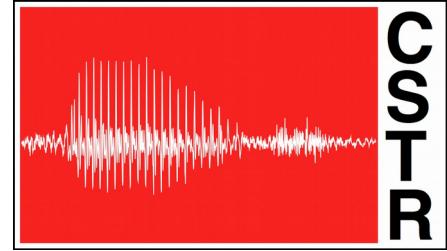




THE UNIVERSITY of EDINBURGH
informatics



Understanding Deep Learning Requires Rethinking Generalisation

Erfan Loweimi

Centre for Speech Technology Research (CSTR)
Listen! 12.3.2019



ICLR

Published as a conference paper at ICLR 2017

UNDERSTANDING DEEP LEARNING REQUIRES RE-THINKING GENERALIZATION

Chiyuan Zhang*

Massachusetts Institute of Technology
chiyuan@mit.edu

Samy Bengio

Google Brain
bengio@google.com

Moritz Hardt

Google Brain
mrtz@google.com

Benjamin Recht[†]

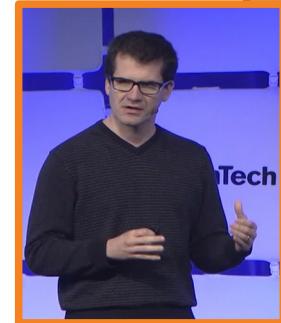
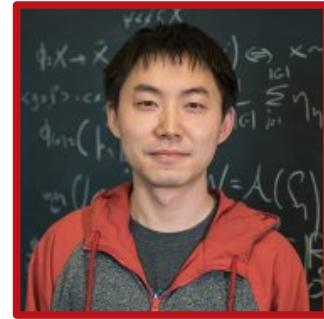
University of California, Berkeley
brecht@berkeley.edu

Oriol Vinyals

Google DeepMind
vinyals@google.com

ABSTRACT

Despite their massive size, successful deep artificial neural networks can exhibit a remarkably small difference between training and test performance. Conventional wisdom attributes small generalization error either to properties of the model family, or to the regularization techniques used during training.



Important Paper ...



iclr2017 @iclr2017 · 23 Feb 2017
Congrats "Understanding Deep Learning Requires Rethinking Generalization" for #ICLR best paper award! Schedule: iclr.cc/doku.php?id=ic...

2 105 278

iclr2017 @iclr2017 · 23 Feb 2017
Congrats "Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data" for #ICLR best paper award!

1 44 138

iclr2017 @iclr2017 · 23 Feb 2017
Congrats "Making Neural Programming Architectures Generalize via Recursion" for #ICLR best paper award! Schedule: iclr.cc/doku.php?id=ic...

27 74



ICLR 2017
Best Paper
Award

Understanding deep learning requires rethinking generalization

<https://arxiv.org/> > cs ▾

11/3/2019 →

by C Zhang - 2016 - **Cited by 765** | Related articles

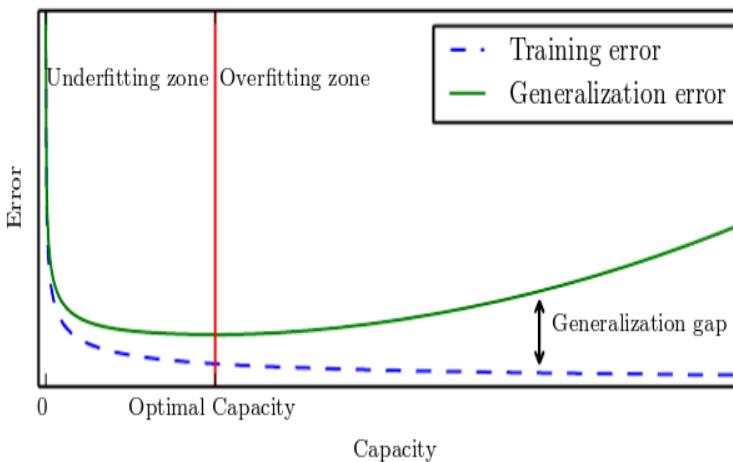
10 Nov 2016 - **Understanding deep learning requires rethinking generalization.** Despite their massive size, successful deep artificial neural networks can exhibit a remarkably small difference between training and test performance.

Highly cited

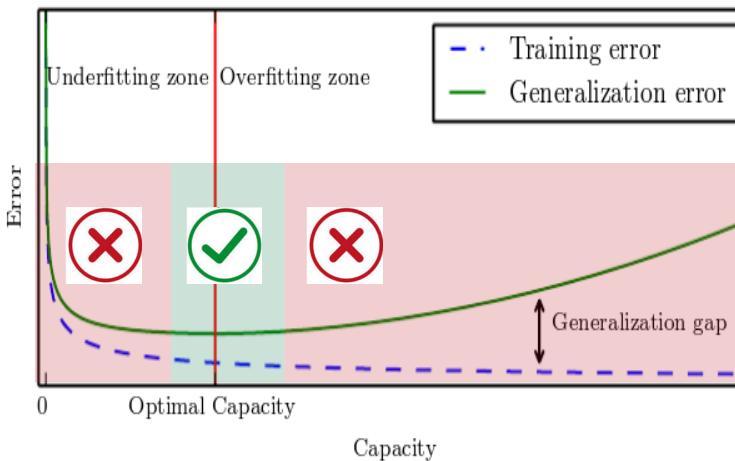
- Why DNNs work/generalise well
- Interpretability/understanding



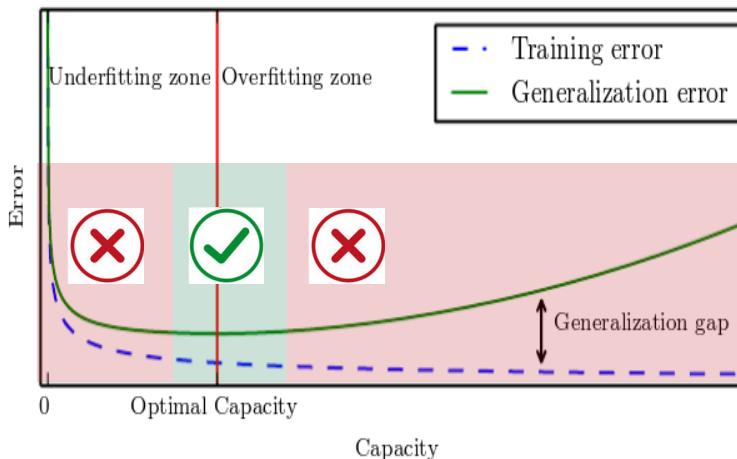
- Why DNNs work/generalise well
- Interpretability/understanding



- Why DNNs work/generalise well
- Interpretability/understanding



- Why DNNs work/generalise well
- Interpretability/understanding



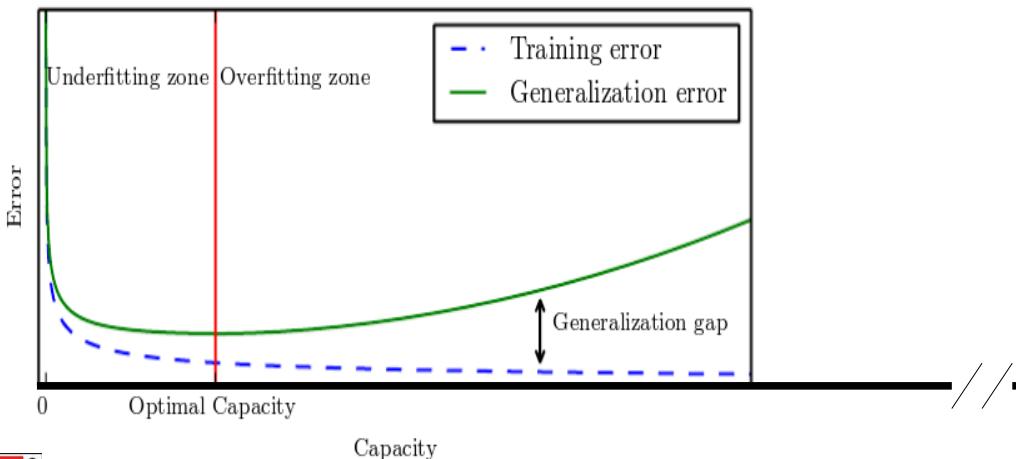
Underfitting: High Bias



Overfitting: High Variance



- Why DNNs work/generalise well

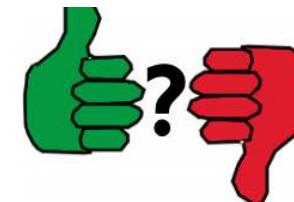


Over-parametrised Zone





Over-parametrisation is Good or Bad?

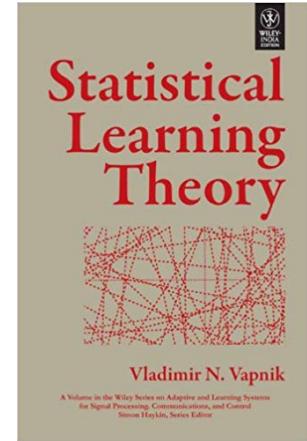


E. Loweimi



Generalisation Error

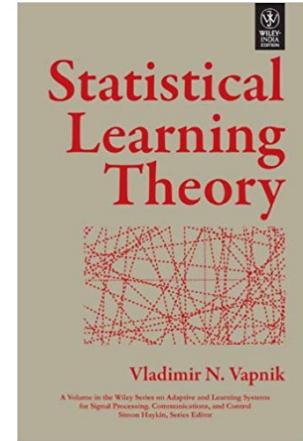
- Definition: $E_{\text{gen}} = E_{\text{test}} - E_{\text{train}}$
- Statistical Learning Theory
 - Complexity measures \leftrightarrow upper bound for E_{gen}



$$E_{\text{gen}} = E_{\text{test}} - E_{\text{train}} \underset{\leq}{\propto} \frac{f_1(\#\text{parameters})}{f_2(N)}$$

Generalisation Error

- Definition: $E_{\text{gen}} = E_{\text{test}} - E_{\text{train}}$
- Statistical Learning Theory
 - Complexity measures \leftrightarrow upper bound for E_{gen}
 - Examples: VC-dim, Rademacher, Uniform stability



$$E_{\text{gen}} = E_{\text{test}} - E_{\text{train}} \underset{\propto}{\leq} \frac{f_1(\#\text{parameters})}{f_2(N)} \text{ e.g. } f\left(\frac{\text{VC-dim}}{N}\right)$$

Over-parametrisation is Good or Bad?



IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 12, NO. 12, DECEMBER 1990

Links Between Markov Models and Multilayer Perceptrons

Herve Bourlard, *Member, IEEE*, and Christian J. Wellekens, *Senior Member, IEEE*

TABLE I
CLASSIFICATION RATES AT THE FRAME LEVEL ON SPICOS DATABASE

# Hidden Units	# Parameters/ # Training Patterns	Training	Test
5	0.23	62.8 (1.010)	54.2 (1.012)
20	0.93	75.7 (1.030)	62.7 (1.035)
50	2.31	86.4 (1.018)	61.4 (1.000)
200	9.3	86.9 (1.053)	59.4 (0.995)
MLE	0.25	45.9	44.8
MAP	0.25	53.8	53.0
50 NCI	0.34	53.5 (1.011)	52.7 (1.012)

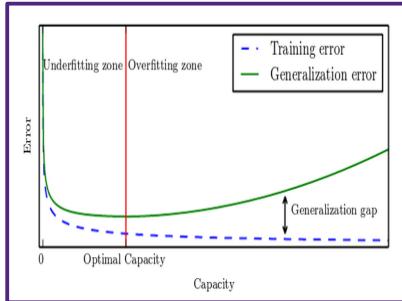
Over-parametrisation is Good or **Bad?**



IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 12, NO. 12, DECEMBER 1990

Links Between Markov Models and Multilayer Perceptrons

Herve Bourlard, *Member, IEEE*, and Christian J. Wellekens, *Senior Member, IEEE*



1. Over-parametrisation increases test error.
2. Results match traditional view.

TABLE I
CLASSIFICATION RATES AT THE FRAME LEVEL ON SPICOS DATABASE

# Hidden Units	# Parameters/ # Training Patterns	Training	Test
5	0.23	62.8 (1.010)	54.2 (1.012)
20	0.93	75.7 (1.030)	62.7 (1.035)
50	2.31	86.4 (1.018)	61.4 (1.000)
200	9.3	86.9 (1.053)	59.4 (0.995)
MLE	0.25	45.9	44.8
MAP	0.25	53.8	53.0
50 NCI	0.34	53.5 (1.011)	52.7 (1.012)



Size of the Model & Databases

CIFAR-10	#train: 50,000
Inception	1,649,402
AlexNet	1,387,786
MLP 1x512	1,209,866
ImageNet	#train: 1,200,000
Inception V3	23,885,392
AlexNet	61,100,840
ResNet-{18; 152}	11,689,512; 60,192,808
VGG-{11;19}	132,863,336; 143,667,240

Size of the Model & Databases

CIFAR-10	#train: 50,000	#parameter/#train
Inception	1,649,402	33
AlexNet	1,387,786	28
MLP 1x512	1,209,866	24
ImageNet	#train: 1,200,000	
Inception V3	23,885,392	20
AlexNet	61,100,840	51
ResNet-{18; 152}	11,689,512; 60,192,808	10; 50
VGG-{11;19}	132,863,336; 143,667,240	110; 120

Size of the Model & Databases

CIFAR-10	#train: 50,000	#parameter/#train
Inception	1,649,402	33
AlexNet	1,387,780	28
MLP 1x512	1,209,866	24
ImageNet	#train: 1,200,000	
Inception V3	23,885,392	20
AlexNet	61,100,840	91
ResNet-{18; 152}	11,689,512; 60,192,808	10; 50
VGG-{11;19}	132,863,336; 143,667,240	110; 120

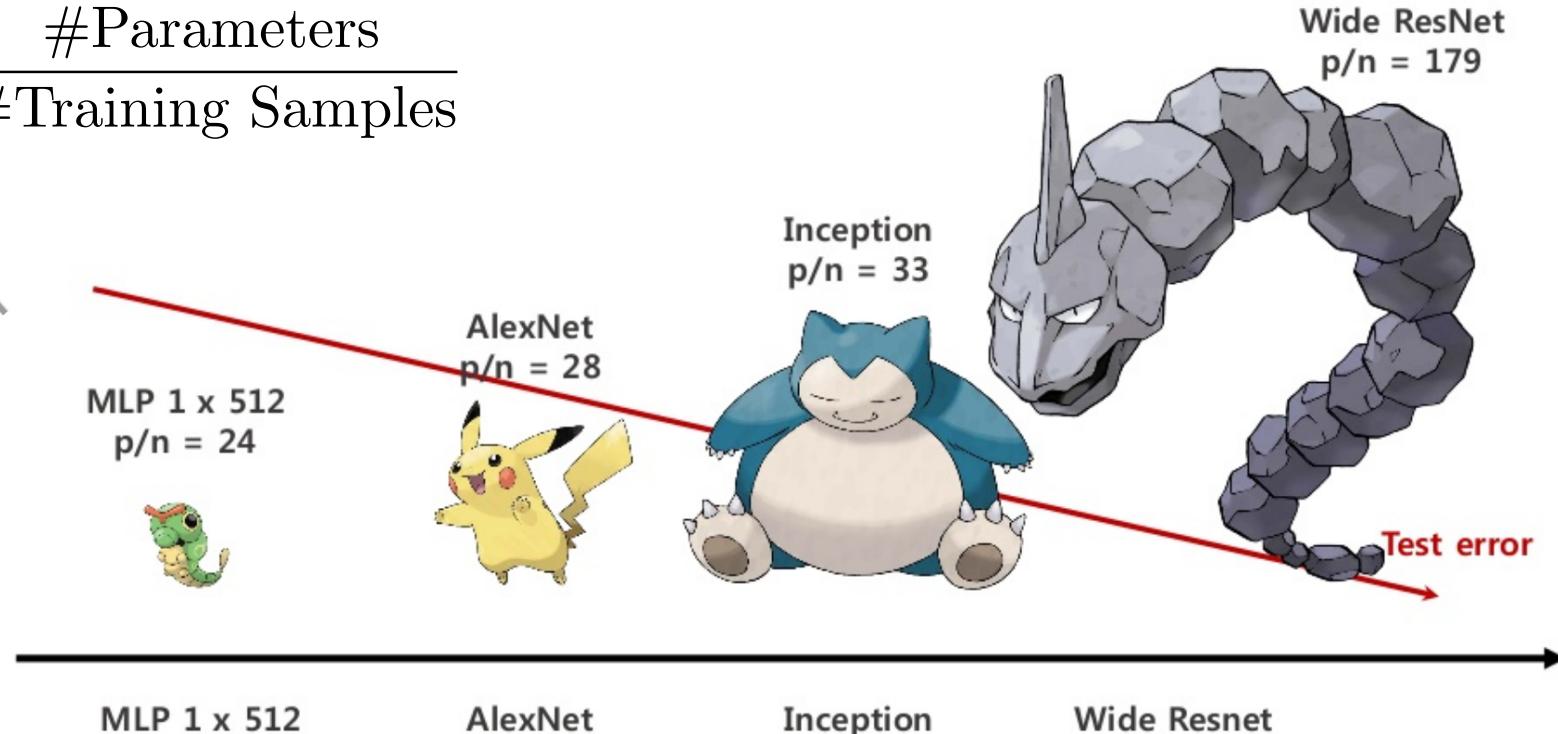
Overparametrised

Over-parametrisation is Good or Bad?



$$p/n = \frac{\text{\#Parameters}}{\text{\#Training Samples}}$$

IM[■]GENET



MLP 1 x 512

AlexNet

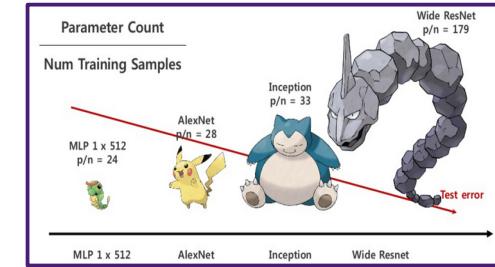
Inception

Wide Resnet

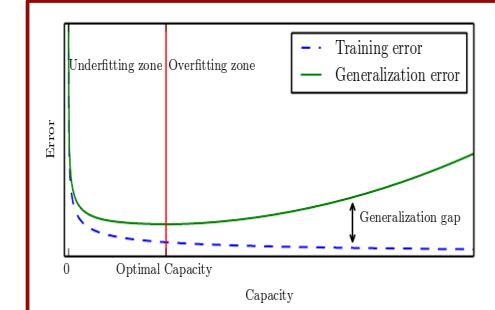
Over-parametrisation is Good or Bad?

- High **#parameters** does **NOT** necessarily ...

- degrade the generalisation performance
 - reflect model complexity



- Traditional views are **NOT** enough for explaining generalisation in the over-parameterised models ($p > n$).

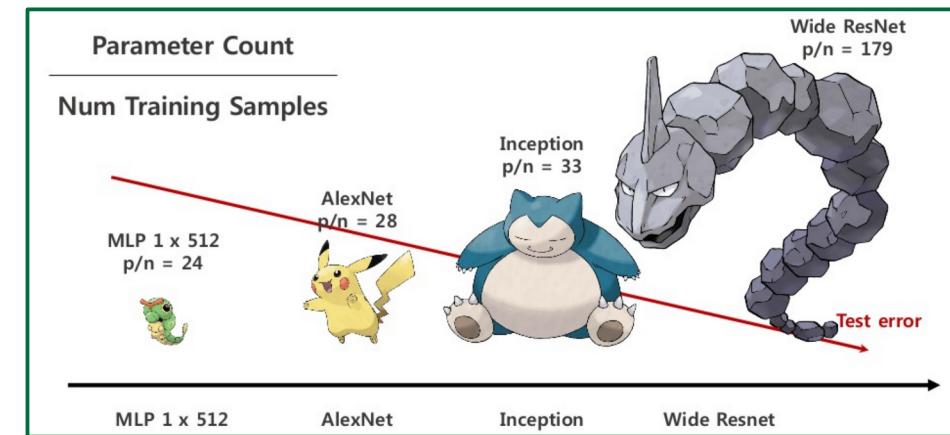


The Puzzle

- Over-parametrisation is good or bad?
 - Both!



TABLE I CLASSIFICATION RATES AT THE FRAME LEVEL ON SPICOS DATABASE			
# Hidden Units	# Parameters/ # Training Patterns	Training	Test
5	0.23	62.8 (1.010)	54.2 (1.012)
20	0.93	75.7 (1.030)	62.7 (1.035)
50	2.31	86.4 (1.018)	61.4 (1.000)
200	9.3	86.9 (1.053)	59.4 (0.995)
MLE	0.25	45.9	44.8
MAP	0.25	53.8	53.0
50 NCI	0.34	53.5 (1.011)	52.7 (1.012)



The Puzzle

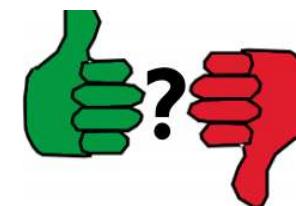


If the map and the terrain disagree, trust the terrain.

Swiss Army Aphorism

The Puzzle

- Over-parametrisation is good or bad?
 - Both!
- “*What is it then that distinguishes NNs that generalise well from those that don’t?*”





Experimental Setup



Experiments – Datasets

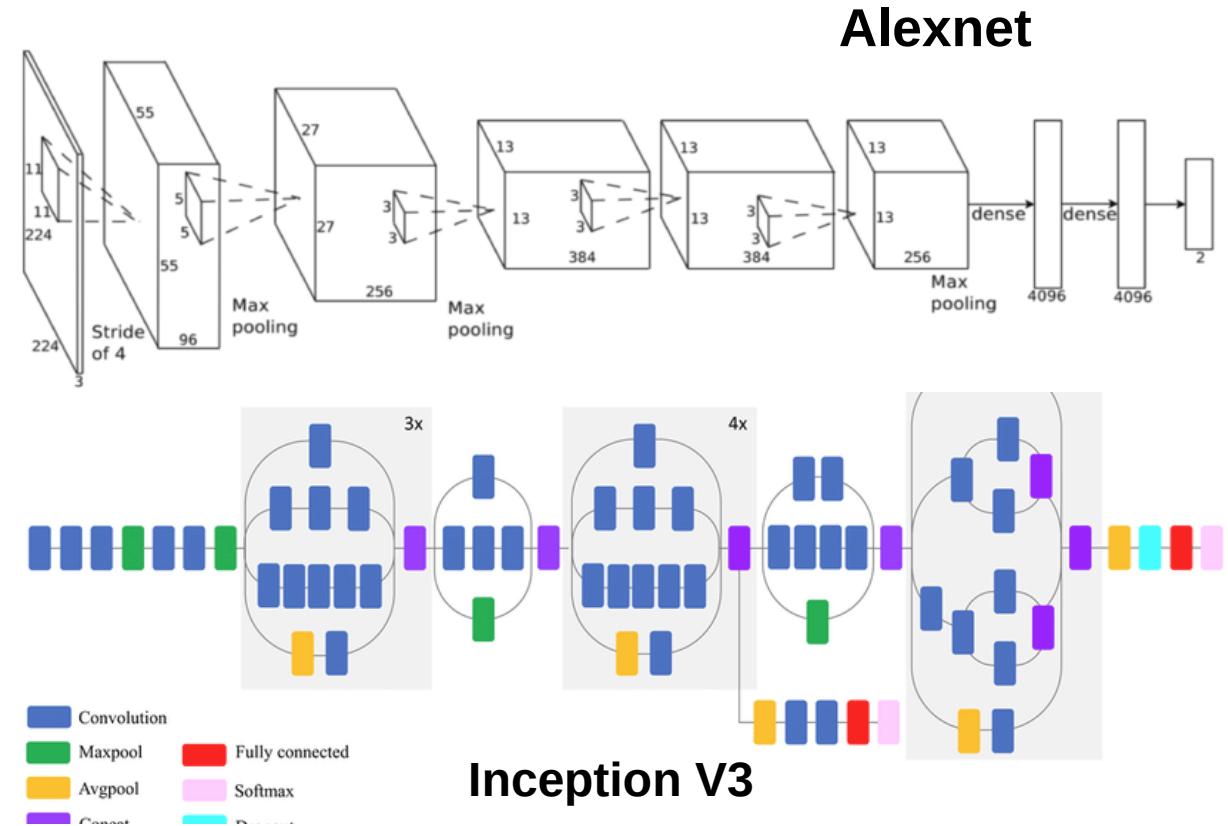
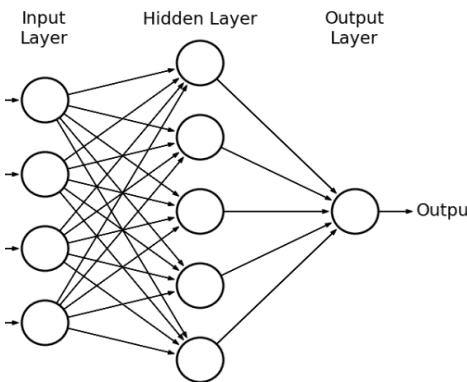
- **CIFAR-10**
 - 10 classes
 - 60k images, 32x32
 - 50k train + 10k test
- **ImageNet (ILSVRC 2012)**
 - 1000 classes
 - 1.35M images, 299x299
 - 1.2 M train + 150k val/test



Architectures

MLP1: 1x512

MLP2: 3x512

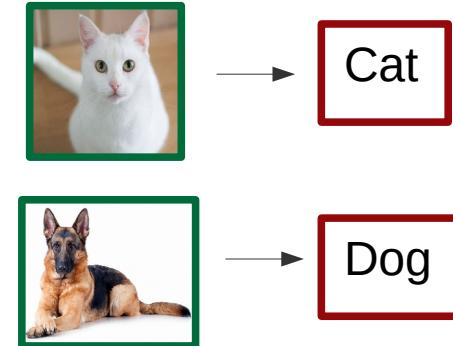


Experiments – Optimisation

- Stochastic Gradient Descend (SGD)
- Momentum: 0.9
- Learning rate:
 - MLP & Alexnet: 0.01, Inception: 0.1
- Learning rate annealing: 0.95 per epoch
- Mini-batch size: 128

Experiments – Randomizations Tests

- Label levels (y)
- Data level (X)

$$X \rightarrow y$$




Randomizations Tests – Labels

- Inspired from Non-parametric statistics



Label Randomisation



Cat



Cat



Dog



Dog



Horse



Flower



Car



Fish

E. Loweimi

Label Randomisation



Cat [1]



Cat [1]



Dog [2]



Dog [2]



Horse [3]



Flower [4]



Car [5]



Fish [6]



E. Loweimi

Label Randomisation



Cat [1]



Cat [1]



Dog [2]



Dog [2]



Horse [3]



Flower [4]



Car [5]



Fish [6]



Uniform sampling
(non-parametric)

E. Loweimi

Label Randomisation



Cat [1]



3 → Horse



Cat [1]



5 → Car



Dog [2]



4 → Flower



Dog [2]



2 → Dog



Horse [3]



3 → Horse



Flower [4]



5 → Car



Car [5]



6 → Fish



Fish [6]



1 → Cat

Label Randomisation



Cat [1]



3 → Horse



Cat [1]



5 → Car



Dog [2]



4 → Flower



Dog [2]



2 → Dog



Horse [3]



3 → Horse



Flower [4]



5 → Car



Car [5]



6 → Fish



Fish [6]



1 → Cat



Randomizations Tests – Labels

- Non-parametric statistics
- Test conditions
 - True labels
 - Random Labels



Randomizations Tests – Labels

- Non-parametric statistics
- Test conditions
 - True labels
 - Random Labels
 - Nothing (mapping/relationship/abstraction) to learn!
 - Does learning/optimisation algorithm converge?

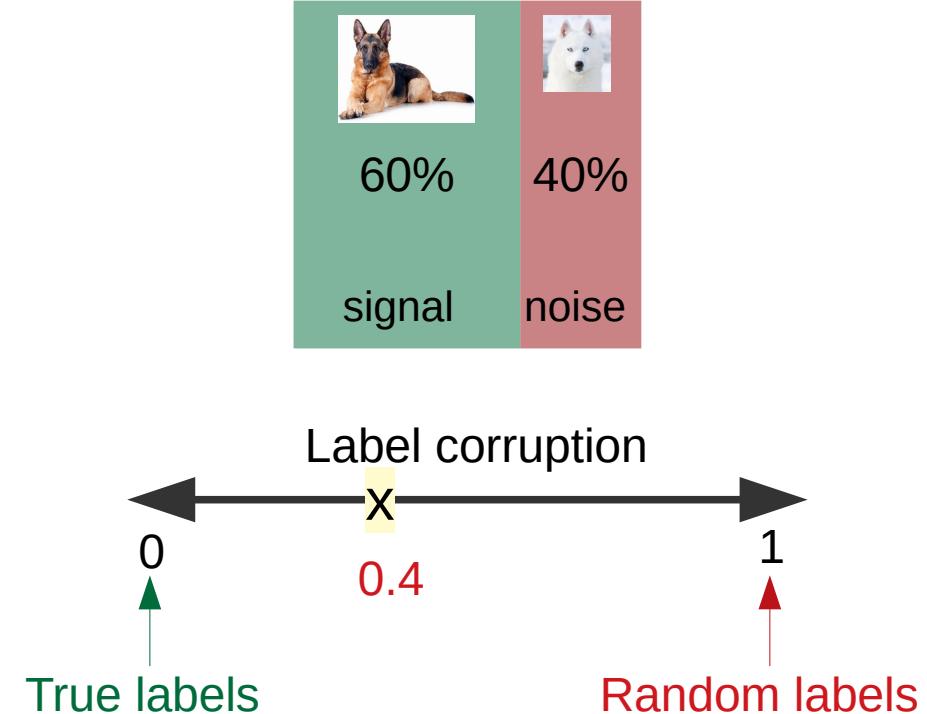
Randomizations Tests – Labels

- Non-parametric statistics
- Test conditions
 - True labels
 - Partially corrupted labels
 - Random Labels



Randomizations Tests – Labels

- Non-parametric statistics
- Test conditions
 - True labels
 - Partially corrupted labels
 - Random Labels





Randomizations Tests – Data

- Shuffled pixels
- Random pixels
- Gaussian



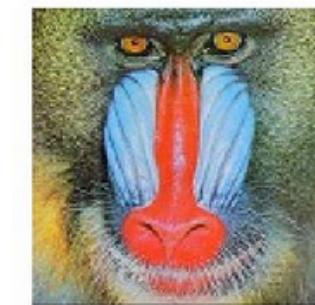
Randomizations Tests – Data

- Shuffled pixels → one permutation trans.
- Random pixels → Many permutation trans.
- Gaussian (parametric) → database mean & variance

1	2
3	4

permutation

3	1
4	2

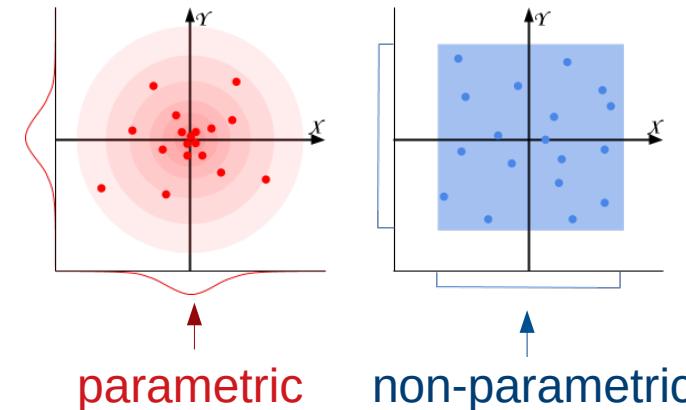


permutation



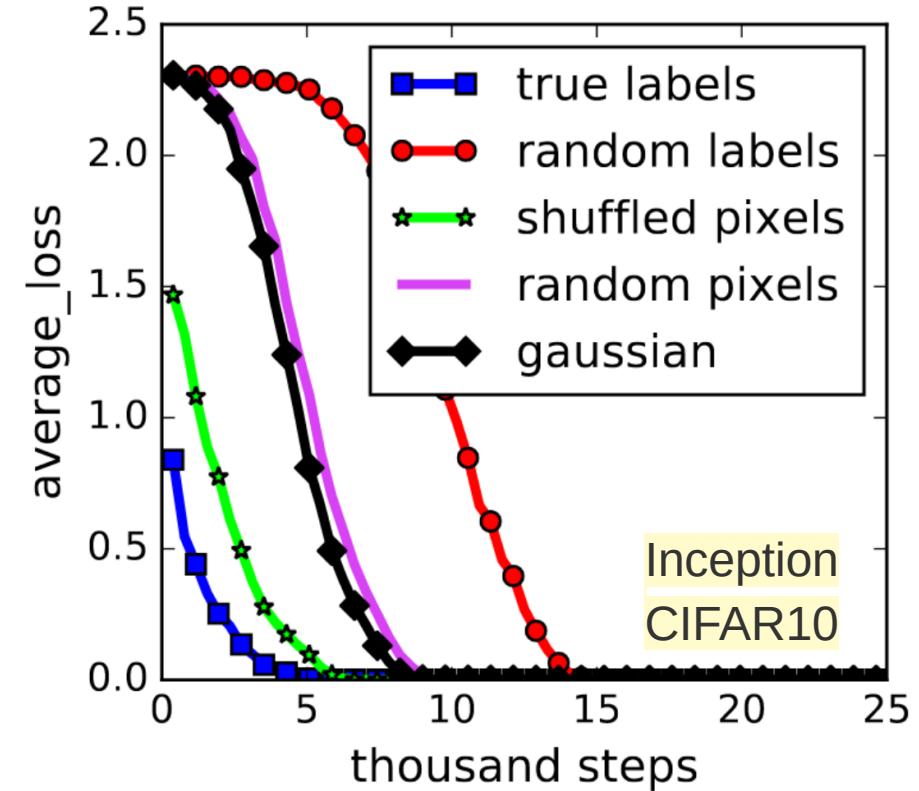
Randomizations Tests – Data

- Shuffled pixels → one permutation trans.
- Random pixels → Many permutation trans.
- **Gaussian (parametric)** → database mean & variance



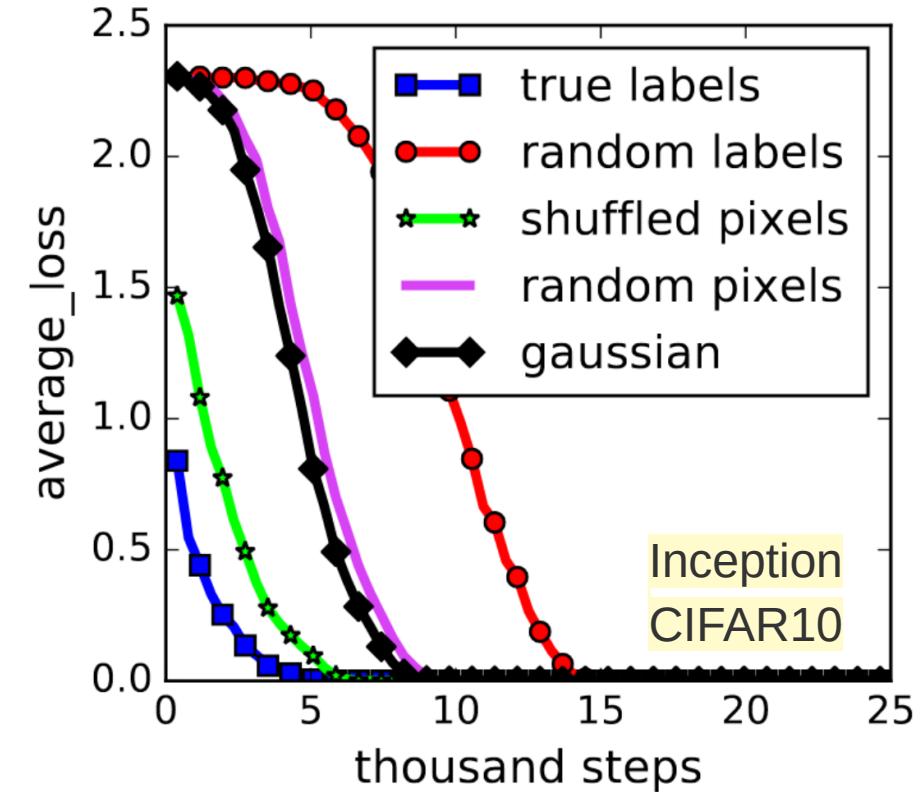
Learning Dynamics – Training Data

NOTE:
Hyperparameters are identical



Learning Dynamics – Training Data

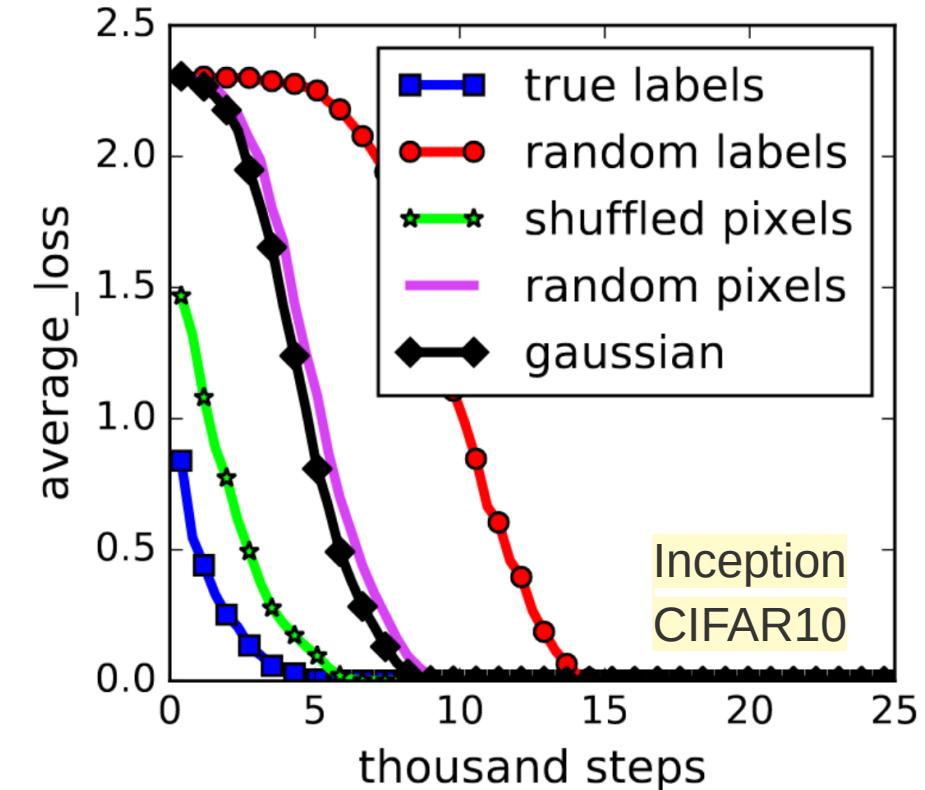
DNN shatters training data,
even with random data/labels.



Learning Dynamics – Training Data

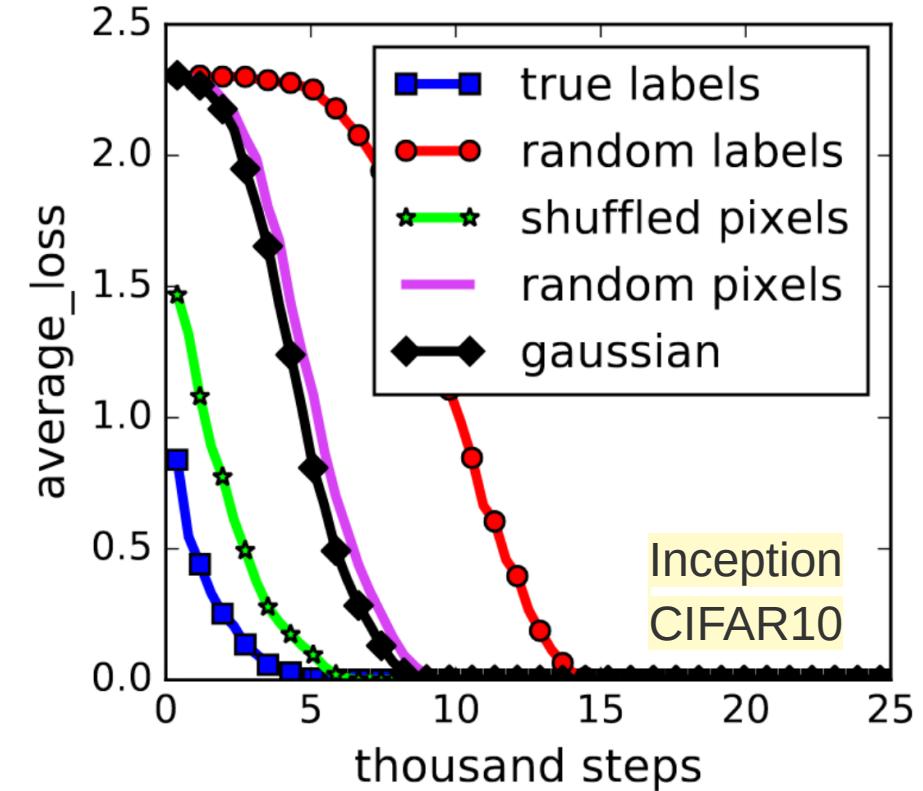
DNN shatters training data,
even with **random data/labels**.

This is **memorisation** not learning.



Learning Dynamics – Training Data

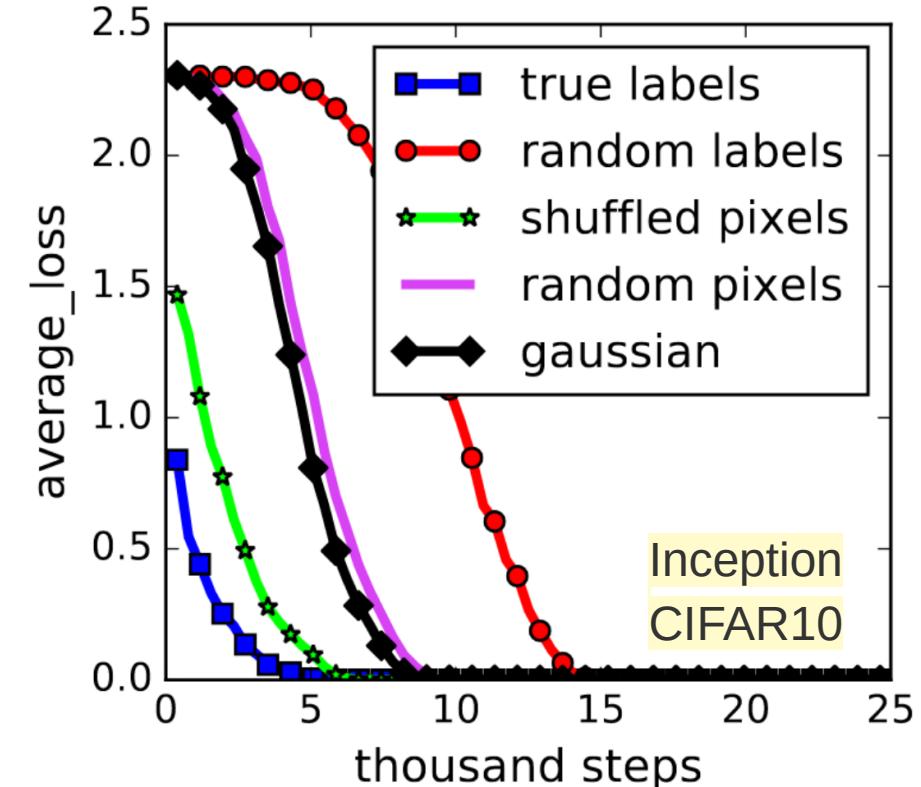
Optimisation remains easy, even when learning is hard → Although slower, it converges
– Optimisation-Learning relationship?



Learning Dynamics – Training Data

Convergence for data randomisation happens earlier than label randomisation. Y?

- better separation (paper)
- randomisation level (I think)

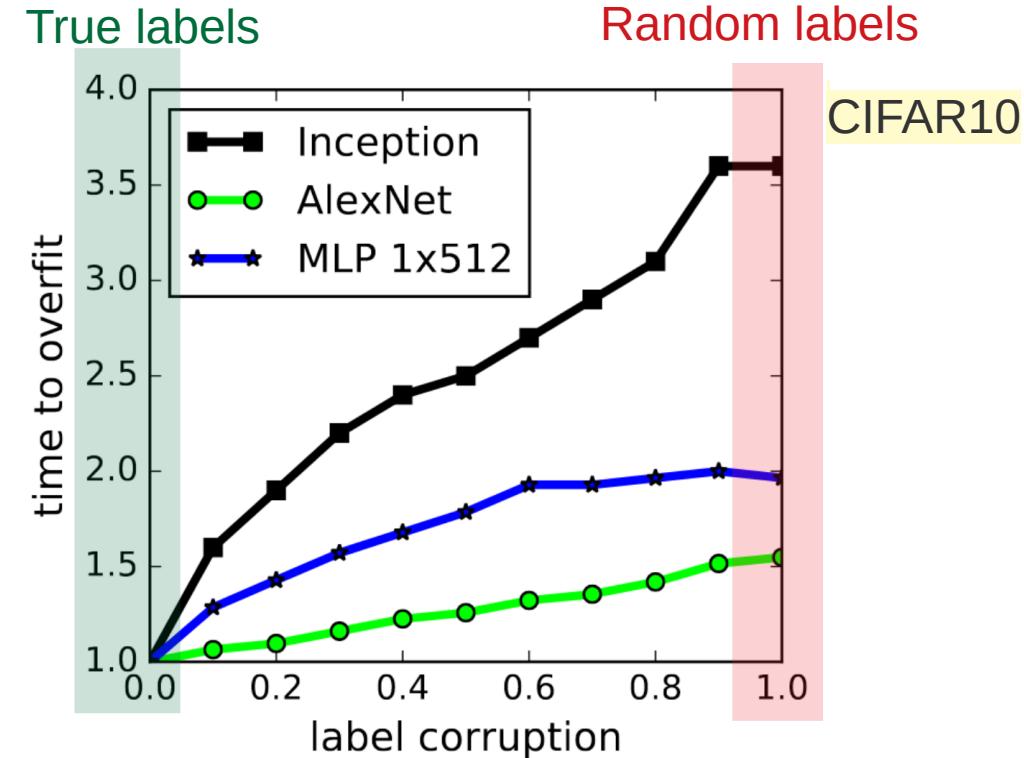


Convergence Slowdown after Label Corruption

Label corruption

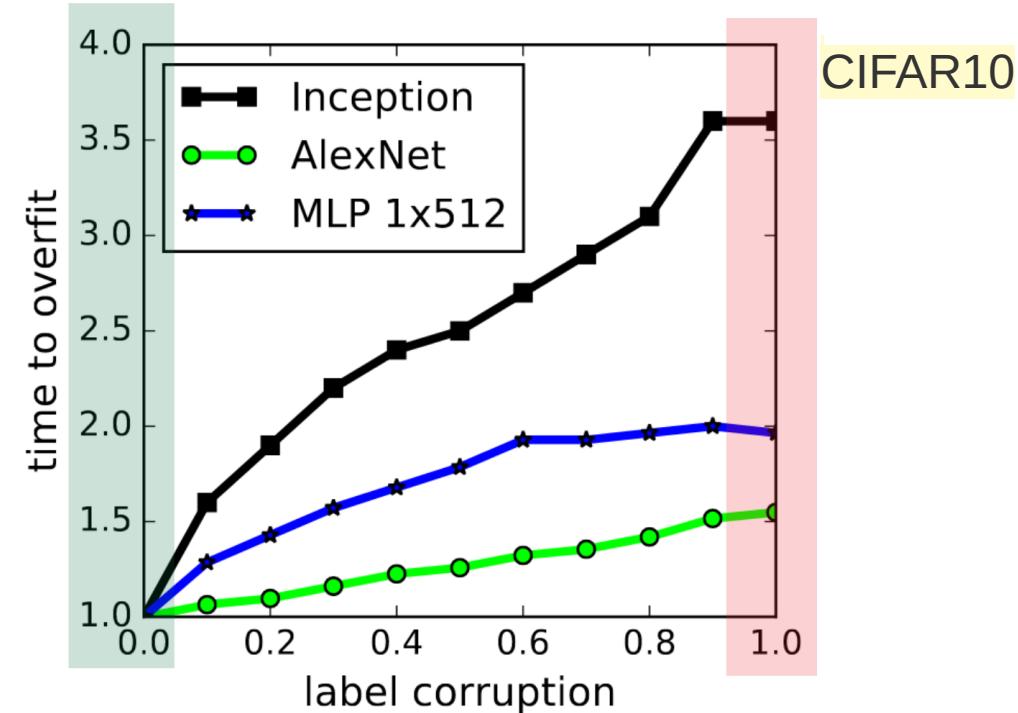


slower convergence



Convergence Slowdown for Different Label Corruption Rations

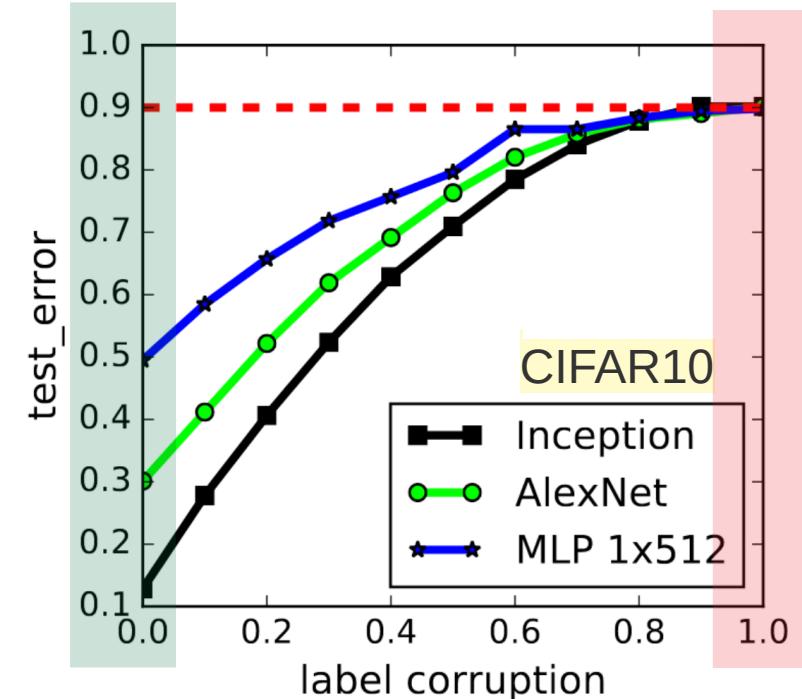
Slower convergence is proportional with **#parameters**, although **architecture** matters, too.



Generalisation Error Growth

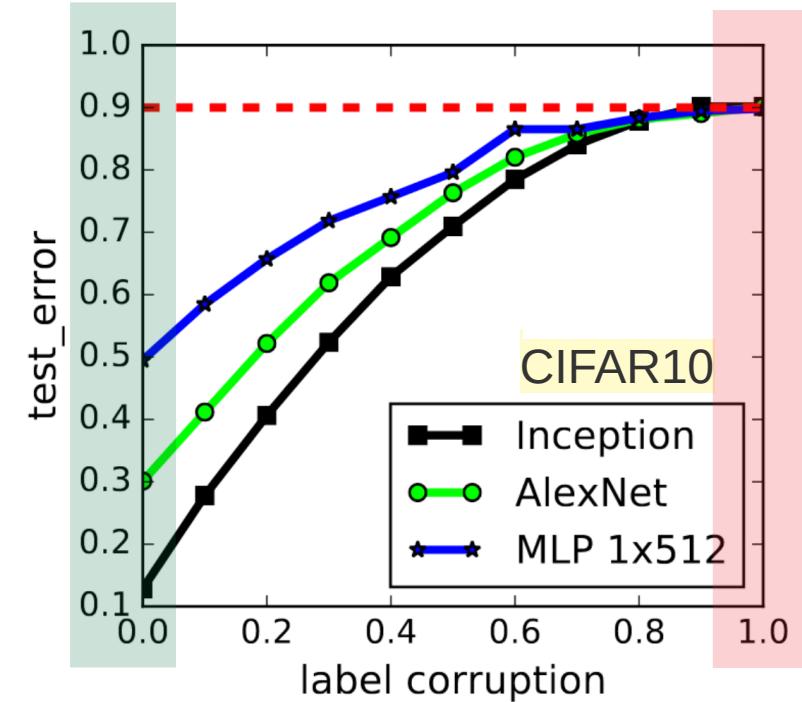
Architectures matters a lot!

- Generalisation performance
Inception > Alexnet > MLP



New Complexity Measures Required!

Steady increase in generalisation error by label corruption.

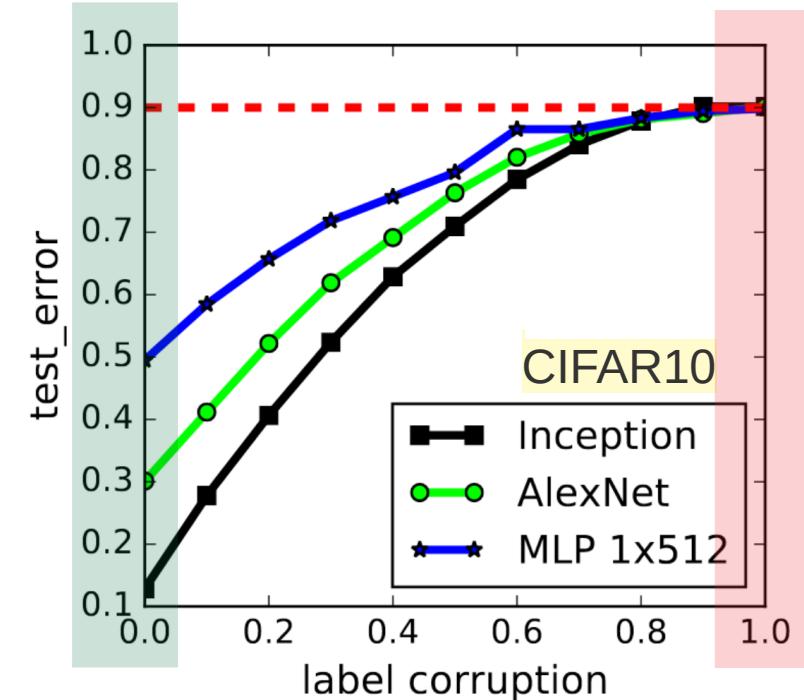


New Complexity Measures Required!

Steady increase in generalisation error by label corruption.

Model capacity & Δ are **NOT** changed **BUT** generalisation error dramatically changed!

$$P_{gen} \Big|_{E_{train}=0} \leq O\left(\frac{VCdim}{N}\right)$$





Regularisation



Regularisation

- Definition: A mechanism that constraints the model (hypothesis space) or empowers the data
- GOAL: Avoids fitting noise
- Effect: **decreases variance**, may **increase the bias**
- Types:
 - Explicit: Data augmentation, weight decay, drop out
 - Implicit: Early stop, batch normalisation

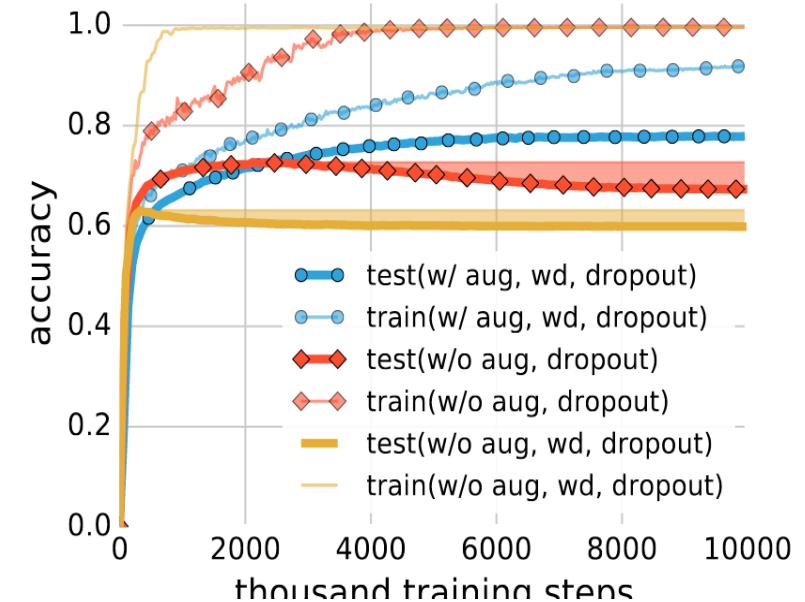
Explicit Regularisation Effect

Dropout

Dropout+WD

Dropout+WD+DAug

ImageNet – Inception V3



Explicit Regularisation Effect

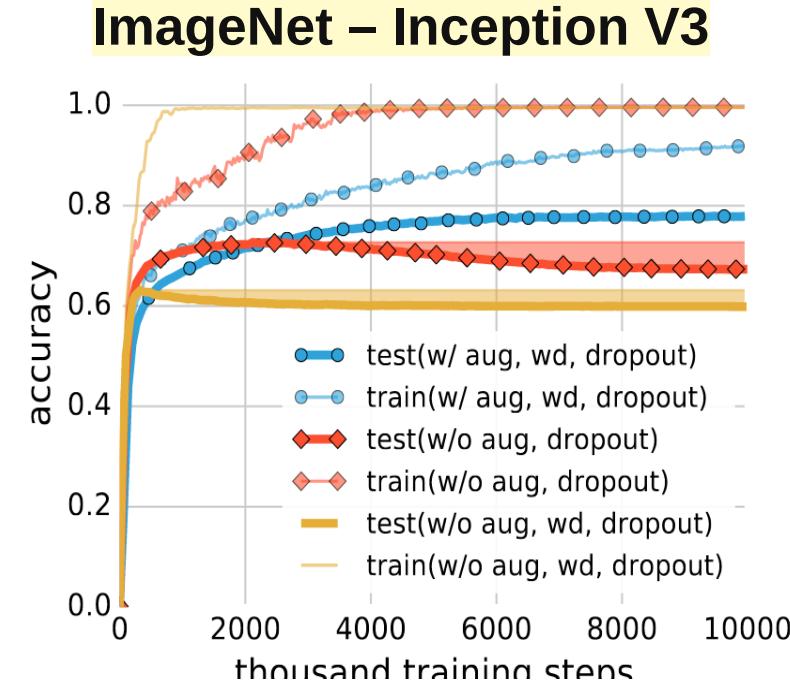
Dropout

Dropout+WD

Dropout+WD+DAug

No remarkable generalisation improvement

– It helps but is not a game-changer



Explicit Regularisation Effect

Dropout

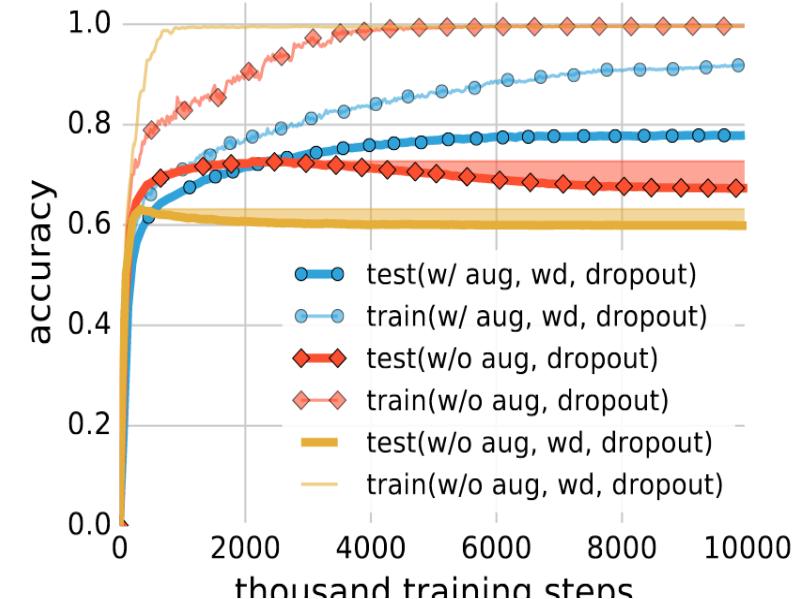
Dropout+WD

Dropout+WD+DAug

Dropout (alone)

- shatters the training data
- benefits from early stopping

ImageNet – Inception V3



Explicit Regularisation Effect

Dropout

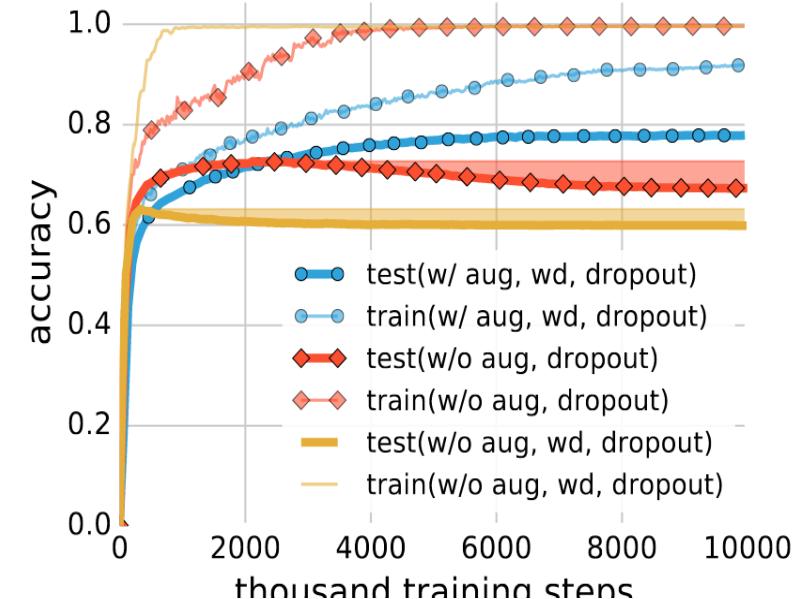
Dropout+WD

Dropout+WD+DAug

Dropout+WD

- shatters the training data
- benefits from early stopping
- **WD** is not necessarily useful!
*** Dropout+WD worse than Dropout**

ImageNet – Inception V3



Explicit Regularisation Effect

Dropout

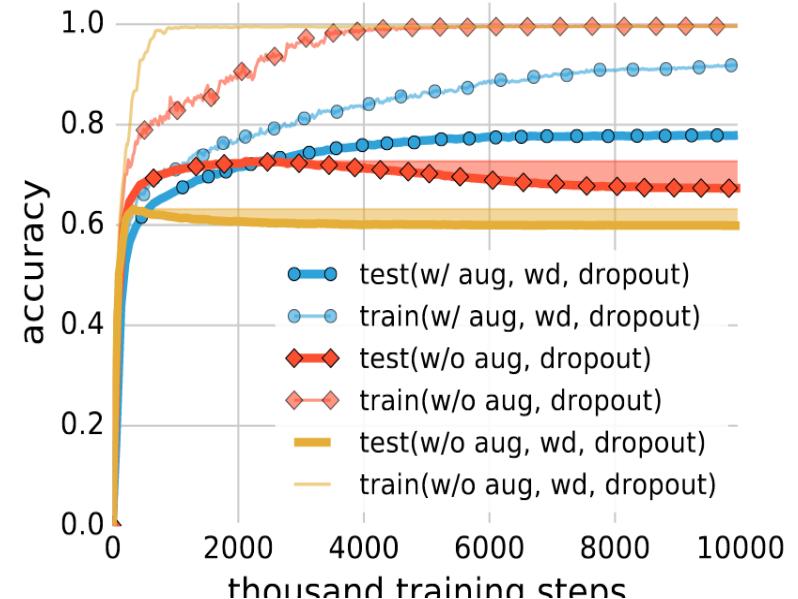
Dropout+WD

Dropout+WD+DAug

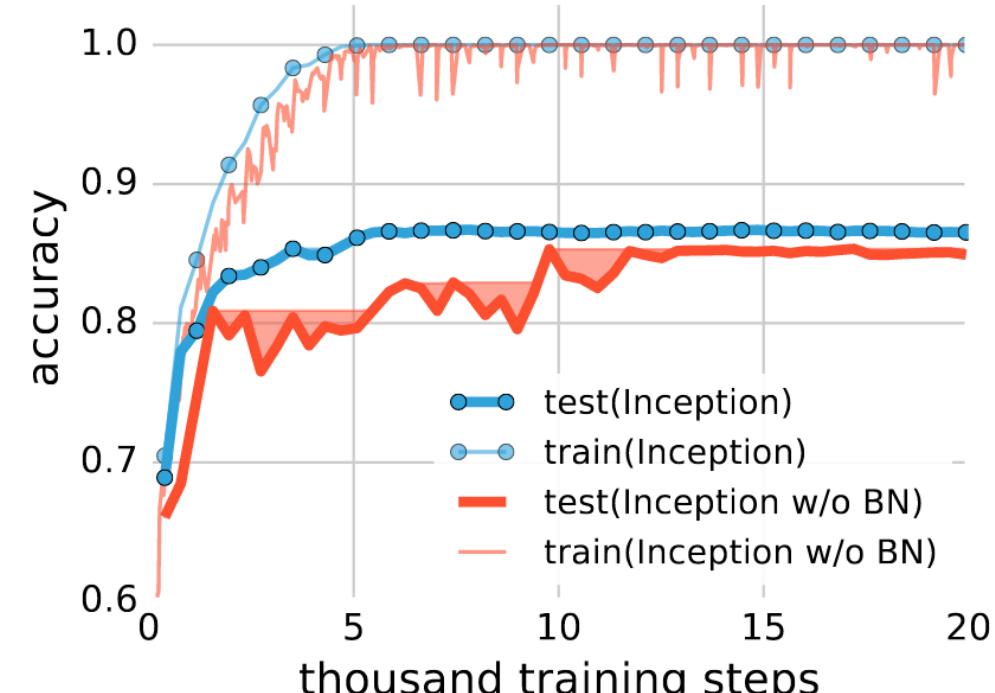
Dropout+WD+DAug

- does not shatter training data
- early stopping is not useful
- is the best after enough iterations

ImageNet – Inception V3



Batch Normalisation (BN) Effect

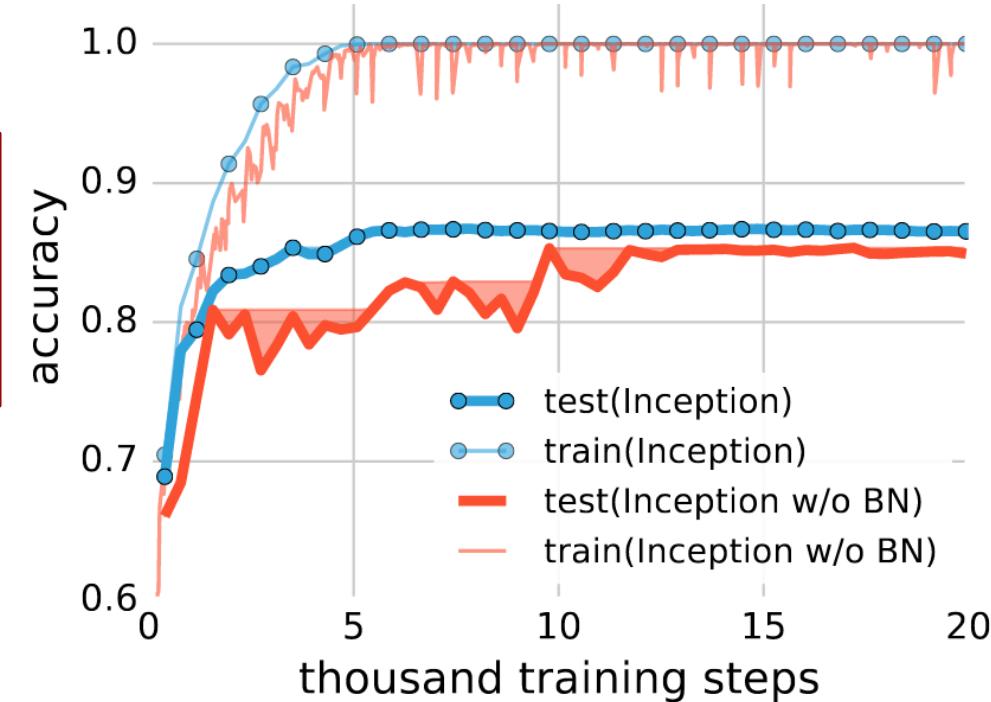


(b) Inception on CIFAR10

Batch Normalisation (BN) Effect

Makes training dynamic smoother

Up to 4% generalisation error reduction
[not much, after enough iterations]



(b) Inception on CIFAR10

Regularisation Effect on Accuracy

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Regularisation Effect on Accuracy

Almost all architectures **shatter**
the training set (~ 100% Accuracy).

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Regularisation Effect on Accuracy

Inception with BN

- No Reg. → 85.75%
- WD → +0.28
- DA → +3.56
- WD+DA → +3.30

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Regularisation Effect on Accuracy

Inception with BN

- No Reg. → 85.75%
- WD → +0.28
- DA → +3.56
- WD+DA → +3.30

Inception withOUT BN

- No Reg. → 82.0
- WD → +1.0

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Regularisation Effect on Accuracy

Alexnet

- No Reg. → 76.07%
- WD → +1.29
- DA → +2.96
- **WD+DA → +5.15**

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Regularisation Effect on Accuracy

MLP 3x512

- No Reg. → 52.39%
- WD → +0.96

MLP 1x512

- No Reg. → 50.51
- WD → -0.12

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Architecture Effect

Inception with BN → 85.75

Inception withOUT BN → 82.0

Alexnet → 76.07

MLP 3x512 → 52.39

MLP 1x512 → 50.51

Max Performance Improvement

- By Reg.: +3.56 (85.75 → 89.31)
- By Arch.: +35.24 (50.51 → 85.75)

CIFAR10

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	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	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	99.82	9.86
MLP 3x512	1,735,178	no	yes	100.0	53.35
		no	no	100.0	52.39
		(fitting random labels)	no	100.0	10.48
MLP 1x512	1,209,866	no	yes	99.80	50.39
		no	no	100.0	50.51
		(fitting random labels)	no	99.34	10.61

Architecture Effect

Inception with BN → 85.75

Inception without BN → 82.0

Alexnet → 76.07

MLP 3x512 → 52.39

MLP 1x512 → 50.51

Max Performance Improvement

- By Reg.: +3.56 (85.75 → 89.31)
- By Arch.: +35.24 (50.51 → 85.75)



“... while regularization is important, bigger gains can be achieved by simply changing the model architecture.”

Regularisation Effect on Accuracy

Inception V3 **shatters** ImageNet when
... **True labels** R used & **regularises** R off

ImageNet – Inception V3

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

Accuracy – ImageNet – Inception V3

Top-1

- No Reg → 59.80
- No Reg & ES → +3.36 (63.16)
- WD → +7.38
- WD & ES → +12.77(ES → +5.39)
- DA → +12.15
- DA+WD+DO → +18.04%

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

Accuracy – ImageNet – Inception V3

Top-5

- No Reg → 80.38
- No Reg & ES → +4.11 (84.49)
- WD → +6.06
- WD & ES → +10.93 (ES → +4.87)
- DA → +10.05
- DA+WD+DO → +13.54%

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

Accuracy – ImageNet – Inception V3

Top-5

- Alexnet (Challenge Winner) → 83.6
- Inception V3
 - No Reg. → 80.38
 - ES → 84.49
 - WD → 86.44
 - WD & ES → 91.31
 - DA → 90.43
 - DA+DO+WD → 93.2

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

Accuracy – ImageNet – Inception V3

Top-5

- Alexnet (Challenge Winner) → 83.6
- Inception V3
 - No Reg. → 80.38
 - ES → 84.49
 - WD → 86.44
 - WD & ES → 91.31
 - DA → 90.43
 - DA+DO+WD → 93.2

Win the challenges by spending time on ...

1. ARCHITECTURE, rather than REGULARISATION!
2. Data augmentation



Finite Sample Expressivity



Universal Approximation Theorem (UAT)

- A *MLP with one hidden layer can approximate any continuous function up to an arbitrary constant*
 - Proved by *Cybenko* for Sigmoidal Activation in 1989

Math. Control Signals Systems (1989) 2: 303–314

Mathematics of Control,
Signals, and Systems
© 1989 Springer-Verlag New York Inc.



Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our results settle an open question about representability in the class of single hidden layer neural networks. In particular, we show that arbitrary decision regions can be arbitrarily well approximated by continuous feedforward neural networks with only a single internal, hidden layer and any continuous sigmoidal nonlinearity. The paper discusses approximation properties of other possible types of nonlinearities that might be implemented by artificial neural networks.

Key words. Neural networks, Approximation, Completeness.

Universal Approximation Theorem (UAT)

- A *MLP with one hidden layer can approximate any continuous function up to an arbitrary constant*
 - Proved by *Hornik* for any bounded & non-constant function in 1991

Neural Networks, 4(2), 251–257,
1991

Approximation Capabilities of Multilayer Feedforward Networks

KURT HORNICK

Technische Universität Wien, Vienna, Austria

(Received 30 January 1990; revised and accepted 25 October 1990)



Abstract—We show that standard multilayer feedforward networks with as few as a single hidden layer and arbitrary bounded and nonconstant activation function are universal approximators with respect to $L^p(\mu)$ performance criteria, for arbitrary finite input environment measures μ , provided only that sufficiently many hidden units are available. If the activation function is continuous, bounded and nonconstant, then continuous mappings can be learned uniformly over compact input sets. We also give very general conditions ensuring that networks with sufficiently smooth activation functions are capable of arbitrarily accurate approximation to a function and its derivatives.

Keywords—Multilayer feedforward networks, Activation function, Universal approximation capabilities, Input environment measure, $L^p(\mu)$ approximation, Uniform approximation, Sobolev spaces, Smooth approximation.



Universal Approximation Theorem (UAT)

“... It is not the specific choice of the activation function, but rather the MLP architectures which give the NNs the potential of being universal function approximator.”

wikipedia





UAT – Some Points

- Approximation \equiv Fitting
 - No distinction between learning or memorising





UAT – Some Points

- Approximation \equiv Fitting
 - No distinction between learning or memorising
- Silent about algorithmic learnability of the weights
 - It's about existence, NOT how to learn



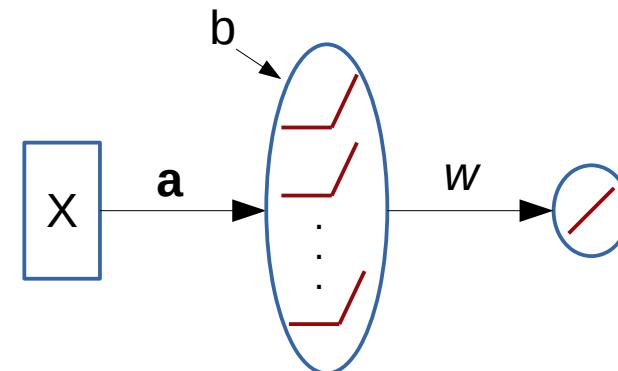
UAT – Some Points

- Approximation \equiv Fitting
 - No distinction between learning or memorising
- Silent about algorithmic learnability of the weights
 - It's about existence, NOT how to learn
- Defined on the *population* level
 - What about the “sample” level?

UAT@Finite-Sample Level

- **Theorem.** *There exists a two-layer NN with ReLU activation and $2n+d$ weights that can represent any function on a sample of size n in d dimensions.*

$$C(x) = \sum_{i=1}^n w_i \phi(x; a, b) = \sum_{i=1}^n w_i \text{ReLU}\{a^T x - b_i\}$$



UAT@Finite-Sample Level

- **Theorem.** *There exists a two-layer NN with ReLU activation and $2n+d$ weights that can represent any function on a sample of size n in d dimensions.*

Too flexible → overfitting
Any reason to generalise well?



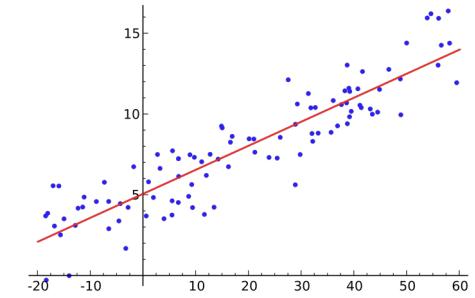
Appeal to Linear Systems



Overdetermined Linear Systems

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

- Known > Unknown, $n > d \equiv \# \text{parameters}$
 - No solution
 - Least Square Error (LSE)



$$\min_w \|Xw - y\|_2^2$$



Correlation matrix

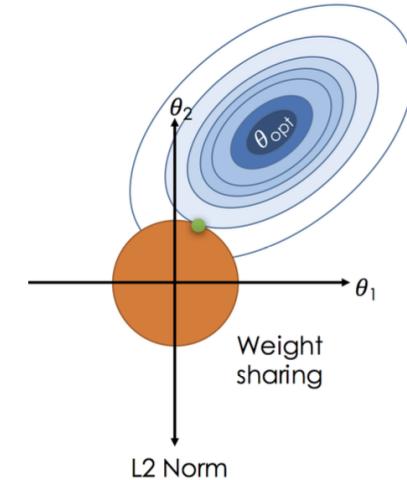
$$w = (X^T X)^{-1} X^T y$$

d x d

Underdetermined Linear Systems

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

- Over-parametrised, $n < d \equiv \# \text{parameters}$
 - Infinite solutions when X is full-rank
 - Minimum L₂-norm solution



$$\begin{aligned} \min_w \quad & \|w\|_2^2 \\ \text{s.t.} \quad & y = Xw \end{aligned}$$



Gram (Kernel) matrix

$$w = X^T (X X^T)^{-1} y$$

$\boxed{X X^T}$
 $n \times n$



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$w_d^{(t+1)} = w_d^{(t)} - \eta \nabla_{w_d} E = w_d^{(t)} - \eta e^{(t)} x_d^{(t)}$$





SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d} \quad E = e^T e = \|Xw - y\|_2^2$$

$$w_d^{(t+1)} = w_d^{(t)} - \eta \nabla_{w_d} E = w_d^{(t)} - \eta e^{(t)} x_d^{(t)}$$

$$w^{(t+1)} \Big|_{w^{(0)}=0} = X^T \alpha$$



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$w_d^{(t+1)} = w_d^{(t)} - \eta \nabla_{w_d} E = w_d^{(t)} - \eta e^{(t)} x_d^{(t)}$$

$$w^{(t+1)} \Big|_{w^{(0)}=0} = X^T \alpha$$

Weights lie in the span of data points



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$w^{(t+1)} \Big|_{w^{(0)}=0} = X^T \alpha \quad \longrightarrow \quad \begin{aligned} y &= Xw^{(t)} \\ &= XX^T \alpha \end{aligned}$$



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$w^{(t+1)} \Big|_{w^{(0)}=0} = X^T \alpha \quad \xrightarrow{\hspace{1cm}} \quad \begin{array}{l} \text{when } t \rightarrow \infty \text{ equality holds!} \\ \downarrow \\ y = Xw^{(t \rightarrow \infty)} \\ = XX^T \alpha \end{array}$$



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

square

$$y = \boxed{XX^T}\alpha \longrightarrow \alpha = (XX^T)^{-1}y \longrightarrow w \stackrel{t \rightarrow \infty}{=} X^T(XX^T)^{-1}y$$

Unique solution
for α



SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$y = \boxed{XX^T}\alpha \longrightarrow \alpha = (XX^T)^{-1}y \longrightarrow \boxed{w \stackrel{t \rightarrow \infty}{=} X^T(XX^T)^{-1}y}$$

Unique solution
for α

– SGD solution \equiv min L₂-norm solution for underdetermined linear system

SGD Solution for a Linear System

$$y = Xw, \quad X \in \mathbb{R}^{n \times d}$$

$$E = e^T e = \|Xw - y\|_2^2$$

$$y = \boxed{XX^T}\alpha \longrightarrow \alpha = (XX^T)^{-1}y \longrightarrow \boxed{w \stackrel{t \rightarrow \infty}{=} X^T(XX^T)^{-1}y}$$

Unique solution
for α

- SGD solution \equiv min L₂-norm solution for underdetermined linear system
- ====>>> SGD *implicitly* does some regularisation

SGD Implicit Regularisation

$$w_t \xrightarrow{t \rightarrow \infty} X^T (XX^T)^{-1}y$$

Fairly good generalisation, corroborates SGD performs some implicit regularisation.

Dataset	Preprocessing (Kernelise)	Test Error
MNIST	None	1.2%
	Gabor filters	0.6%
CIFAR-10	None	46%
	Random ConvNet	17%

SGD Implicit Regularisation – Caveat

$$w_t \xrightarrow{t \rightarrow \infty} X^T (X X^T)^{-1} y$$

UNFORTUNATELY,

Minimum L₂-norm is not predictive
of generalisation performance.

Dataset	Preprocessing (Kernelise)	Test Error	L ₂ - norm
MNIST	None	1.2%	220
	Gabor filters	0.6%	390
CIFAR-10	None	46%	-
	Random ConvNet	17%	-



Do we need the global optimum for a better generalisation?





Do we need the global optimum for a better generalisation?

Big Dumb Neural Nets: A Working Brute Force Approach to Speech Recognition

Nelson Morgan

ICNN'94

“... we essentially never reach the global minimum. In fact it is not even clear that we would want to reach the global minimum for the training data, since we wish to generalise to the best possible minimum for previously unseen data, which is what we will be using the net for.”

The Loss Surfaces of Multilayer Networks

Anna Choromanska Mikael Henaff Michael Mathieu Gérard Ben Arous Yann LeCun
achoroma@cims.nyu.edu mbh305@nyu.edu mathieu@cs.nyu.edu benarous@cims.nyu.edu yann@cs.nyu.edu

Courant Institute of Mathematical Sciences
New York, NY, USA

AISTATS'15

“Struggling to find the global minimum on training (as opposed to one of the many good local ones) is not useful in practice and may lead to overfitting.”



Wrap-up – Paper Contributions

Understanding Deep Learning Requires Rethinking Generalization

ICLR 2017

- Optimisation is easy, even when learning is impossible
- Investigating the role of regularisation
- Finite-sample expressivity of NNs
- SGD's implicit regularisation in linear models

Wrap-up – Conclusions

- Conventional wisdom about generalisation fails to explain why DNNs generalise well in practice
- #parameters does not necessarily mean model complexity
- [Large enough] DNNs easily fit (learn/memorise) [any, even random] labels
- Explicit regularisation may improve generalisation performance BUT is neither necessary nor sufficient for controlling generalisation error
- DNN optimisation continues to remain easy even when learning is hard or generalisation is poor
- SGD MAY act as an implicit regulariser
- Finding global optimum is not important for having a good generalisation



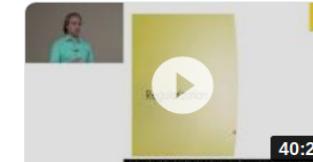
That's it!

- Thanks for Your Attention!
- Appendices
 - A1. References
 - A2. VC-dimension
 - A3. SGD Solution for a Linear System



References

- Paper
- Slides
- Poster
- Codes
- Presentations 
- Comments in OpenReview.net



VC Dimension



- A measure of hypothesis class *capacity*
- *Cardinality* of the largest set of points algorithm can *shatter*
 - Shatter: 0 training error for any label assignment
- Gives an upper bound for the generalisation error
 - Criticism: too loose
- Valid when #parameters << #training data (N)

$$P_{gen} \Big|_{E_{train} > 0} \leq O\left(\sqrt{\frac{VC_{dim}}{N}}\right)$$

$$P_{gen} \Big|_{E_{train} = 0} \leq O\left(\frac{VC_{dim}}{N}\right)$$