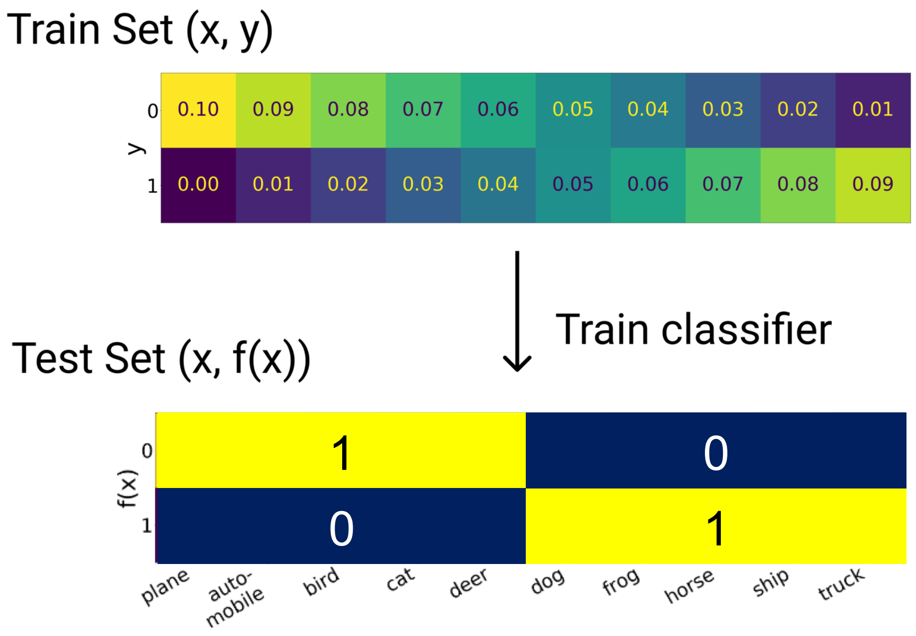
<u>Q:</u> What happens at test time?

<u>A:</u> ~Same distribution!

We use a method for **classification.**
We **don't get** a good classifier: high test error!

We get an approximate **sampler**:

$$f(x) \sim p(y \mid x)$$



Happens for:

- Interpolating **neural networks**
- Interpolating **kernel regressors**
- Interpolating **decision trees**

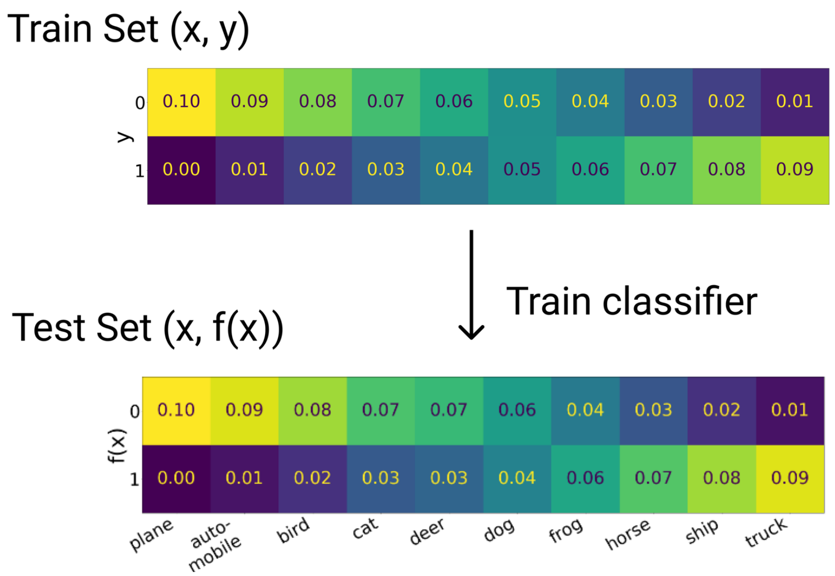
Best thought of as samplers.

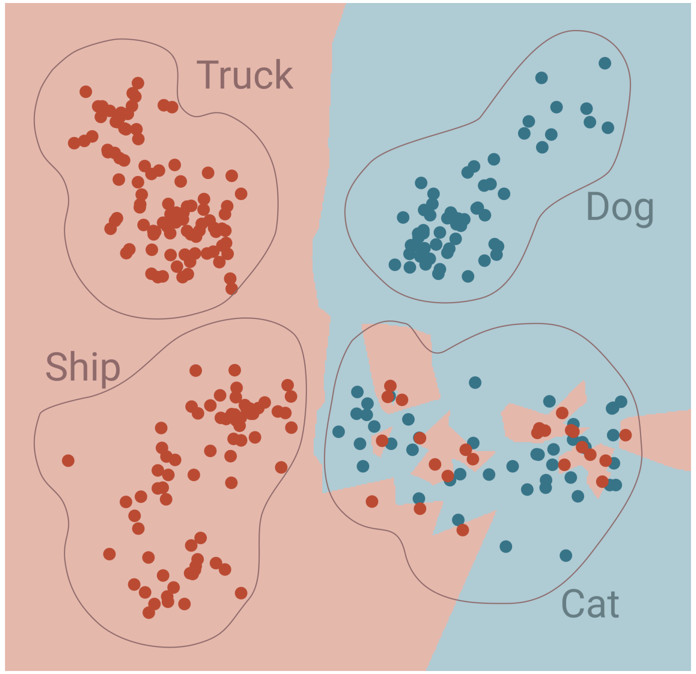
Classical generalization is insufficient language:
Large generalization gap… but "distributional generalization"

Classifier sensitive to *subclass-structures*

1-Nearest-Neighbors (in a well-clustered space) would have the same behavior.

… but why do ResNets?





Distributional Generalization

Key idea: two distributions over $\mathcal{X} \times \mathcal{Y}$

This fully determines $f : \mathcal{X} \to \mathcal{Y}$ on-distribution!

<u>Train Distribution</u>	<u>Test Distribution</u>
$(x_i, f_S(x_i))_{S; x_i \sim S}$	$(x, f_S(x))_{S; x \sim \mathcal{D}}$

Classical Generalization asks:

When is Error(Train) ≈ Error(Test)?

Distributional Generalization asks:

*“When (and **in what sense**) are Train & Test outputs are close **as distributions**”*

$$\boxed{(x, f(x))_{x \in \text{TrainSet}} \overset{?}{\approx} (x, f(x))_{x \in \text{TestSet}}}$$

Distributional Generalization

Key idea: two distributions over $\mathcal{X} \times \mathcal{Y}$

This fully determines $f : \mathcal{X} \to \mathcal{Y}$ on-distribution!

Train Distribution

$$(x_i, f_S(x_i))_{S; x_i \sim S}$$

Test Distribution

$$(x, f_S(x))_{S; x \sim \mathcal{D}}$$

Defn: *A learning procedure satisfies* **classical generalization** *if:*

$$\text{TrainError}(f_S) \approx \text{TestError}(f_S)$$

$$\mathbb{E}_{S;\, x_i \sim S}\left[\, \ell\big(x_i, f_S(x_i)\big) \,\right] \approx \mathbb{E}_{S;\, x \sim \mathcal{D}}[\ell(x, f_S(x))]$$

for loss function $\ell(x, \hat{y}) \;:= \mathbb{I}\{\hat{y} \neq y^*(x)\}$

predicted label

true label

Train and Test are "close" w.r.t. loss ℓ.
Are they also close for other losses?

Train Distribution **Test Distribution**

$$(x_i, f_S(x_i))_{S; x_i \sim S} \qquad (x, f_S(x))_{S; x \sim \mathcal{D}}$$

Defn. *A training procedure satisfies* **classical generalization** *if:*

$$\mathbb{E}_{S;\, x_i \sim S}\left[\, \ell\left(x_i, f_S(x_i)\right) \,\right] \approx \mathbb{E}_{S;\, x \sim \mathcal{D}}[\ell(x, f_S(x))]$$

Defn. *A training procedure satisfies* ***𝒯*-distributional generalization** *(𝒯–DG) for a family of tests 𝒯*

$$\mathcal{T} \subseteq \{T: \mathcal{X} \times \mathcal{Y} \to [0,1]\}$$

if

$$\forall\, T \in \mathcal{T}: \quad \mathbb{E}_{S;\, x_i \sim S}\left[\, T\left(x_i, f_S(x_i)\right) \,\right] \approx \mathbb{E}_{S;\, x \sim \mathcal{D}}[T(x, f_S(x))]$$

Ex:

1. $\mathcal{T} = \{\ell\}$ $\Leftrightarrow$ classical generalization
2. $\mathcal{T} = \{\ell_g\}_{g \in G}$, ℓ_g= "*subgroup loss on g*" $\Leftrightarrow$ generalization of subgroup-errors
3. $\mathcal{T} = \{all\ bounded\ tests\}$ $\Leftrightarrow$ *TV-closeness*

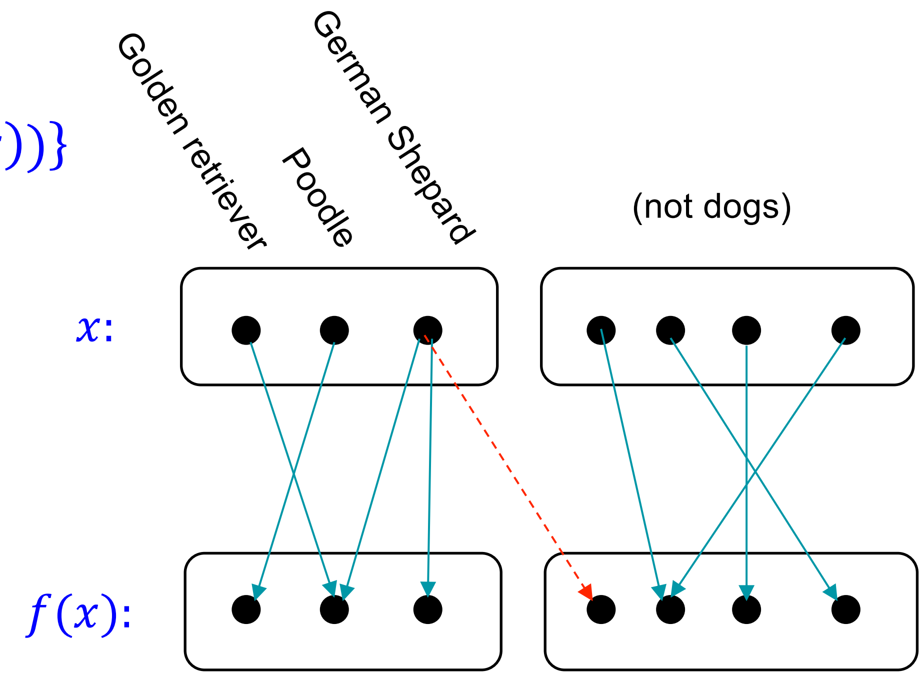
AlexNet Example

predicted label true label

Overall error: $\ell(x, \hat{y}) := \mathbb{I}\{\hat{y} \neq y^*(x)\}$

Dog-coarsened error: $\ell_{dog}(x, \hat{y}) := \mathbb{I}\{\mathrm{Dog}(\hat{y}) \neq \mathrm{Dog}(y^*(x))\}$

$\mathrm{Dog}: \mathcal{Y} \to \{0, 1\}$



Train Distribution

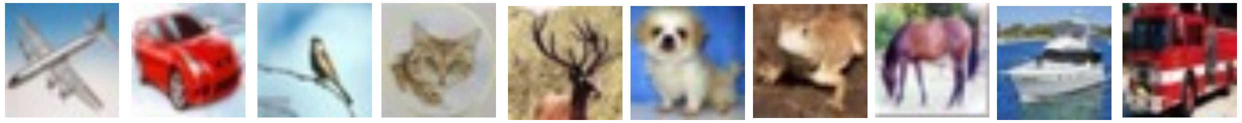
Test Distribution

$(x_i, f_S(x_i))_{S; x_i \sim S} \approx^{\ell_{dog}} (x, f_S(x))_{S; x \sim \mathcal{D}}$

“AlexNet classifies most dog-images as some type of dog”

Noisy-Binary-CIFAR

Type: 0 1 2 3 … 9



f interpolates S

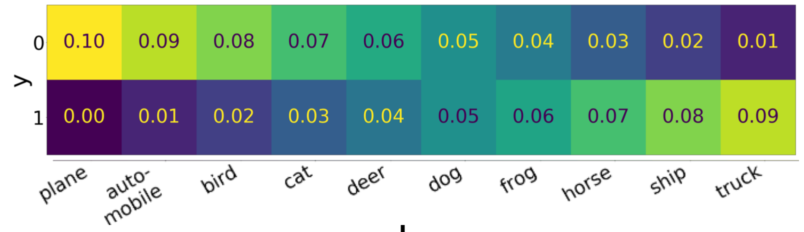
<u>Source</u> <u>Train Distribution</u> <u>Test Distribution</u>

$$(x, y)_{x,y\sim D} \equiv (x_i, f_S(x_i))_{S; x_i \sim S} \approx (x, f_S(x))_{S; x \sim \mathcal{D}}$$

For the partition $L: x \mapsto \mathrm{Type}(x)$

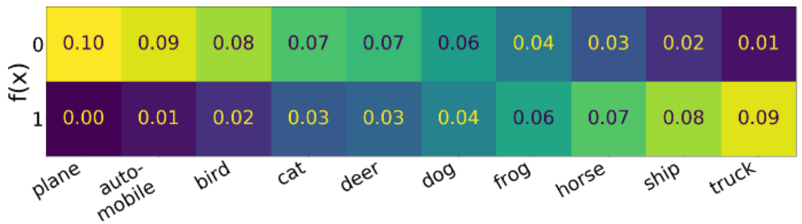
$$(L(x), y) \approx_{TV} \left(L(x), f_S(x)\right)$$

Train Set (x, y)

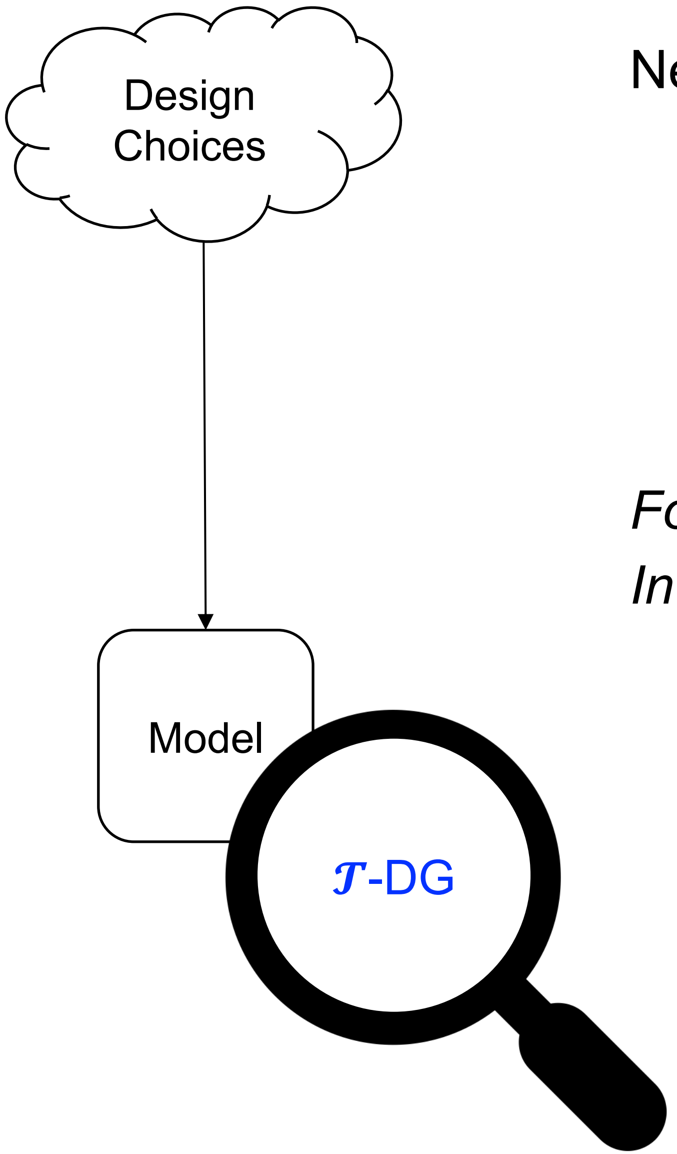


Train classifier

Test Set (x, f(x))



The story so far…



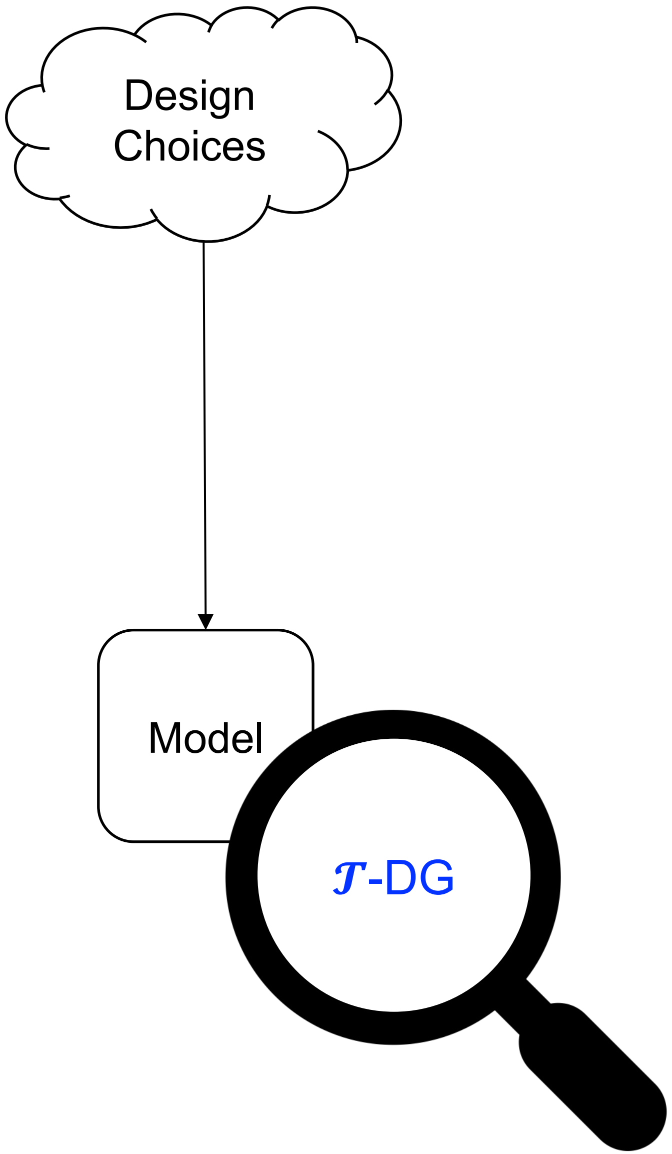
New **definition** of generalization:

language for describing **WHAT** happens in experiments

*…but **WHEN** does it happen?*

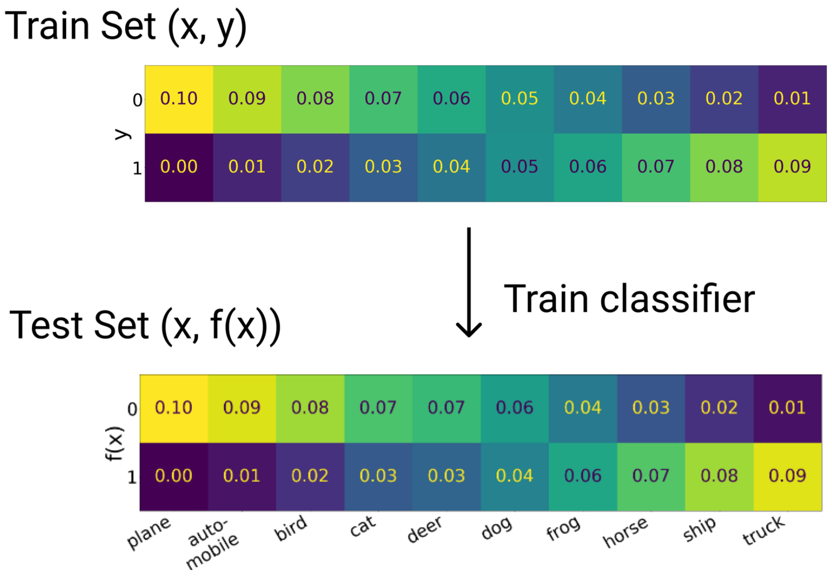
For given design choices, what $\mathcal{T}$-DG holds?

In what sense are Train & Test distributions close?



This is subtle!

Depends on architecture, distribution, num samples…

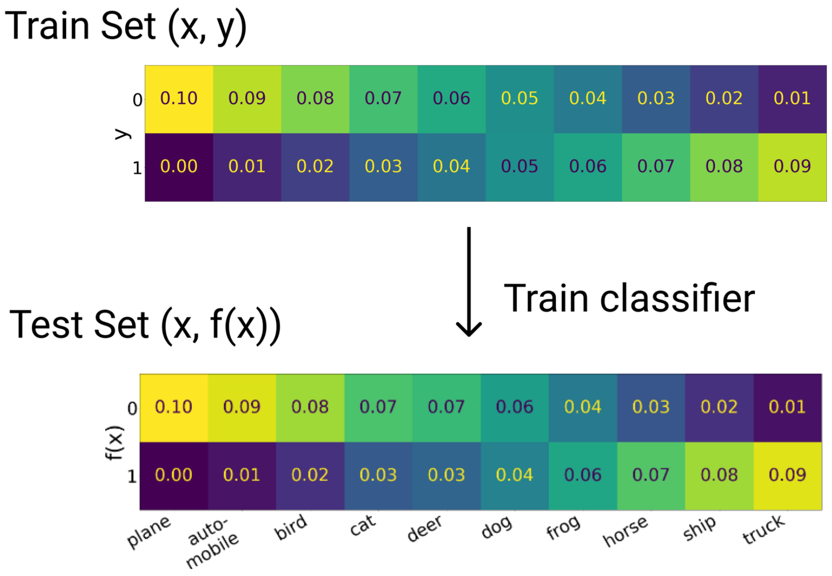


Feature Calibration

<u>Conjecture (informal)</u>:

> *Marginal distributions of $\boldsymbol{f(x)}$ and $\boldsymbol{y}$ match, when conditioned on any "good" subgroup $\boldsymbol{L(x)} \in \{\mathbf{0}, \mathbf{1}\}$*

Eg: $p(f(x) \mid x \in CAT) \approx p(y \mid x \in CAT)$



What is a "good" subgroup?

- Many "good subgroups"! (cats, animals, objects,…)

- <u>**Defn:**</u> "Good" subgroups are those which are ***themselves learnable***, *by the same procedure*

Feature Calibration

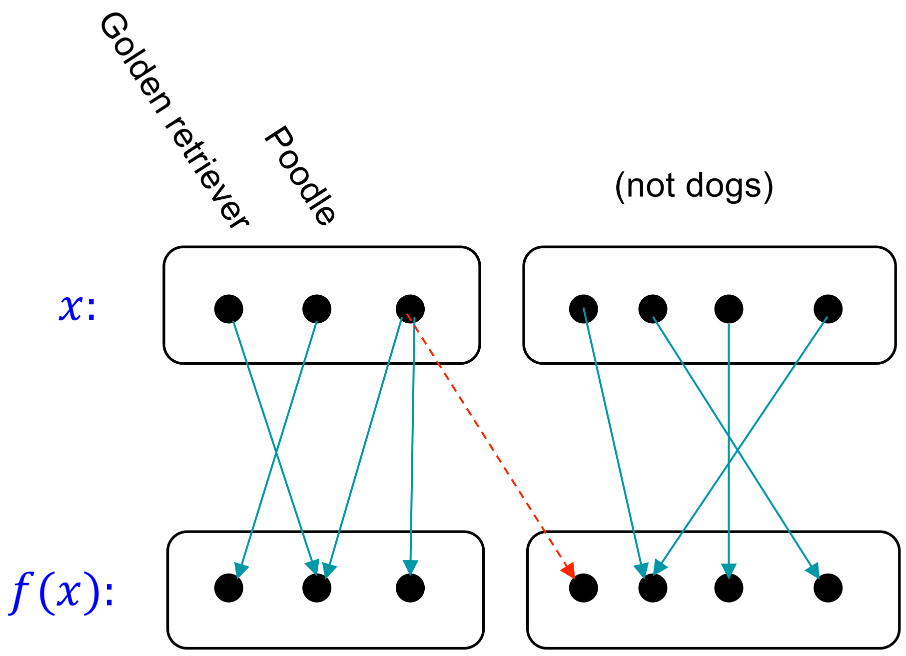
<u>Conjecture (informal)</u>:

Marginal distributions of $\boldsymbol{f(x)}$ and $\boldsymbol{y}$ match,
when conditioned on any "good" subgroup $\boldsymbol{L(x)} \in \{\mathbf{0}, \mathbf{1}\}$

Eg: $p(f(x) \mid x \in DOGS) \approx p(y \mid x \in DOGS)$

*"**IF** AlexNet could learn to classify dogs vs. not-dogs (when trained on this binary task),*

***THEN** AlexNet will classify most dogs as dogs (when trained on 1000-class ImageNet)"*



See the paper for..

Formal statement of Feature Calibration Conjecture

"classifiers are approximate density estimators"

Many **experiments testing conjecture** (NNs, kernels, decision trees)

Proof of Conjecture for 1-Nearest-Neighbor (non-asymptotic)

Significance

Guidance for Theory:

Try to understand DG, not just classical generalization.

Even classifiers with "bad" have certain "good" structure.

Important to understand models as functions $f: \mathcal{X} \to \mathcal{Y}$

Implicit Bias: Many models with same train *and test* errors. Which one do we get?

DG = "universal implicit bias" of interpolating models.

Benign Overfitting:

Interpolation not always "benign": Noise in train → noise in test.

Ensembling:

Approximate-sampling connection suggests noise-reduction benefits of ensembling.

What You See is What You Get: Distributional Generalization for Algorithm Design in Deep Learning

Bogdan Kulynych*
EPFL

Yao-Yuan Yang*
UCSD

Yaodong Yu
Berkeley

Jarosław Błasiok
Columbia

Preetum Nakkiran
UCSD

In submission to ICML 2022

Differential Privacy $\Longrightarrow$ Distributional Generalization

Recall: Strongest form of Distributional Generalization = **Total-variation closeness**

Train Distribution		Test Distribution
$(x_i, f_S(x_i))_{S; x_i \sim S}$	$\approx^{\mathrm{TV}}$	$(x, f_S(x))_{S; x \sim \mathcal{D}}$

Usually too strong to hold. But it holds for DP-algorithms!

Training procedure is **Differentially Private** $\Longrightarrow$ Learnt model **distributionally-generalizes** in TV

(DP-SGD, etc…)

$$(x, f(x))_{x \in \text{TrainSet}} \approx^{\mathrm{TV}} (x, f(x))_{x \in \text{TestSet}}$$

Related to "robust generalization" [Cummings et al 2016]

What You See is What You Get

$$(x, f(x))_{x \in \text{TrainSet}} \approx_{\text{TV}} (x, f(x))_{x \in \text{TestSet}}$$

DG in Total-Variation is **"WYSIWYG generalization"**
What you **see (on train set)** is what you **get (at test time).**

<u>WYSIWYG Implies:</u>

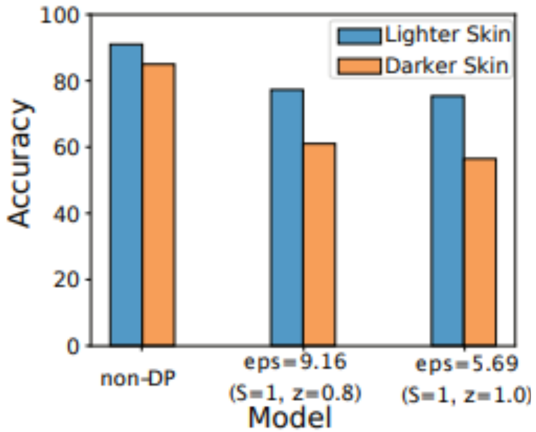
- TrainError ≈ TestError
- For all subgroups $S \subseteq \mathcal{X}$: TrainError(S) ≈ TestError(S)
- Train calibration (ECE) ≈ Test calibration (ECE)
- Adversarial robust train loss ≈ Adversarial robust test loss

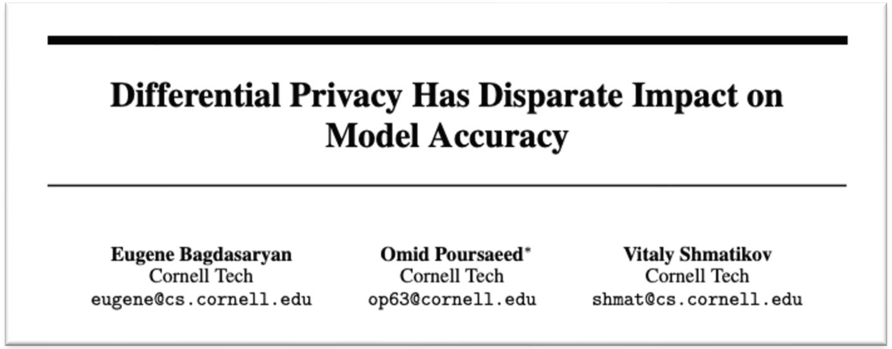
*Generalization for **all properties**, not just average error*

Why is WYSIWYG useful?

Reason 1: Diagnostic

Any “bad behavior” of the model (at test time) is detectable at train time.





Differential Privacy Has Disparate Impact on Model Accuracy

Eugene Bagdasaryan, Cornell Tech, eugene@cs.cornell.edu
Omid Poursaeed*, Cornell Tech, op63@cornell.edu
Vitaly Shmatikov, Cornell Tech, shmat@cs.cornell.edu

Ex: Using DP has disparate impact at test time… “because” it has disparate impact on train set

WYSIWYG = “no surprises at test time”

Why is WYSIWYG useful?

Reason 2: Algorithmic

"Fix" any bad test behavior by fixing it **on the train set**
(not true for standard SGD!)

Why is WYSIWYG useful?

CelebA subgroups: {male, female} × {blonde, not-blond}

blond-male is a **minority** subgroup

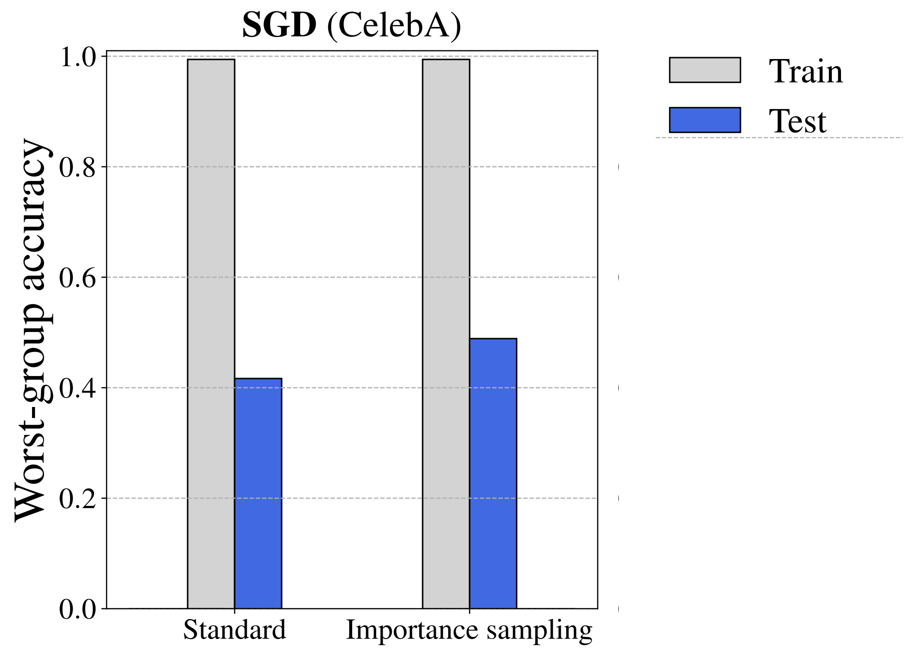
Standard SGD:

Train accuracy:	100%
Test accuracy:	95%
Worst-group accuracy:	42%

Attempted fix: **Importance sampling/weighting**

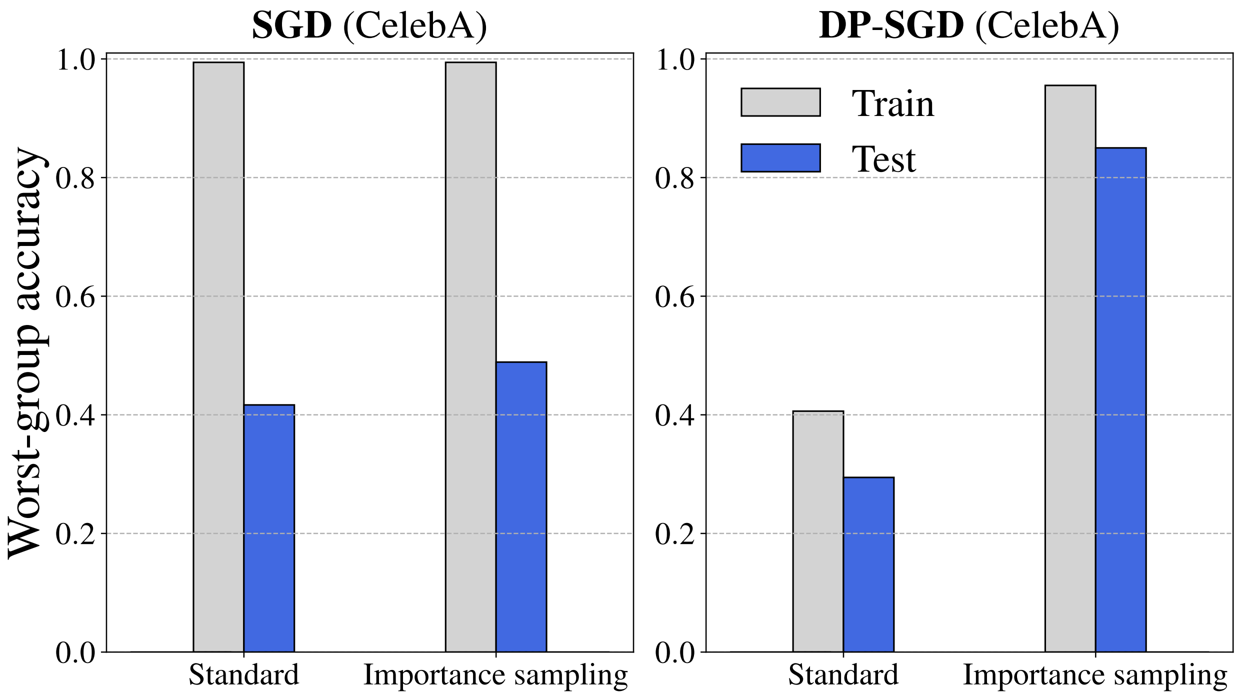
(up-weight minority group in train loss)

This works for small models... **fails for overparameterized nets** [Byrd, Lipton 2018]



Why is WYSIWYG useful?

With DP-SGD: **Importance-sampling works as expected!**



WYSIWYG: fix train behavior → fix test behavior

DP useful even when privacy not required

Blueprint for Algorithm Design

1. Optimize for
   good behavior on train set
2. Compose with DP (or other DG-method) to obtain
   good behavior on test set

Step (2): DP is only one path to DG
in general, **regularization** induces distributional generalization

See the paper for…

More Applications:

- Simple, SOTA-competitive algorithms for Distributional Robustness
- Improve disparate impact at all privacy budgets (via Importance-sampled DP-SGD)

Theory:

- Tight bounds between DP, TV-stability, and Distributional Generalization
- Algorithm & privacy-analysis of Importance-Sampled DP-SGD

Heuristic Experiments:

- Evidence that “many regularizers induce DG”

CONCLUSIONS

We…

Looked closer at models as **functions** $(f: \mathcal{X} \to \mathcal{Y})$,

and defined **new kinds of generalization** to capture behavior.

Connected tools to differential-privacy, to understand & improve some applications.

Not specific to deep learning! General tools for understanding learning systems…

Many more open questions…

"New kinds of generalization" in large language models?

What is "special" about deep learning?

What is learnt in pretraining?

What matters about architecture?

Why do we "learn representations"?

….

What does Deep Learning teach us about learning?

Thanks!

preetum@ucsd.edu