



Understanding Deep Learning (Still) Requires Rethinking Generalization

Authors: C. Zhang, S. Bengio, M. Hardt,
B. Recht, and O. Vinyals

By Subas Rana and Afsaneh Shams

Agenda

1. Research Question
2. Introduction
3. Effective capacity of neural networks
4. The role of regularization
5. Experimental findings
6. Finite-sample expressivity
7. Conclusion
8. Related work



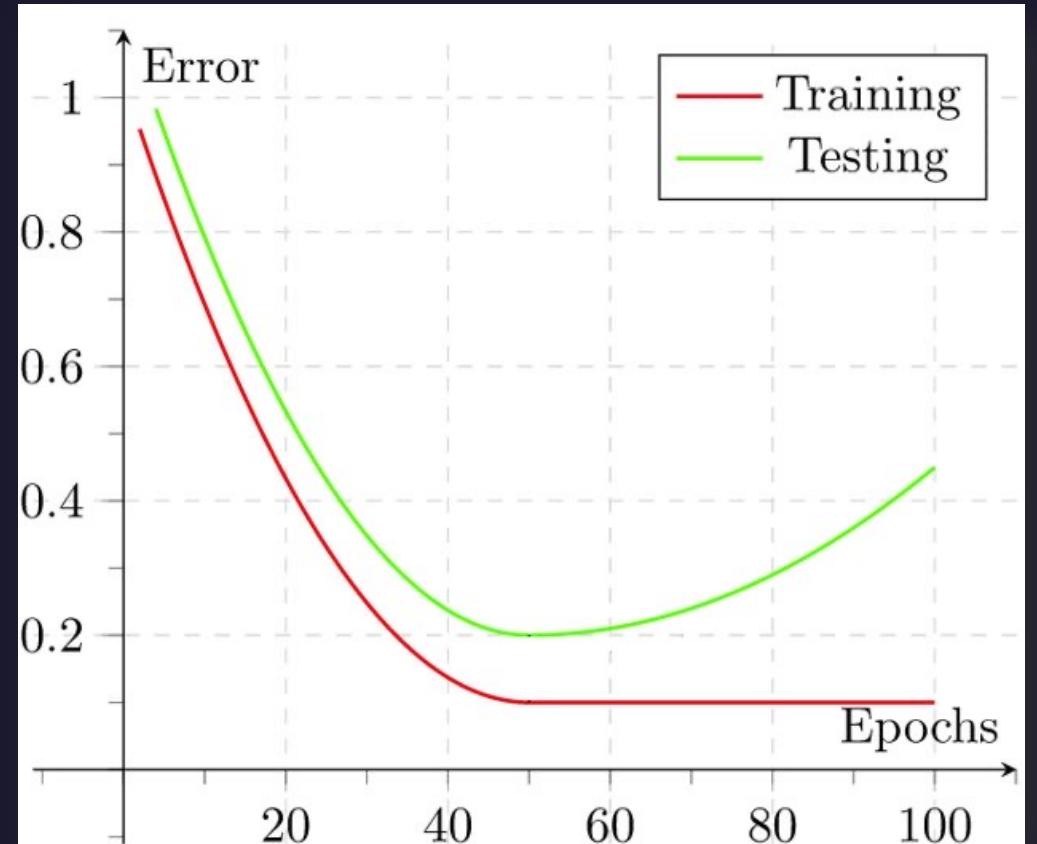
Research question

"What practices do and do not promote generalization, and what does and does not measure generalization?"



Generalization error

Difference between **training error** and **test error**



Introduction



Introduction

Deep neural networks can exhibit a small gap between training and test performance

Conventional wisdom attributes this to properties of the model or regularization techniques

It tries to mitigate some of the existing misconceptions

Through a host of randomization tests and touches upon how generalization error may or may not be related to regularization

It also throws light on some interesting points like finite-sample expressivity of neural nets

Randomization

Primary experiment

- Replace each label independently with a random label chosen from the set of valid labels.
- The randomization breaks any relationship between the image and the label
- Identical settings and model choice
- If it turns out to be the same in both cases, it could not possibly be a good measure of generalization



”Deep neural networks easily fit random labels”

Randomization

Primary experiment

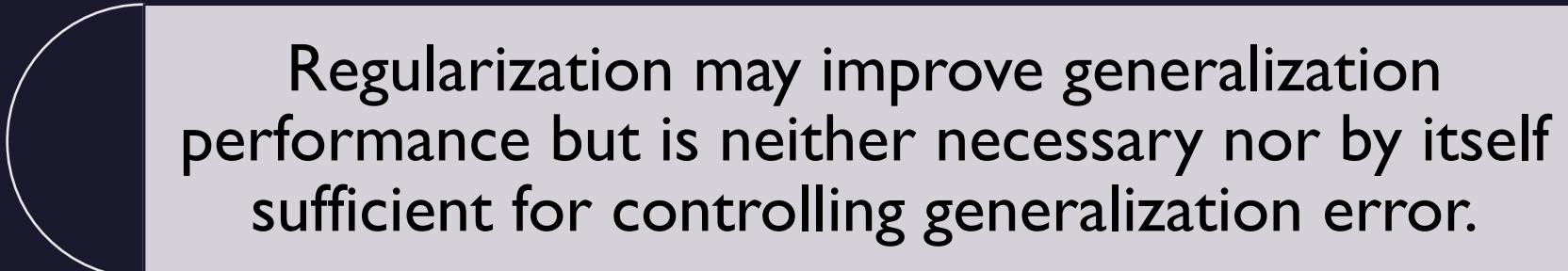
- 0 training error
- Test error is no better than random guessing
- CIFAR10 and ImageNet

Implication:

- The effective capacity of neural networks is sufficient for memorizing the entire data set.
- Training time increases only by a small constant factor compared with training on the true labels.
- Randomizing labels is solely a data transformation, leaving all other properties of the learning problem unchanged.

The role of regularization

- Regularization?
- Train overparameterized neural networks
- Implicit and Explicit Regularization



Regularization may improve generalization performance but is neither necessary nor by itself sufficient for controlling generalization error.



Finite-sample Expressivity



A very simple two-layer ReLU network with $p = 2n + d$ parameters can express any labeling of any sample of size n in d dimensions.



Effective Capacity of Neural Networks



Effective Capacity Of Neural Networks

- The size of a model family is often huge as it counts all possible functions in a certain set.
- By effective capacity, we informally refer to the size of the subset of models that is effectively achievable by the learning procedure.
- “Well-behaved” functions produced by some specific optimization algorithms, with bounded computation budget, and sometimes with explicit or implicit regularizations.

Effective Model Capacity – Feed-Forward Neural Network

- Choose a methodology inspired by nonparametric randomization tests.
- There is no longer any relationship between the instances and the class labels.
- Surprisingly, several properties of the training process for multiple standard architectures are largely unaffected by this transformation of the labels.

Experimental Findings

Dataset

CIFAR10:

50,000 Train

10,000 Test

10 Classes

32, 32, 3

airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



ImageNet:

1,281,167 Train

50,000 Validation

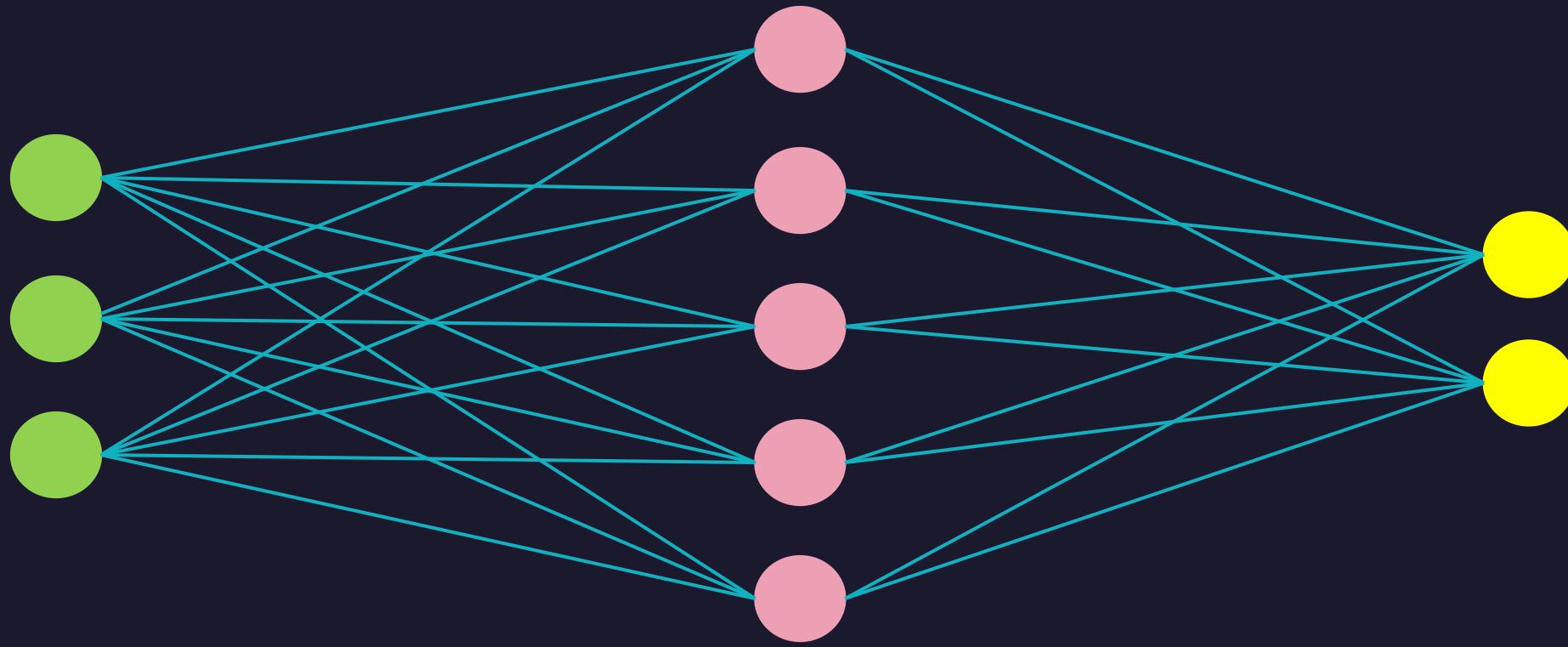
100,000 Test

1000 classes

299, 299, 3



Experimental Setup



Input

Shuffled pixels

Random pixels

Gaussian

Machine

Inception (V3)

Alexnet

MLPs

Output

True labels

Partially corrupted labels

Random labels

Randomization Test

- **True labels**

The original dataset without modification.



Human

Monkey

Bird

Cat

Deer

Dog

Frog

Building

Ship

Truck



Randomization Test (cont.)

- True labels
- Partially corrupted labels

Independently with probability p , the label of each image is corrupted as a uniform random class.



Human

Monkey

Bird

Cat

Deer

Dog

Frog

Building

Ship

Truck

Randomization Test (cont.)

- True labels
- Partially corrupted labels
- **Random labels**

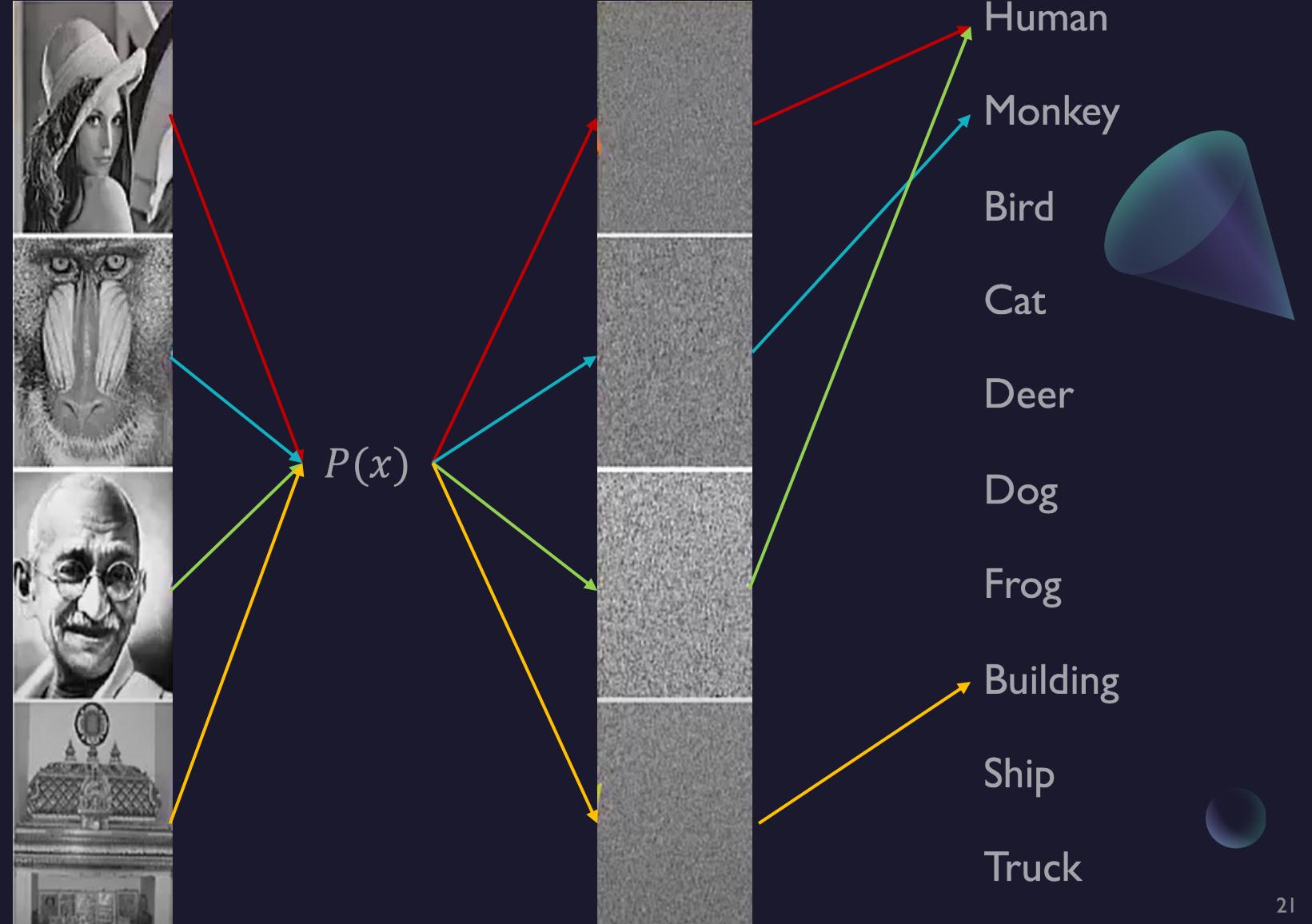
All the labels are replaced with random ones.



Randomization Test (cont.)

- True labels
- Partially corrupted labels
- Random labels
- **Shuffled pixels**

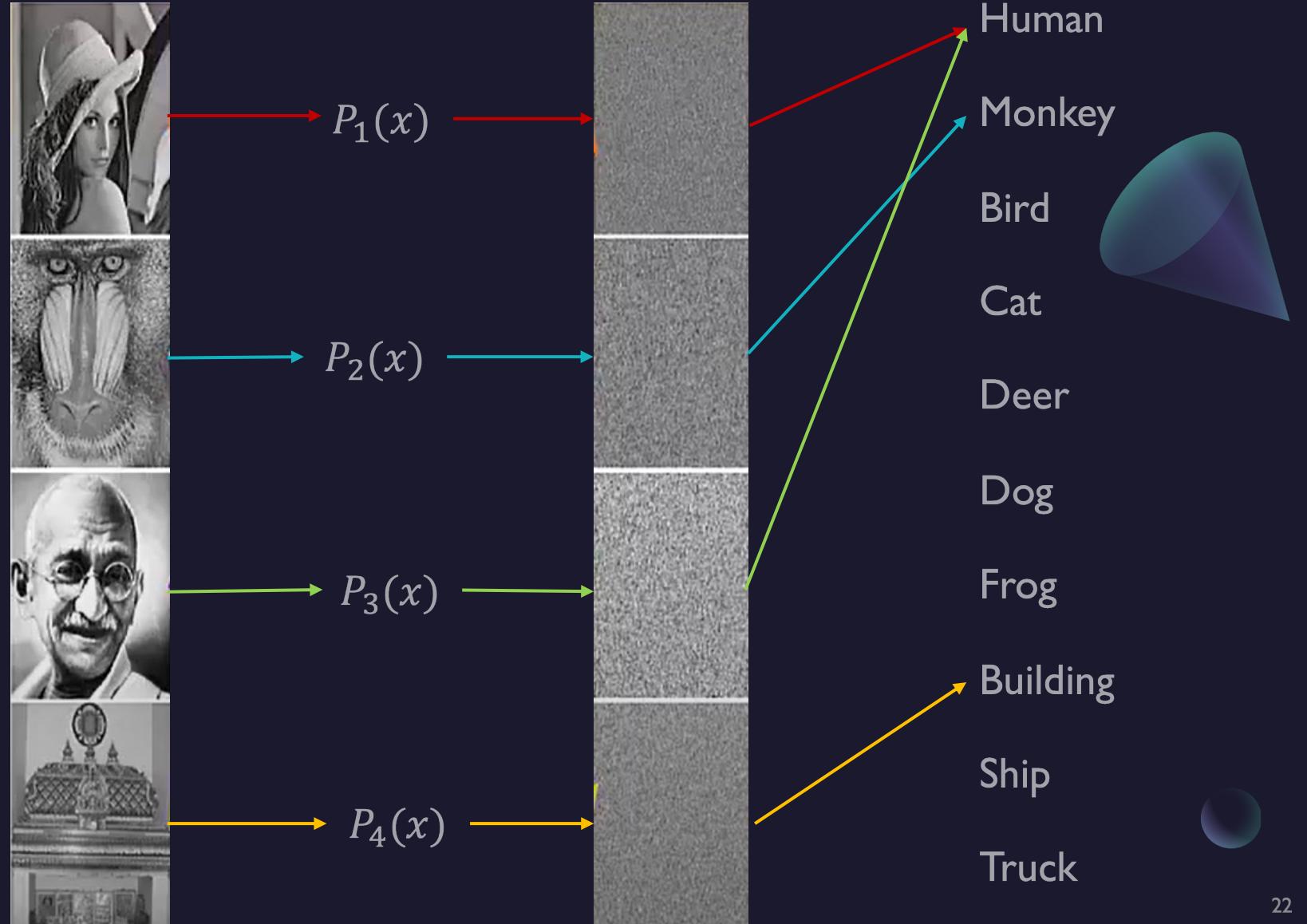
A random permutation of the pixels is chosen and then the same permutation is applied to all the images in both training and test set.



Randomization Test (cont.)

- True labels
- Partially corrupted labels
- Random labels
- Shuffled pixels
- **Random pixels**

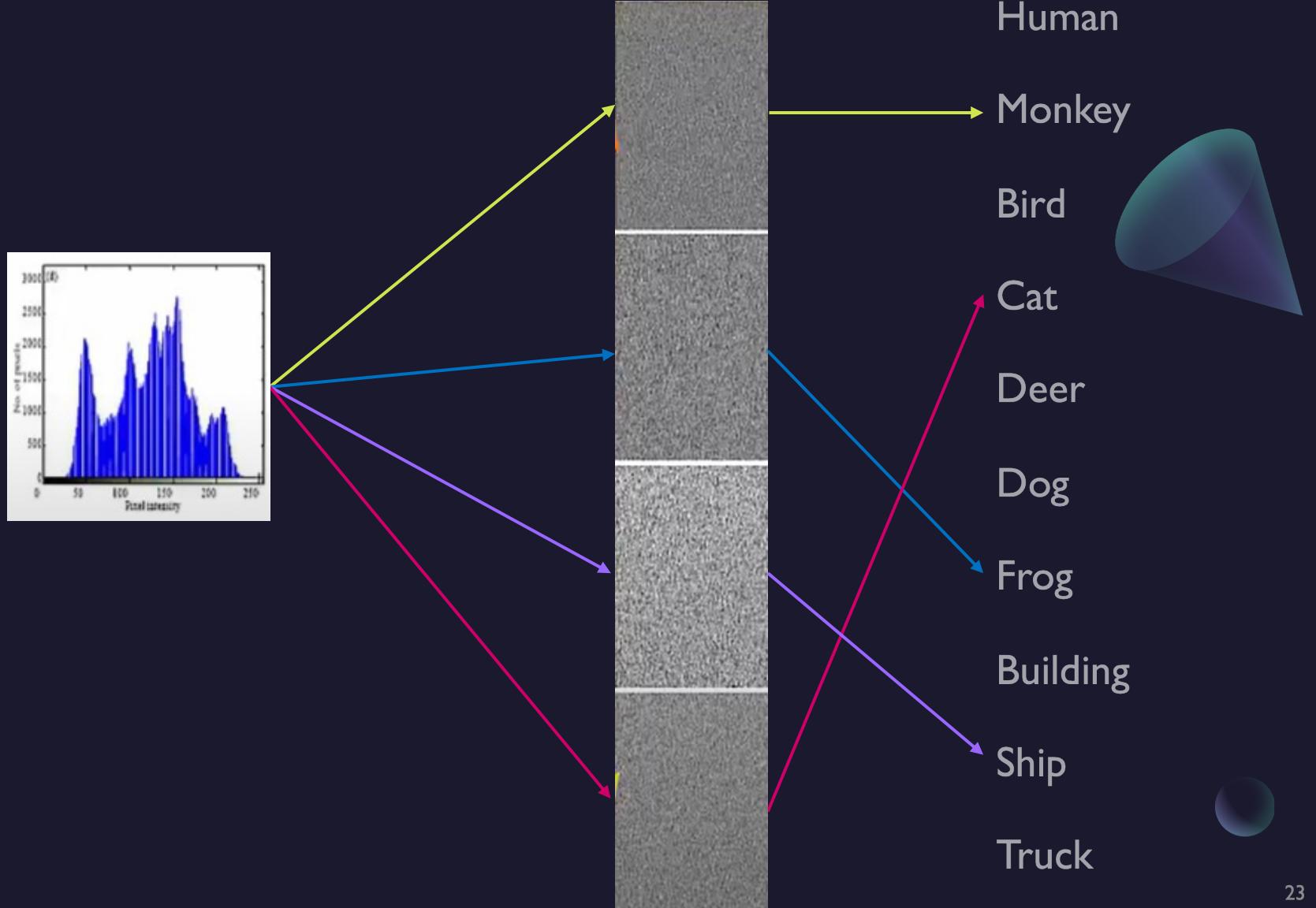
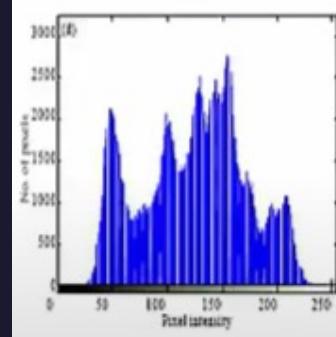
A different random permutation is applied to each image independently.



Randomization Test (cont.)

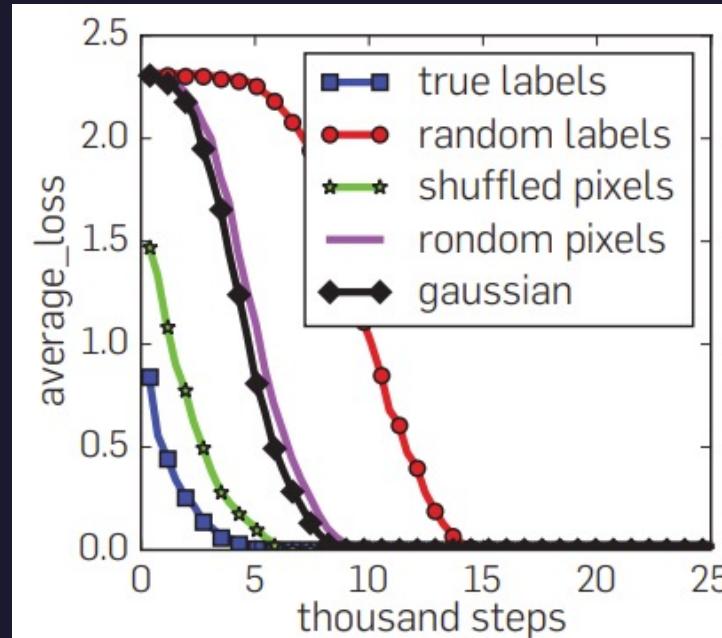
- True labels
- Partially corrupted labels
- Random labels
- Shuffled pixels
- Random pixels
- **Gaussian**

A Gaussian distribution is used to generate random pixels for each image.

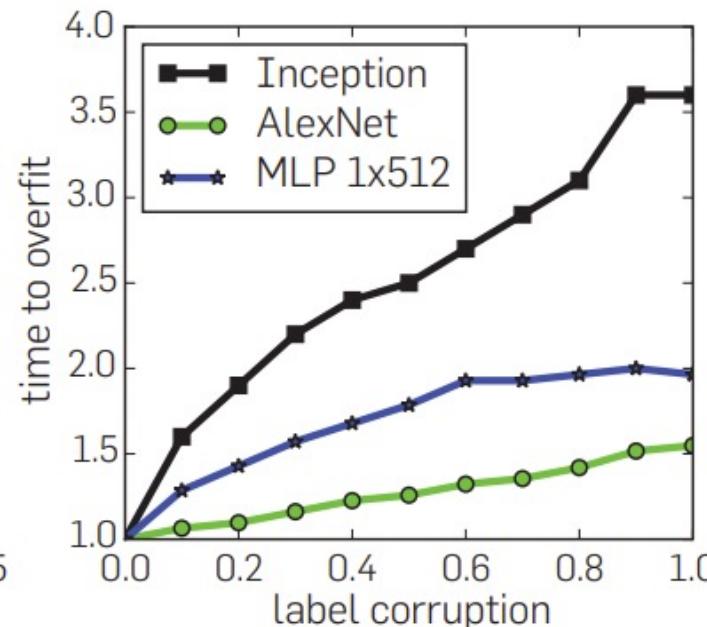


Results of Randomization Tests

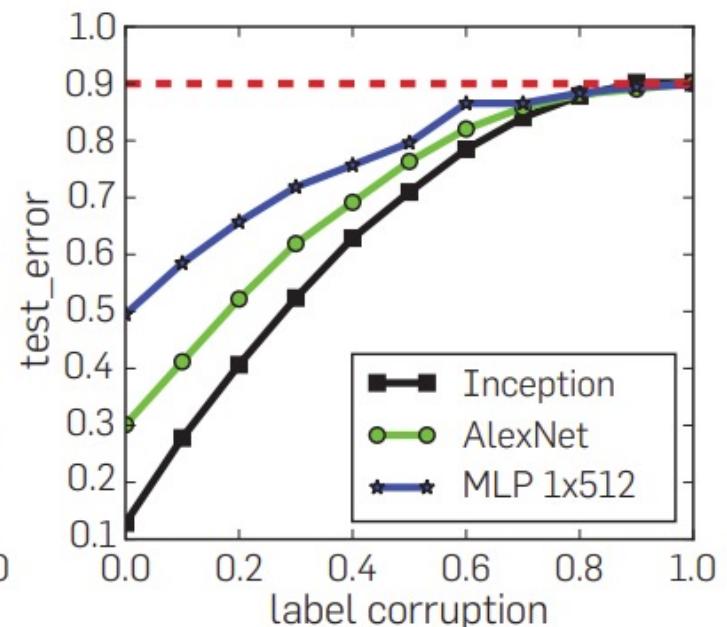
- Fitting random labels and random pixels on CIFAR10.
- (a) The training loss of various experiment settings decaying with the training steps.
- (b) The relative convergence time with different label corruption ratio.
- (c) The test error (also the generalization error since training error is 0) under different label corruptions.



(a) Learning curves



(b) Convergence slowdown



(c) Generalization error growth

Findings

For fitting random labels:

- No need to change the learning rate schedule.
- Once the fitting starts, it converges quickly.
- It converges to (over)fit the training set perfectly. Also note that “random pixels” and “Gaussian” start converging faster than “random labels.”

Partially corrupted labels:

- The behavior of neural network training with a varying level of label corruptions from 0 (no corruption) to 1 (complete random labels) on the CIFAR10 dataset.
- The networks fit the corrupted training set perfectly for all the cases.



Randomization on CIFAR10

- The training and test accuracy (in %) of various models on the CIFAR10 dataset.
- No hyperparameter tuning when switching from the true labels to the random labels.
- With some modification of the hyperparameters, perfect accuracy could be achieved on random labels.



Model	# params	Random crop	Weight decay	Train accuracy	Test accuracy
Inception (fitting random labels)	1,649,402	Yes	Yes	100.0	89.05
		Yes	No	100.0	89.31
		No	Yes	100.0	86.03
		No	No	100.0	85.75
		No	No	100.0	9.78
Inception w/o BatchNorm (fitting ran- dom labels)	1,649,402	No	Yes	100.0	83.00
		No	No	100.0	82.00
		No	No	100.0	10.12
Alexnet (fitting ran- dom labels)	1,387,786	Yes	Yes	99.90	81.22
		Yes	No	99.82	79.66
		No	Yes	100.0	77.36
		No	No	100.0	76.07
		No	No	99.82	9.86
MLP 3 × 512 (fitting ran- dom labels)	1,735,178	No	Yes	100.0	53.35
		No	No	100.0	52.39
		No	No	100.0	10.48
MLP 1 × 512 (fitting ran- dom labels)	1,209,866	No	Yes	99.80	50.39
		No	No	100.0	50.51
		No	No	99.34	10.61

Randomization on ImageNet

- The performance of the Inception v3 model on ImageNet with true labels and random labels, respectively.
- No hyperparameter tuning when switching from the true labels to the random labels.
- With some modification of the hyperparameters, perfect accuracy could be achieved on random labels.

data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

Complexity and Stability

Rademacher Complexity

- Imagine where we replace true labels with Rademacher random variables:
- $\sigma_i = \begin{cases} +1 & \text{with probability } 0.5 \\ -1 & \text{with probability } 0.5 \end{cases}$
- This gives us the Rademacher correlation which shows what is the best that a random classifier could do.

$$\hat{R}_n(F) = E_\sigma[\sup_{f \in F} \frac{1}{n} \sum_{i=1}^n \sigma_i f(z_i)]$$

- Where F is the function class on a dataset $\{z_1, \dots, z_n\}$.
- It can be shown that:

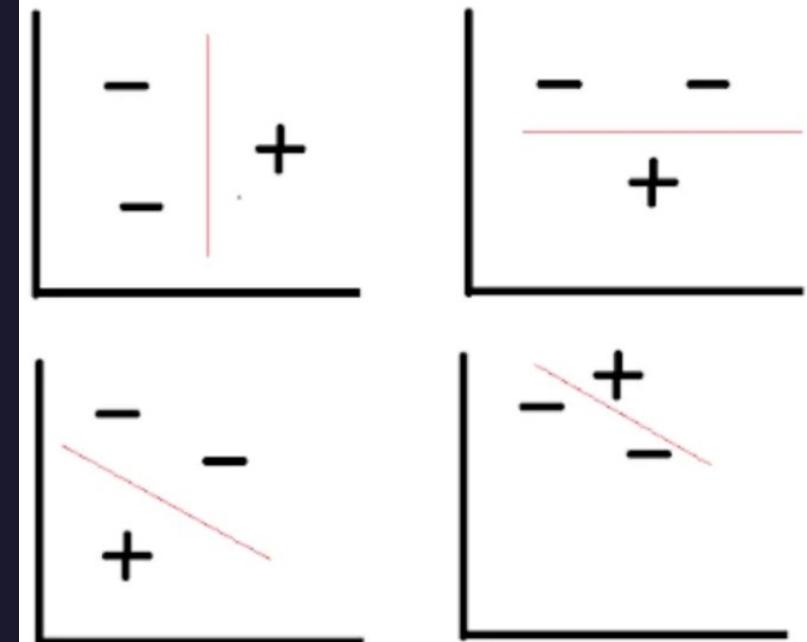
$$E_{test} \leq E_{train} + \hat{R}_n(F) + \sqrt{\frac{\ln(\frac{1}{\eta})}{n}}$$



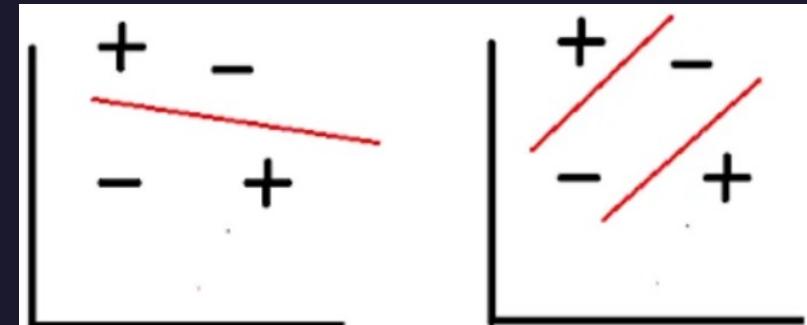
VC-Dimension

- A classification model f with some parameter vector θ is said to shatter a set of data points (x_1, x_2, \dots, x_n) if, for all assignments of labels to those points, there exists a θ such that the model f makes no errors when evaluating that set of data points.
- The VC dimension of a model f is the maximum number of data points that can be arranged so that f shatters them.

3 points:



4 points:



VC-Dimension – Statistical Learning Theory

- Probabilistic upper bound on test error:

$$P(E_{test} \leq E_{train} + \sqrt{\frac{1}{N} \left[D \left(\log \left(\frac{2N}{D} \right) + 1 \right) - \log(\frac{\eta}{4}) \right]}) = 1 - \eta$$

- Validity only when $D \ll N$.



Uniform Stability

- Stability is the property of the algorithm used for training.
- Uniform stability of an algorithm A measures how sensitive the algorithm is to the replacement of a single example.
- If an algorithm does not rely too heavily on any data point, then it generalizes well.
- The weakest stability measure is directly equivalent to bounding generalization error and does take the data into account.



Uniform Stability

- Input: A training set $S = \{z_1, \dots, z_n\}$

- Generalization gap:

$$\varepsilon_{gen} = |E_{test} - E_{train}|$$

- Good learning algorithm minimize both generalization gap and test error.

- Algorithm $A, A(S) = \text{output on training set } S$.

- A is ε -stable if:

$$E|l(A(S); z) - l(A(S \setminus z_i); z)| \leq \varepsilon$$

$$|E_{test} - E_{train}| \leq \varepsilon$$



Conclusions and Implications

- Conclusion:
 - Deep neural networks **easily fit random** labels.
- Implications:
 - The **effective capacity** of neural networks is **sufficient for memorizing** the entire data set.
 - Even **optimization** on random labels **remains easy**.

The role of Regularization



Regularization

Explicit
Regularization

- Data Augmentation
- Weight Decay
- Dropout

Implicit
Regularization

- Early stopping
- Batch Normalization
- SGD

L2 Regularization – “Weight Decay”

- Standard weight update:

$$W_{t+1} \leftarrow W_t - \eta_t \frac{\partial L(W)}{\partial W_t}$$

- $L(W) = l_0(W) + \frac{\lambda}{2} W^2$

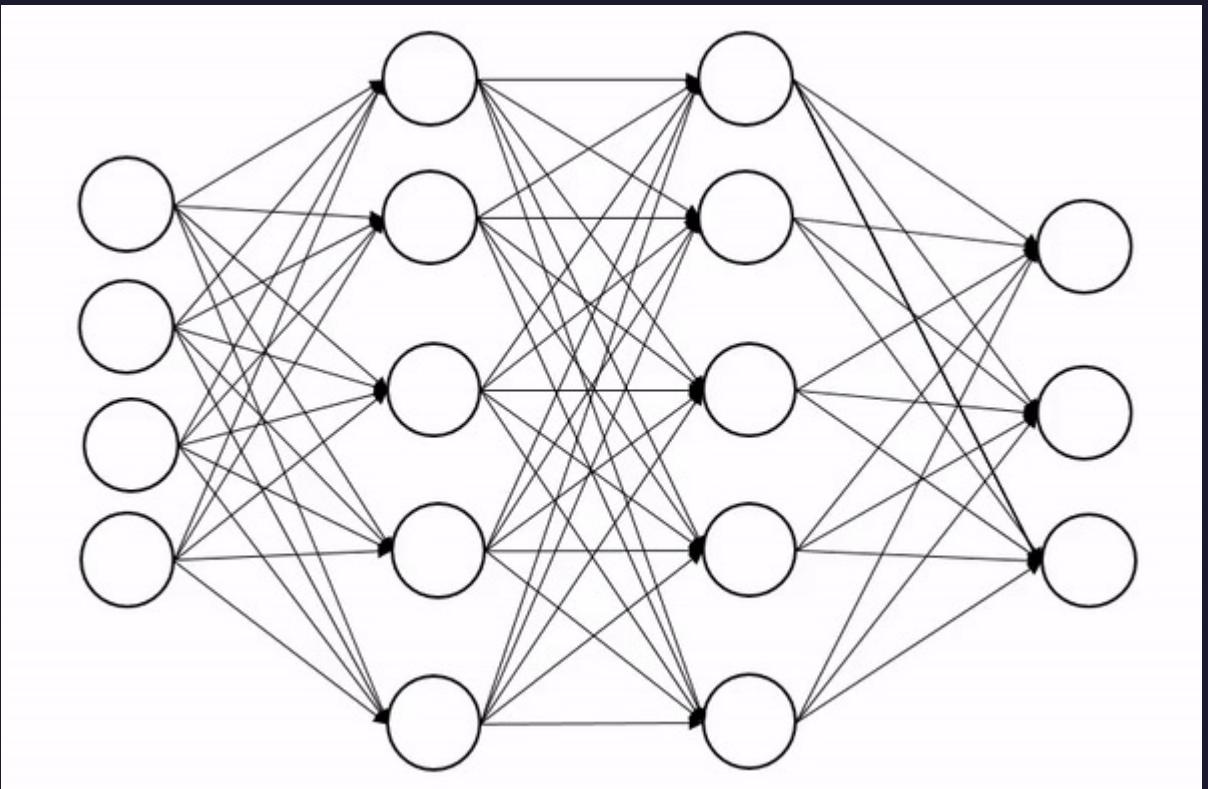
- New weight update:

$$W_{t+1} \leftarrow W_t - \eta_t \frac{\partial L(W)}{\partial W_t} - \eta_t \lambda W_t$$

- Forces the weights to become small, “decay”.
- `optimizer = torch.optim.Adam(params, lr=3e-4, weight_decay=1e-3)`

Dropout

- Randomly drop neurons from layers in the network.
- Removes reliance on individual neurons.
- Maybe doesn't learn redundancies.
- Maybe learns a more nuanced set of feature detectors.
- Dropout can be used after any non-output layer
- `self.dropout = nn.Dropout(0.25)`
- Only the InceptionV3 for ImageNet uses dropout in the experiments.



Data Augmentation

- Domain-specific transformations of the input data.
- For image data, commonly used transformations include random cropping, and random perturbation of brightness, saturation, hue, and contrast.
- Increases the input space (i.e. all possible images we care about).



Experimental findings



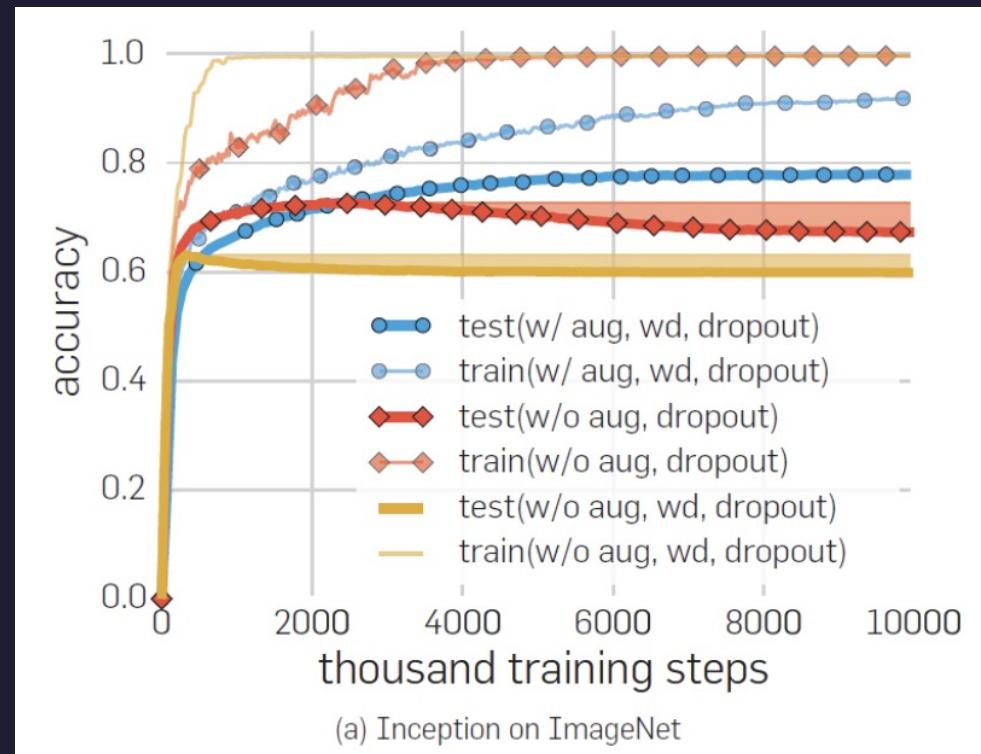
Explicit Regularization Results

Model	# params	Random crop	Weight decay	Train accuracy	Test accuracy
Inception	1,649,402	Yes	Yes	100.0	89.05
		Yes	No	100.0	89.31
		No	Yes	100.0	86.03
		No	No	100.0	85.75
(fitting random labels)		No	No	100.0	9.78
Inception w/o BatchNorm	1,649,402	No	Yes	100.0	83.00
		No	No	100.0	82.00
(fitting random labels)		No	No	100.0	10.12
Alexnet	1,387,786	Yes	Yes	99.90	81.22
		Yes	No	99.82	79.66
		No	Yes	100.0	77.36
		No	No	100.0	76.07
(fitting random labels)		No	No	99.82	9.86
MLP 3 × 512	1,735,178	No	Yes	100.0	53.35
		No	No	100.0	52.39
(fitting random labels)		No	No	100.0	10.48
MLP 1 × 512	1,209,866	No	Yes	99.80	50.39
		No	No	100.0	50.51
(fitting random labels)		No	No	99.34	10.61

Explicit Regularization Results

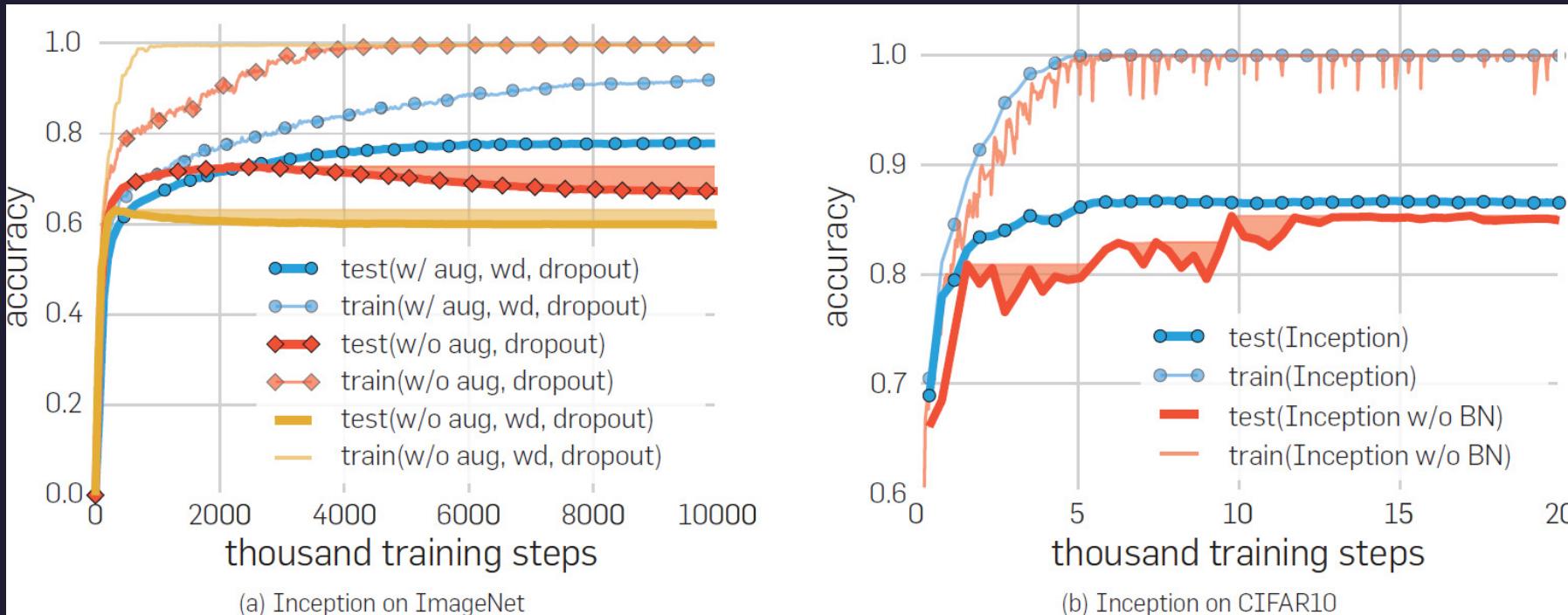
data aug	dropout	weight decay	top-1 train	top-5 train	top-1 test	top-5 test
ImageNet 1000 classes with the original labels						
yes	yes	yes	92.18	99.21	77.84	93.92
yes	no	no	92.33	99.17	72.95	90.43
no	no	yes	90.60	100.0	67.18 (72.57)	86.44 (91.31)
no	no	no	99.53	100.0	59.80 (63.16)	80.38 (84.49)
Alexnet (Krizhevsky et al., 2012)			-	-	-	83.6
ImageNet 1000 classes with random labels						
no	yes	yes	91.18	97.95	0.09	0.49
no	no	yes	87.81	96.15	0.12	0.50
no	no	no	95.20	99.14	0.11	0.56

Explicit Regularization Results



- Explicit regularization may improve generalization but is neither necessary nor sufficient itself

Implicit Regularization Results



- Early stopping could potentially improve generalization
- Batch normalization doesn't help much with generalization performance

Appeal to Linear Models

- Appealing to linear models, they analyze how SGD acts as an implicit regularizer.
- For linear models, SGD always converges to a solution with a small ℓ_2 -norm.
- Hence, the algorithm itself is implicitly regularizing the solution.
- Minimum norm is not predictive of generalization performance.



Finite Sample Expressivity



Finite Sample Expressivity of Neural Networks

- When we talk about population level it means that with infinite number of inputs how the network performs.
- At the “population level”, depth k networks typically more powerful than depth $k-1$ networks.
- Given a finite sample size n , even a two-layer neural network can represent any function once the number of parameters p exceeds n .
- Theorem I:

“There exists a two-layer neural network with ReLU activations and $2n+d$ weights that can represent any function on a sample of size n in d dimensions.”



Finite Sample Expressivity of Neural Networks

- A network C can represent any function of a sample size n in d dimensions if:
 - For every samples $S \subseteq \mathbb{R}^d$ with $|S| = n$ and
 - Every function $f: S \rightarrow \mathbb{R}$.
- There exists a setting of weights of C such that
$$C(x) = f(x) \text{ for every } x \in S.$$



Conclusion

- Effective capacity of neural networks.
- Successful neural networks are large enough to shatter the training data.
- Optimization continues to be easy even when generalization is poor.
- SDG may be performing implicit regularization by converging to solutions with minimum l_2 -norm.
- Traditional measures of model complexity struggle to explain the generalization of large neural networks.
- They do not present any better measures, but they just show that the traditional measures can not be used to explain generalization.

Related Work

- Conventional generalization bounds based on uniform stability is inadequate for overparameterized deep neural networks, extensive efforts were made toward tighter generalization bounds.
- Based on overparameterized deep networks generalize even without any explicit regularization, and the analysis of implicit regularization in linear models, there is renewed interest in seeking to explain generalization in deep learning by characterizing the implicit regularization induced by the learning algorithms.
- In-depth analysis on memorization of overparameterized models also extends intuition on overfitting from the traditional U-shaped risk curve to the “double descent” risk curve.
- The randomization test proposed in this paper serves as the backbone in the experimental design in many of studies. Dedicated workshops on phenomena in deep learning are being organized in all major machine learning conferences nowadays. Even some theory conferences start to consider pure empirical studies that reveal “interesting and not well understood behavior” in the call-for-papers.

References



References:

- Zhang, Chiyuan, et al. "Understanding deep learning (still) requires rethinking generalization." *Communications of the ACM* 64.3 (2021): 107-115.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O. Understanding deep learning requires rethinking .
- <https://www.youtube.com/watch?v=O42vde4tbG0>.
- <https://www.youtube.com/watch?v=mEYerIMYb5Q>.
- Cybenko, George. "Approximation by superpositions of a sigmoidal function." *Mathematics of control, signals and systems* 2.4 (1989): 303-314.
- Krogh, Anders, and John A. Hertz. "A simple weight decay can improve generalization.“ In *advances in neural information processing systems*, pp. 950-957. 1992.
- Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958.
- Luke Taylor, Geoff Nitschke.“Improving Deep Learning using Generic Data Augmentation.” arXiv preprint. arXiv: 1708.06020 (2017).
- <https://www.youtube.com/watch?v=47dPjn2Rfr8>.
- <https://www.youtube.com/watch?v=puDzy2XmR5c>.
- <https://www.youtube.com/watch?v=yXOMHOpb0n8>

Thank You

